ABSTRACT

EVERETT, JARED S. Forward-Error Correction Coding for Underwater Free-Space Optical Communication. (Under the direction of Dr. Brian L. Hughes).

Wireless communication underwater is made challenging by the fact that radio frequency waves suffer such high attenuation in water that they can only propagate on the order of a few feet. This property makes them of little use in underwater communication. Underwater free-space optical (FSO) communication is an emerging technology that has the potential to offer high-data-rate communication over short-range wireless links in the challenging underwater environment. One well-established method of improving the performance of a communication system is through the use of forward-error correction (FEC). In this thesis, we investigate the application of FEC coding to the underwater FSO communication channel. Specifically, Reed-Solomon, turbo, and low-density parity-check (LDPC) codes are implemented and evaluated.

An underwater experimental testbed is used to measure coded system performance on an underwater FSO link operating at 405 nm. The system utilizes intensity-modulation and direct-detection to transmit on-off-keying, return-to-zero modulated data at a baud rate of 500 kbps. First, a (255,129) Reed-Solomon code is implemented and found to provide a coding gain of approximately 2.5 dB bit-energy-to-noise-density ratio. Second, state-of-the-art turbo codes from the Universal Mobile Telecommunications System (UMTS) standard and the Consultative Committee for Space Data Systems (CCSDS) standard are implemented for six different sets of parameters. Coding gains ranging from 6.8 dB to 9.5 dB are demonstrated for code rates ranging from $r = 1/2$ to $r = 1/6$. Finally, a state-of-the-art LDPC code from the second generation Digital Video Broadcasting satellite standard (DVB-S2) is implemented for four different sets of parameters. Coding gains ranging from 7.7 dB to 9.2 dB are demonstrated for code rates ranging from $r = 1/2$ to $r = 1/4$. In each case, coding gains are measured relative to the uncoded system at a bit-error rate of 10^{-4}. Based on measured attenuation coefficient values, potential range extension of an FEC-coded system is presented for three water types: clear ocean water, coastal ocean water, and turbid harbor water.

Forward-Error Correction Coding for Underwater Free-Space Optical Communication

by
Jared Scott Everett

A thesis submitted to the Graduate Faculty of
North Carolina State University
in partial fullfillment of the
requirements for the Degree of
Master of Science

Electrical Engineering

Raleigh, North Carolina

2009

APPROVED BY:

Dr. John F. Muth

Dr. J. Keith Townsend

Dr. Brian L. Hughes
Chair of Advisory Committee

DEDICATION

To my mother and Frank, for always encouraging me to "study hard and learn all you can," and helping me keep my sense of whimsy along the way.

BIOGRAPHY

Jared Scott Everett was born in Royal Oak, Michigan, United States on November 30, 1983. Ever since his fifth grade math teacher observed that he was a "good math thinker," Jared has gravitated towards math related subjects, which eventually lead him to pursue a career in Electrical Engineering. On May 12, 2007, he received his B.S. in Electrical Engineering, B.S. in Computer Engineering, and B.A. in Arts Applications in Music all from North Carolina State University, earning the highest honor of Summa Cum Laude. After being awarded a Dean's Fellowship, Jared continued to pursue a graduate degree at North Carolina State University where he conducted his masters research in the Wireless Systems Engineering (WiSE) Laboratory under the direction of Prof. Brian Hughes. Upon completion of his M.S. in Electrical Engineering, Jared will begin his professional career at the John's Hopkins University Applied Physics Laboratory as an associate member of the technical staff in the Communications and Networking Technology Group.

ACKNOWLEDGMENTS

First and foremost, I would like to thank my research advisor, Dr. Brian Hughes, for his guidance and support both in and out of the classroom, in graduate school and in undergrad. In particular, I am grateful to Dr. Hughes for welcoming me into the Wireless Systems Engineering (WiSE) Laboratory during a particularly difficult time in my graduate career. Your generosity and commitment to the education of aspiring engineers have truly made an impact in my life.

I would like to thank my committee members, Dr. John Muth and Dr. Keith Townsend, for their valuable time and input. Your comments, suggestions, and encouragement were greatly appreciated. I would also like to recognize the organizations that funded this project and made this research possible: Ambalux Corporation under Office of Naval Research STTR N00014-07-M-0308 and the National Science Foundation under grants CCF-0515164 and ECCS-0636603. Thanks also to my fellow graduate students from the WiSE lab and the underwater optical communications project, in particular: Carlo Domizioli for helping me get started on this project and assisting me along the way; Brandon Cochenour for his valuable insights into the world of underwater optics; Yuhan Dong for showing me how to access the ECE Grendel server, without which I would not have been able to finish the 1000s of processor hours needed for all the simulations; very special thanks to Jim Simpson and William Cox for spending endless hours running experiments for me. This would not have been possible without all your hard work!

Throughout the past two years, I have been extremely fortunate to be surrounded by a supportive family and amazing friends, and to *all* of them I extend my warmest gratitude. To my parents for their continued love and encouragement. To David for his dedicated friendship and enormous sense of humor. To Maya for always keeping me company during the writing process. And finally, to Nicole for listening to my ideas, proofreading, making me laugh, and being there through good and bad. Without your support and encouragement, I may not have made it through graduate school. Thank you for being the wonderful person you are.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

In recent years, the need for high-data-rate underwater wireless communication links has increased tremendously. What began as a technological requirement primarily for military applications has expanded to include significant applications in the commercial and scientific sectors as well. A fundamental challenge in underwater wireless communication is that radio frequency (RF) links, which have enabled high-data-rate mobile communication in terrestrial applications, suffer such high attenuation under water that they can only propagate on the order of a few feet, thus making them of little use for communicating underwater. One attractive option for wireless communication in the underwater environment is *underwater free-space optical (FSO) communication*, which refers to an underwater optical link, typically operating near the blue-green region of the visible spectrum, that uses water as the propagation medium (i.e., no fiber). This emerging technology shows the potential to augment existing underwater communication methods by providing high-data-rate wireless communication over short-range links.

At present, the most common options for communication underwater are tethered communication and acoustic communication. Tethered communication refers to the use of a traditional transmission line or fiber-optic cable to transmit information to a shore station, surface vessel, or buoy. The resulting limits in mobility, however, make this infeasible for mobile applications, such as communication between submarines or autonomous unmanned vehicles (AUVs). Furthermore, the tether is vulnerable to the environment and prone to breakage over time. Underwater acoustic communication, in which acoustic waves are used to communicate wirelessly, has been the subject of much research since the early 1980s [1].

These systems have seen significant improvement over the past three decades. However, they remain inherently bandlimited due to absorption and low operating frequencies (most systems operate below 30 kHz). They are also subject to severe multipath fading and frequency spreading due to Doppler shifts. Typical bandwidths are on the order of >100 kHz over short-range links (<100 m), $10 - 100$ kHz over medium-range links (0.1-10 km), and <10 kHz over long-range links (>10 km) [2]. For applications in which low data rates are acceptable, underwater acoustic links are a good candidate, particularly if long-range links are desired. However, if both high data rate and mobility are required, a method other than tethered and acoustic communication is necessary.

Although it is not likely to displace underwater acoustic communication for long-range wireless links, underwater free-space optical communication is a promising new technology for short-range applications, offering high data rates over distances less than 100 m. Hanson and Radic [3] demonstrated a data rate of 1 Gbps over a distance of 2 meters in a laboratory environment and used Monte Carlo simulation to suggest that data rates of 1 Gbps over a range of 48 m could be achievable in clear ocean water conditions. Chancey [4] used theoretical modeling of the underwater optical channel to suggest that data rates of >10 Mbps could be achieved over longer data links of up to 100 m, also in clear ocean water conditions. In addition to higher data rates, underwater FSO offers other advantages over acoustic communication. While underwater acoustic systems are generally broadcast in nature, underwater FSO links are typically line-of-sight and point-to-point. This property makes underwater optical systems inherently more secure and harder to intercept, which is particularly appealing in military applications where covertness is required. This property also allows for multiple links in relatively close proximity, making full-duplex communication a possibility. Additionally, underwater optical links are not subject to the same level of multipath fading and frequency spread as underwater acoustic links, thus simplifying the digital signal processing of the receiver.

There are many scientific, commercial, and military applications in which high-data-rate, short-range underwater FSO links would be useful. For example, these links have the ability to improve the safety of professional and recreational ocean diving by eliminating the need for cumbersome tethered communication cables that can become snagged or broken, creating a potentially dangerous situation [5]. In addition to diver-to-diver and diver-to-vessel communication, underwater FSO links could be used to enable high-data-rate

communication between vessels, submarines, and AUVs.

Another important application is underwater wireless sensor networks, for which the possible applications are virtually limitless. These networks can include stationary nodes, mobile notes attached to a submarine or AUV, or a combination of both stationary and mobile nodes, as in [6]. Such sensor networks could be used for military/security purposes to monitor for underwater intruders. Terrestrial uses of wireless sensor networks include monitoring of environmental conditions including temperature change, water runoff, and pollution [7], [8]. It is certainly within reason to expect that ocean temperature, pollutants, and other measures of ecosystem health could be monitored using an underwater wireless sensor network. In particular, this technology would enable scientific observation in areas that humans and large vessels cannot easily navigate, such as deep-sea vents or underwater geological fault lines. Underwater FSO links could be an enabling technology, allowing mobile nodes in the form of AUVs to collect vast amounts of scientific data, then bring the data back to a central storage location and quickly download the data via a high-data-rate optical modem, a concept proposed in [9].

An underwater FSO link is limited by transmitter power and water turbidity. The term *turbidity* refers to the cloudiness or haziness of a liquid caused by suspended small particles. One well-established method of improving the performance of a communication system is through the use of forward-error correction (FEC). Forward-error correction is a method of error-control coding in which redundant bits are systematically introduced into the transmitted sequence in such a way that the receiver can correct a limited number of errors in the received message. Properly designed FEC coding schemes improve the power efficiency of a system, typically at the cost of decreased channel utilization. For an underwater FSO link, this translates into lower transmitter power requirements or extended link range. Today's state-of-the-art FEC codes, in particular turbo codes and low-density parity-check (LDPC) codes, can achieve performance close to the optimal power efficiency dictated by information theory for a given bandwidth efficiency [10], [11]. Although the advantages of FEC have been investigated for a wide array of communication channels, little has been done so far to investigate the benefits of FEC on an underwater FSO link.

This thesis focuses on the novel application of FEC coding to an underwater FSO link. The primary contribution of this thesis is to experimentally demonstrate the implementation of Reed-Solomon coding, turbo coding, and low-density parity-check (LDPC)

coding on an underwater FSO link using an underwater testbed developed by Cox and Simpson in [12] and [13]. For each of these codes, we will present experimental coding gains in power efficiency and attempt to predict what these gains could mean in terms of range extension of the system. A secondary contribution of this thesis is to begin to establish a theoretical model for the underwater free-space optical channel. This is done in two ways. First, a statistical analysis of the channel noise is presented in order to develop the channel model in Chapter 3. Second, the theoretical performance based on the channel model is compared with experimental results to demonstrate how the experimental data agrees with and differs from theory.

Chapter 2

Background

In this chapter we present an overview of forward-error correction. An understanding of this subject will be necessary in the application of FEC to the underwater optical communication channel. We begin by reviewing the fundamentals of forward-error correction in Section 2.1, including an introduction to block codes and convolutional codes. Then we present the fundamental theory behind each of the three types of codes implemented in this thesis. This includes Reed-Solomon codes in Section 2.2, turbo codes in Section 2.3, and finally low-density parity-check (LDPC) codes in Section 2.4.

2.1 Forward-Error Correction

In his landmark 1948 work, Claude Shannon showed that information can be communicated across a noisy channel with an arbitrarily small probability of error as long as the information is transmitted at a rate less than the capacity of the channel [14]. He proved this using an argument based on random coding. Although Shannon's work provided limits on reliable communication, it did not provide any details on how to construct a system to achieve those limits. Thus, the field of error-control coding was developed in an attempt to achieve the promise of Shannon's work. Error-control coding typically refers to two types of coding: forward-error correction (FEC) and error detection. Forward-error correction, sometimes referred to simply as "error correction," refers to the structured addition of redundancy to a data sequence at the transmitter in such a way that the receiver is able to correct a small number of symbol errors in the data sequence introduced by a noisy

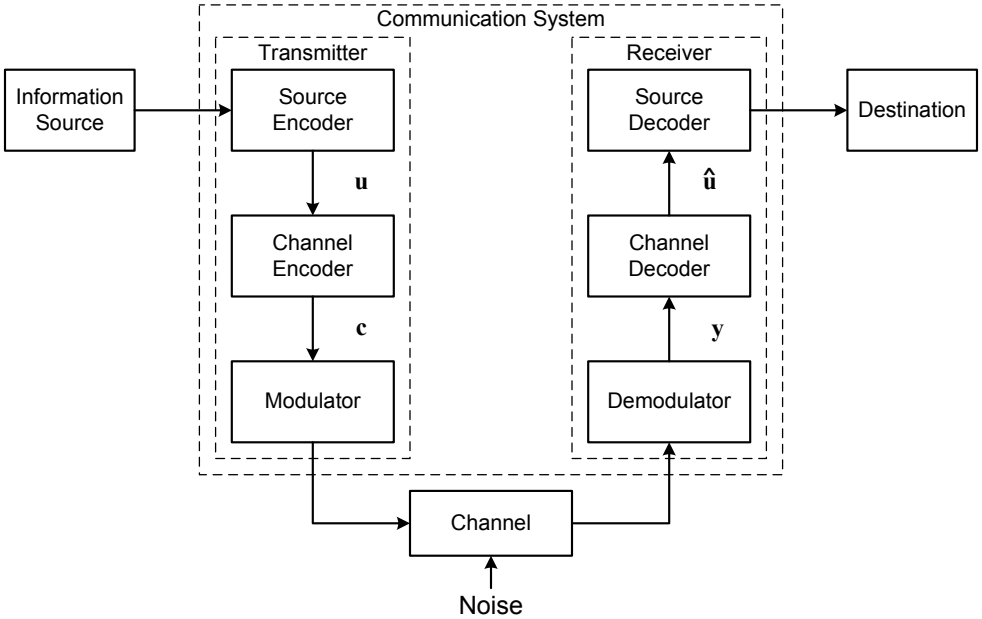


Figure 2.1: Block diagram of a typical digital communication system including forward-error correction.

channel without the use of feedback. Error detection is similar, but with the ultimate goal of detecting symbol errors in the received data sequence. For the purposes of this thesis, we are interested only in the former.

A block diagram of a typical digital communication system including FEC coding is shown in Figure 2.1. The information source can be either a person or a machine (e.g., a computer, or a data terminal). The output of the information source can be in the form of a continuous waveform or a sequence of discrete symbols. The source encoder transforms the source information into a sequence of bits called the *information sequence*, $\mathbf{u}$. The channel encoder transforms the information sequence into a discrete sequence of encoded symbols, $\mathbf{c}$, called a *codeword*. Typically, the codeword is also expressed as a sequence of bits. However, in some classes of codes, such as Reed-Solomon codes, non-binary symbols are used. Next, the modulator transforms the encoded symbols into waveforms that can be transmitted across the channel. In the channel, the waveforms are corrupted by noise. The demodulator processes the corrupted received waveforms, typically by means of a correlator of matched filter receiver, and produces a sequence of either discrete (quantized) or continuous (unquantized) outputs called the *received sequence*, $\mathbf{y}$. The channel decoder

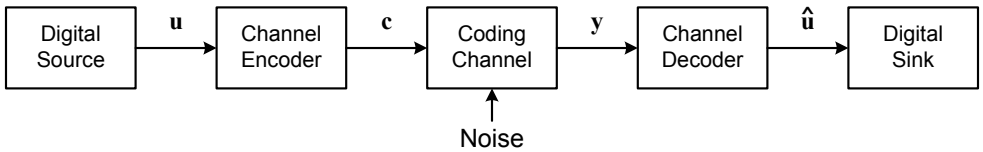


Figure 2.2: Simplified block diagram of a coding system.

attempts to use the redundancy introduced by the channel encoder to correct errors, thus producing the *estimated information sequence*, $\hat{\mathbf{u}}$. Ideally, $\hat{\mathbf{u}}$ will be an exact replica of $\mathbf{u}$; however, decoding errors may occur. Finally, the source decoder transforms $\hat{\mathbf{u}}$ into an estimate of the original information source. This estimate is then delivered at the destination.

From a coding standpoint, the system can be simplified in three ways. First, since we are only concerned with the information sequence, $\mathbf{u}$, and not the form of the information source, the information source and source encoder can be reduced to a single *digital source* block. Second, the destination and source decoder can similarly be reduced to a *digital sink* that takes as its input the binary sequence, $\hat{\mathbf{u}}$. Again, the context of the received information is unimportant. Third, the modulator, demodulator, and analog channel can be combined into a single block that we will call the discrete-time *coding channel*. The simplified block diagram of a coding system is shown in Figure 2.2. This block diagram allows us to focus on the system from an error-control coding standpoint.

The question of how to implement the channel encoder and decoder is one of the fundamental questions of error-control coding. In this section we will discuss the basic approaches to solving this problem. Most FEC codes known today fall into two main categories: *Block codes* and *Convolutional codes* (also referred to as *trellis codes*). These two structurally different types of codes will be introduced below.

2.1.1 Block Codes

The concept of coding data in blocks dates back to Shannon's original 1948 work [14]. In block coding, a message block of k information symbols is uniquely mapped to n coded channel symbols, which make up a codeword. A value of particular importance is the code rate, r which is defined as

$$r = \frac{k}{n} \ , \tag{2.1}$$

Table 2.1: The $(7, 4)$ Hamming code: a binary linear block code with $k = 4$ and $n = 7$.

Message, u	**Codeword, c**
0000	0000000
1000	0111000
0100	1100100
1100	1011100
0010	1010010
1010	1101010
0110	0110110
1110	0001110
0001	1110001
1001	1001001
0101	0010101
1101	0101101
0011	0100011
1011	0011011
0111	1000111
1111	1111111

the fraction of information symbols to coded symbols. Typically $k < n$ for practical codes. A basic example of a linear binary block code is given in Table 2.1.

For *binary* block codes, one symbol corresponds to one bit and operations are performed over the binary Galois field $GF(2)$. A Galois field, also referred to as a finite field, consists of a finite set of elements F on which two operations, addition "+" and multiplication "$\cdot$", are defined such that $(F, +)$ and $(F^{\times}, \cdot)$ are each commutative groups, where $F^{\times}$ are all the non-zero elements in F. The binary Galois field, $GF(2)$ consists of the elements $0, 1$ and the operations of modulo-2 addition and multiplication. Now, let $GF(2)^n$ be the n-dimensional vector space over $GF(2)$. An (n, k) binary linear block code, $\mathcal{C}$, is then defined as the k-dimensional subspace of $GF(2)^n$ that is spanned by the k linearly independent basis vectors $\mathbf{g}_0, \mathbf{g}_1, \ldots, \mathbf{g}_{k-1}$. Consider the message $\mathbf{u} = [u_0, u_1, \cdots, u_{k-1}]$ and corresponding codeword $\mathbf{c} = [c_0, c_1, \cdots, c_{n-1}]$ in code $\mathcal{C}$. The mapping $\mathbf{u} \rightarrow \mathbf{c}$ can be written as

$$\mathbf{c} = u_0\mathbf{g}_0 + u_1\mathbf{g}_1 + \cdots + u_{k-1}\mathbf{g}_{k-1}. \tag{2.2}$$

This relationship can also be represented in matrix form, $\mathbf{c} = \mathbf{u}G$. Here G is a $k \times n$ matrix,

the rows of which are the basis vectors of $\mathcal{C}$:

$$G = \begin{bmatrix} \mathbf{g}_0 \\ \mathbf{g}_1 \\ \vdots \\ \mathbf{g}_{k-1} \end{bmatrix}. \tag{2.3}$$

This matrix is called the *generator matrix*, G, of code $\mathcal{C}$. The codewords of $\mathcal{C}$ are given by all the linear combinations of rows in G. The encoding process can be viewed as a one-to-one mapping of messages in the k-dimensional message vector space onto codewords in the n-dimensional vector space defined by $\mathcal{C}$. Furthermore, a linear block code is said to be *systematic* if the information symbols, $\mathbf{u}$, appear explicitly in the codeword. In particular, if the information symbols appear at the end of the codeword then the generator matrix takes the form

$$G = \begin{bmatrix} P & I_k \end{bmatrix} \tag{2.4}$$

where P is a $k \times (n-k)$ matrix of parity-check symbols, and I_k is the $k \times k$ identity matrix.

Since $\mathcal{C}$ is a subspace of the larger vector space $GF(2)^n$, there exists a dual space $\mathcal{C}^\perp$ that consists of all vectors in $GF(2)^n$ that are orthogonal to every vector in $\mathcal{C}$. This dual space is also a linear block code of dimension $n-k$ that is spanned by a set of linearly independent basis vectors, say $\mathbf{h}_0, \mathbf{h}_1, \ldots, \mathbf{h}_{n-k-1}$. The dual space $\mathcal{C}^\perp$ has its own generator matrix

$$H = \begin{bmatrix} \mathbf{h}_0 \\ \mathbf{h}_1 \\ \vdots \\ \mathbf{h}_{n-k-1} \end{bmatrix}. \tag{2.5}$$

This matrix is called the *parity-check matrix*, H, of code $\mathcal{C}$. Since the dual space $\mathcal{C}^\perp$ is orthogonal to the code $\mathcal{C}$, the multiplication of any valid codeword of $\mathcal{C}$ with any basis vector of $\mathcal{C}^\perp$ will result in the all zero vector. This leads to the following property of the parity-check matrix:

$$\mathbf{c}H^T = \mathbf{0}, \tag{2.6}$$

where $\mathbf{c}$ is any valid codeword of $\mathcal{C}$, H^T is the transpose of the parity-check matrix H, and $\mathbf{0}$ is the all zero vector. An example of a generator matrix and corresponding parity-check matrix is given in Figure 2.3. In this case the generator matrix is in systematic form.

$$G = \begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix} \qquad H = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 & 1 \end{bmatrix}$$

Figure 2.3: Generator and parity-check matrices for the $(7, 4)$ Hamming code.

Decoding of linear block codes is often done using minimum Hamming distance decoding. The *Hamming distance* between any two vectors binary $\mathbf{x}$ and $\mathbf{y}$ refers to the number of places where $\mathbf{x}$ and $\mathbf{y}$ differ. Consider the case where the n-dimensional received vector, $\mathbf{y}$, at the output of the channel is corrupted by noise. Although this vector exists within $GF(2)^n$, it may not exist within the subspace $\mathcal{C}$ of the code. In this case, the decoder will estimate the transmitted codeword $\mathbf{c}$ by the codeword $\hat{\mathbf{c}} \in \mathcal{C}$ that is closest to $\mathbf{y}$ in terms of Hamming distance. As long as the number of errors is low enough that $\mathbf{y}$ is closest to the transmitted codeword $\mathbf{c}$, no error will occur in the decoded message $\hat{\mathbf{u}}$. This concept can be visualized by recalling that each possible received vector, $\mathbf{y}$, is a point in the n-dimensional vector space $GF(2)^n$. The codewords of $\mathcal{C}$ make up a subset of these points. Around each valid codeword is a sphere such that no two spheres are overlapping. This is accomplished if the radius of the sphere is one less than half of the minimum distance between any two spheres. All points (i.e. received vectors) within a sphere are decoded as the codeword at its center. A diagram of this concept is shown in Figure 2.4.

The ability of a linear block code to correct errors is closely related to its minimum distance d_{min}, which is defined as the minimum Hamming distance between any two distinct codewords of the code. A code with minimum distance d_{min} can correct any pattern of t symbol errors provided that

$$d_{min} \geq 2t + 1 \ . \tag{2.7}$$

If more than t errors occur, then the received vector may be pushed out of one codeword's "sphere" and into another, resulting in a decoding error.

Cyclic codes are a class of linear block codes which have the property that a cyclic shift of any codeword is also a codeword. This means that if $\mathbf{c} = [c_0, c_1, \cdots, c_{n-1}]$ is any codeword of $\mathcal{C}$, then $\mathbf{c}^{(1)} = [c_{n-1}, c_0, c_1, \cdots, c_{n-2}]$ is also a codeword. Cyclic codes are important in practice because encoding and decoding can be implemented using sequential

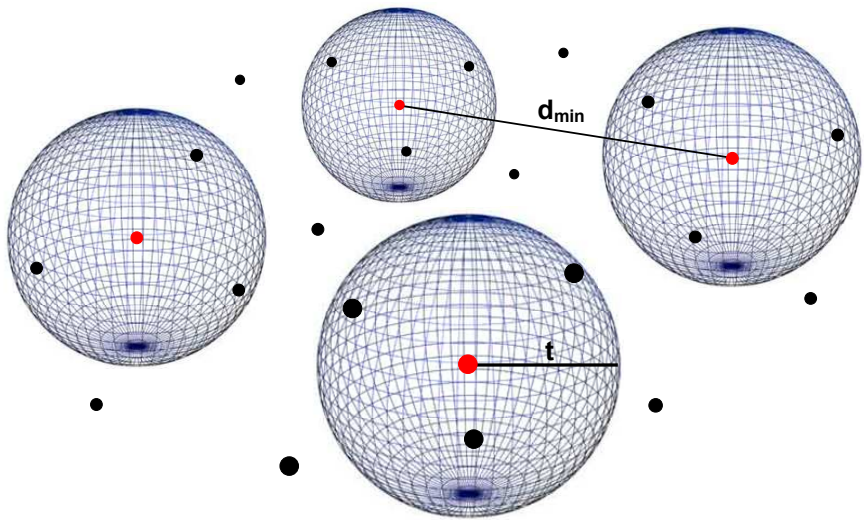


Figure 2.4: A representation of codewords as centers of spheres of radius $t = \lfloor 1/2\,(d_{min} - 1) \rfloor$ in the n-dimensional vector space $GF(2)^n$.

logic or shift registers and because their special algebraic structure leads to more efficient decoding algorithms [15]. In addition to being expressed in vector form, cyclic codes can also be equivalently expressed as polynomials:

$$\mathbf{c} = [c_0, c_1, \cdots, c_{n-1}] \qquad \leftrightarrow \qquad c(x) = c_0 + c_1 x + c_2 x^2 + \cdots + c_{n-1} x^{n-1} \ ,$$

where x is an indeterminant variable.

In addition to the generator and parity-check matrices of regular block codes, cyclic codes can also be defined and generated by means of a *generator polynomial*, $g(x)$. The generator polynomial is the unique, non-zero minimum degree code polynomial that divides all other code polynomials in the code. That is to say, any code polynomial in the code is a multiple of the code's generator polynomial. This results in a useful encoding property of cyclic codewords. If the message is expressed in polynomial form, then the corresponding codeword can be generated by

$$c(x) = u(x)g(x) \tag{2.8}$$

where $c(x)$ is the codeword polynomial, $u(x) = u_0 + u_1 x + \cdots + u_{x-1} x^{k-1}$ is the message polynomial, and $g(x)$ is the generator polynomial of the code. A basic understanding of cyclic codes will be useful in the discussion of Reed-Solomon codes, a powerful class of cyclic linear block codes, later in this chapter.

2.1.2 Convolutional Codes

Convolutional codes, which function in a manner fundamentally different than block codes, were first proposed by Elias in 1955 [16]. Convolutional encoders also create n output bits for every k input bits; however, instead of using blocks, encoding is performed using a sliding window of data. In this case, k and n are small integers and, similar to block codes, $k < n$ in practical codes. The encoded message now depends on both the current input message, comprised of k symbols, and m previous messages, each of which is also k symbols in length. This value, m, is the *memory order* of the code. For convolutional codes, redundancy is a function of the memory order of the encoder, as well as the number of parity bits in the codes. Another important parameter is the *constraint length* of the code, denoted by K, which is equivalent to $m + 1$. For nonrecursive convolutional codes, such as the one shown in Figure 2.5 below, the constraint length indicates the maximum number of input symbols (past and present) that any of the n output symbols can depend on. The impulse responses of the encoder can last at most K time units and are written as

$$\mathbf{g}^{(ij)} = \left[g_0^{(ij)}, g_1^{(ij)}, \cdots, g_m^{(ij)}\right] \ , \tag{2.9}$$

where $\mathbf{g}^{(ij)}$ refers to the impulse response between in i-th input symbol, u_i, and the j-th output symbol, c_j. The impulse responses $\mathbf{g}^{(ij)}$ are called the *generator sequences*, or simply the *generators*, of the encoder. In tables, the generators are often expressed in octal, with zeros padded on the left if necessary.

A simple example of a binary convolutional encoder is given in Figure 2.5. The data sequence, $\mathbf{u}$, enters from the left and is stored in the binary linear shift register. Each time a new data bit arrives at the input to the encoder, the previous data bits are shifted to the right into the next flip-flop. At the output, two code bits are generated, c_i^1 and c_i^2, for

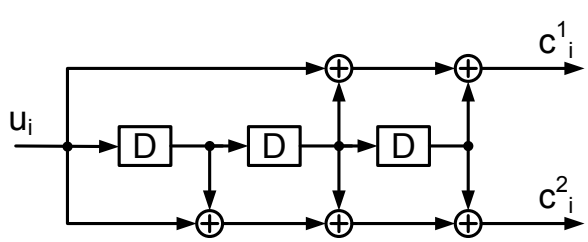


Figure 2.5: Example of a binary convolutional encoder with $k = 1$, $n = 2$, and $m = 3$.

every input data bit. These bits are multiplexed together to form the output $c_i = \begin{bmatrix} c_i^1 & c_i^2 \end{bmatrix}$. For this example, $k = 1$ and $n = 2$ making this a rate $r = 1/2$ code. The memory order and constraint length are $m = 3$ and $K = 4$ respectively. The generator sequences, read left to right, are $\mathbf{g}^{(11)} = 1011$ and $\mathbf{g}^{(12)} = 1111$. Alternatively, the generators can be expressed in octal as $\mathbf{g}^{(11)} = 13$ and $\mathbf{g}^{(12)} = 17$. Since the input sequence does not appear at the output, this encoder is not systematic. For this reason, conventional convolutional codes of this structure are sometimes referred to as *non-systematic convolutional codes* or NSC codes.

One major advantage of convolutional codes is that computationally efficient decoding algorithms that exploit soft-decision information are known, such as the famous Viterbi algorithm [17], [18]. Since our discussion of convolutional codes is limited in application to Turbo codes, which have a very distinct decoding structure, we will not discuss the decoding of convolutional codes in detail here. However, there are a number of excellent texts on the subject, such as [15, pp. 515-738].

2.1.3 Concatenated Codes

Multiple coding schemes can be used simultaneously to improve error correction capability. In this case, two relatively small, low-complexity constituent codes are used together to create a "big code" that is more powerful than the individual codes alone. This technique is called the *concatenation* of codes. By breaking the encoding and decoding process into two smaller parts, concatenated codes have the advantage that they can provide the same performance as a single, big code but typically at a lower cost in terms of complexity. There are basically two ways of concatenating codes: *serial concatenation* and *parallel concatenation*.

Serial concatenation was first introduced by David Forney in 1966 [19]. In this method, the encoder, physical channel, and decoder are grouped together into a *super channel* or *outer channel*. A second encoder and decoder are then used on this super channel. The encoder/decoder pair acting on the outer channel constitute the outer code, $\mathcal{C}_1$. The encoder/decoder pair acting on the physical, or inner, channel comprise the inner code, $\mathcal{C}_2$. The message block of length k is first encoded by the outer encoder, which generates a code vector of length n_1. This code vector is then the input of the inner encoder, which generates a code vector of length n_2 which is transmitted across the physical channel. After transmission, the inner decoder decodes the message first, producing a decoded vector of

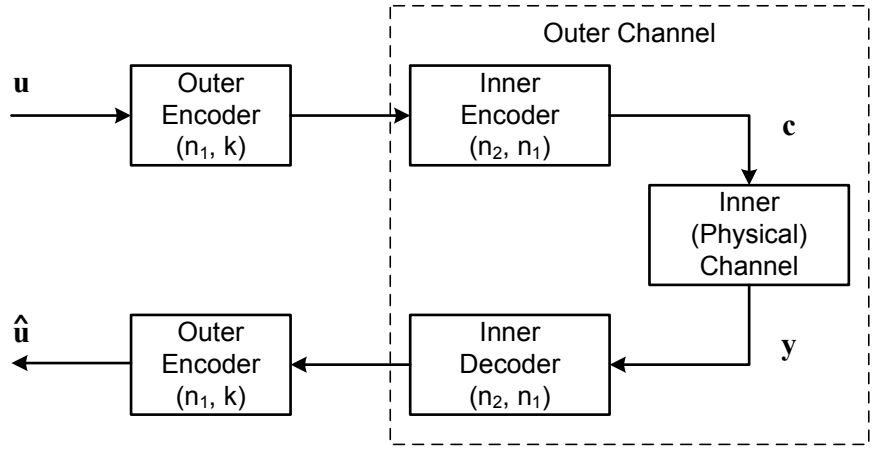


Figure 2.6: Structure of a serial concatenated system.

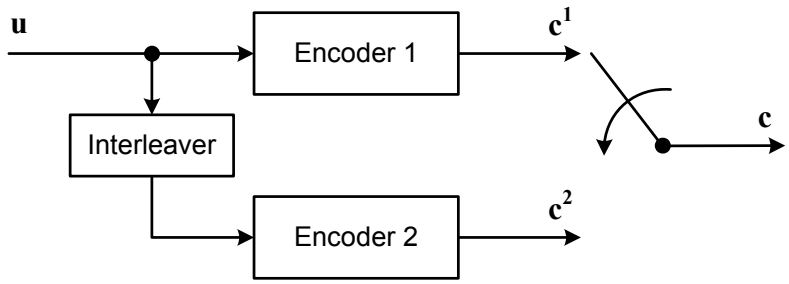


Figure 2.7: Structure of a parallel concatenated system.

length n_1 which is then decoded by the outer decoder into the received message $\hat{\mathbf{u}}$. The structure of a serial concatenated system is shown in Figure 2.6. A widely used example of a serial concatenated code is the compact disk standard, which utilizes two shortened Reed-Solomon codes, $\mathcal{C}_{RS}\,(28, 24)$ and $\mathcal{C}_{RS}\,(32, 28)$, separated by an interleaver.

A second concatenation method, called parallel concatenation, was first introduced by Berrou, Glavieux, and Thitimajshima in 1993 as a part of their turbo coding scheme [20]. In this method, the input message is encoded by both the first and second constituent encoders in parallel. The input to the second encoder is typically interleaved before encoding to ensure that the outputs of the two encoders will be substantially different. Finally, the outputs of the two codes are multiplexed together before being transmitted over the channel. The manner in which they are multiplexed depends on the number of outputs of each code and the structure of the parallel concatenated code. However, it is typical for the

overall multiplexed output to alternate bit by bit between the first and second encoder. A diagram of a typical parallel concatenated coding system is shown in Figure 2.7. Parallel concatenation is an essential component of turbo coding systems. An understanding of the concept of parallel concatenation will be useful in our discussion of turbo codes in section 2.3.

2.2 Reed-Solomon Codes

Reed-Solomon codes are a class of non-binary cyclic block codes that were first proposed in 1960 [21]. This class of codes includes some of the most powerful known block codes, and simple decoding algorithms for these codes are known. Reed-Solomon codes are commonly used in serial concatenated systems as the outer code and are also used in computer memories and compact disc storage.

In Section 2.1.1 we focused primarily on *binary* block codes. Reed-Solomon codes, however, are *non-binary*. This means that they are defined over the Galois field $GF(q)$, where $q > 2$, rather than the binary field $GF(2)$. An (n, k) non-binary block code over $GF(q)$ maps k q-ary message symbols onto n q-ary code symbols. In general, q is chosen such that $q = p^m$ where p is any prime number. In practice, however, the most useful codes are of the form $q = 2^m$. Thus, a non-binary Reed-Solomon code $\mathcal{C}_{RS}(n, k)$ over $GF(2^m)$ maps k m-bit message bytes onto n m-bit code bytes. In other words, the $\mathcal{C}_{RS}(n, k)$ q-ary code can be regarded as an equivalent (nm, km) binary code.

A Reed-Solomon code over $GF(q)$ can be defined by its parity-check matrix, which takes on a special structure given by

$$H = \begin{bmatrix} \alpha_1 & \alpha_2 & \cdots & \alpha_n \\ \alpha_1^2 & \alpha_2^2 & \cdots & \alpha_n^2 \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_1^{2t} & \alpha_2^{2t} & \cdots & \alpha_n^{2t} \end{bmatrix} \tag{2.10}$$

where $\alpha_1, \alpha_2, \cdots, \alpha_n$ are all the distinct non-zero elements of $GF(q)$ and t is the maximum number of errors corrected by the code, where $t \leq n/2$. A matrix of this form is called a *Vandermonde matrix*. Reed-Solomon codes have block length $n = q - 1$ and dimension $k = q - 1 - 2t$. Since k can be chosen for a specific value of t, is is easy to "choose" the

error correction capability of a Reed-Solomon code. All error-correcting codes must satisfy the Singleton bound, which is given by

$$d_{min} \leq n - k + 1 \tag{2.11}$$

In the case of Reed-Solomon codes, $d_{min} = n-k+1$. Since Reed-Solomon codes achieve the Singleton bound, they are said to be *maximum distance separable*. This is a very desirable property to have in a code because it allows for the maximum number of correctable errors, t, for a given dimension, k, and block length, n.

Since Reed-Solomon codes are cyclic, they can alternatively be defined in terms of their generator polynomial. If α is a primitive element over $GF(q)$, then $\alpha^{q-1} = 1$. The $\mathcal{C}_{RS}(n,k)$ code is then the cyclic linear block code generated by the polynomial

$$\begin{aligned} g(x) &= (x-\alpha)(x-\alpha^2)\cdots(x-\alpha^{n-k}) && (2.12)\\ &= (x-\alpha)(x-\alpha^2)\cdots(x-\alpha^{2t}) && (2.13)\\ &= g_0 + g_1 x + g_2 x^2 + \cdots + g_{2t} x^{2t} && (2.14) \end{aligned}$$

where the coefficients, g_i, are over the field $GF(2^m)$. The polynomial form of Reed-Solomon codes is useful for systematic encoding. The systematic encoding of Reed-Solomon codes can be performed polynomial division. Let $p(x)$ be the remainder when $x^{2t}u(x)$ is divided by $g(x)$, so

$$x^{2t}u(x) = q(x)g(x) + p(x) \ , \tag{2.15}$$

where $u(x)$ is the message polynomial to be encoded and $q(x)$ is the quotient. The systematic code polynomial is formed by subtracting this remainder from the polynomial $x^{2t}u(x)$, i.e., $c(x) = x^{2t}u(x) - p(x)$. As an example, consider the message $\mathbf{u} = [001 \quad 101 \quad 111 \quad 010 \quad 011]$ encoded using the $\mathcal{C}_{RS}(7,5)$ code that operates over the Galois field $GF(2^3)$, generated from the primitive polynomial $p_i(x) = 1 + x^2 + x^3$. The generator polynomial for this code is given by:

$$\begin{aligned} g(x) &= (x-\alpha)(x-\alpha^2)\\ &= \alpha^3 + (\alpha+\alpha^2)x + x^2\\ &= \alpha^3 + \alpha^6 x + x^2 \ . && (2.16) \end{aligned}$$

First the message is converted into polynomial form:

$$u(x) = \alpha^2 + \alpha^3 x + \alpha^4 x^2 + \alpha x^3 + \alpha^6 x^4 \ . \tag{2.17}$$

Next, the the message polynomial is multiplied by $x^{2t} = x^2$:

$$\begin{aligned} x^{2t}u(x) &= x^2 u(x) \\ &= x^2\left(\alpha^2 + \alpha^3 x + \alpha^4 x^2 + \alpha x^3 + \alpha^6 x^4\right) \\ &= \alpha^2 x^2 + \alpha^3 x^3 + \alpha^4 x^4 + \alpha x^5 + \alpha^6 x^6 \ . \end{aligned} \tag{2.18}$$

Dividing the polynomial in (2.18) by the generator polynomial, $g(x)$, in (2.16) using polynomial division, we obtain the remainder:

$$p(x) = \alpha^5 x \ . \tag{2.19}$$

Finally, the code polynomial is formed by subtracting (2.19) from (2.18), which gives

$$c(x) = \alpha^5 x + \alpha^2 x^2 + \alpha^3 x^3 + \alpha^4 x^4 + \alpha x^5 + \alpha^6 x^6 \ , \tag{2.20}$$

which in equivalent vector form is equal to $\mathbf{c} = [000 \quad 110 \quad 001 \quad 101 \quad 111 \quad 010 \quad 011]$. This form is clearly systematic, as the original message, $\mathbf{u}$ appears in the last five 3-bit bytes, i.e., the last $mk = 15$ bits, of the codeword.

As mentioned above, there are several known efficient algorithms for the decoding of Reed-Solomon codes. Notable amongst these is the Berlekamp-Massey-Forney algorithm [22, pp. 128-136], which we will utilize in our implementation of a Reed-Solomon coded system in Chapter 4.

2.3 Turbo Codes

Turbo codes were first introduced by Berrou, Glavieux and Thitimajshima in 1993 [20]. Their work presented a simple coding scheme that utilized two parallel concatenated recursive systematic convolutional (RSC) encoders and iterative decoding to achieve a BER performance within 0.5 dB of the Shannon Bound for the AWGN channel. Turbo codes were the first practical codes to achieve performance this close to information-theoretic limits. Today, state-of-the-art turbo codes are capable of achieving BER performance arbitrarily close to the Shannon limit [10]. In this section, we discuss the theory behind these powerful codes.

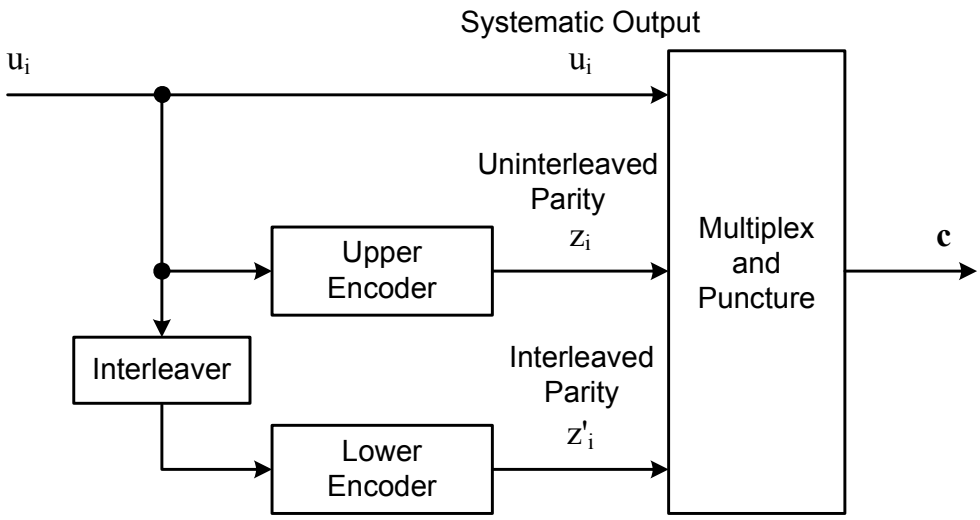


Figure 2.8: General structure of a turbo encoder.

2.3.1 Turbo Encoder

The turbo encoder consists of two parallel concatenated, systematic encoders separated by an interleaver. In general, the systematic bits of the second encoder are suppressed, as they are just a reordering of the systematic bits of the first encoder. A diagram of a generic turbo encoder is shown in Figure 2.8. The parity bits of the upper encoder are denoted z_i and those of the lower encoder are denoted z'_i. Although almost any type of encoder could be used for the constituent encoders, in practice they are almost always recursive systematic convolutional (RSC) encoders. The RSC encoders are similar in nature to the non-systematic convolutional (NSC) encoders described in Section 2.1.2. For $k = 1$, an RSC encoder can be created from an NSC encoder by feeding one of the two coded outputs back to the input of the encoder. It is this feedback structure that makes it recursive. An example of a rate $r = 1/2$ RSC encoder is given in Figure 2.9. Although the two constituent encoders can be different (asymmetric), they are typically the same (symmetric) in practice [23].

At the output of the turbo encoder, the systematic bit is multiplexed with the parity bits from the two constituent encoders. In order to increase the rate of the code, some of the parity bits may be omitted from the multiplexed codeword in a structured way. For example, if the each constituent encoder produces one parity bit, the overall rate of the code after the systematic output is added would be $r = 1/3$ and the code sequence will

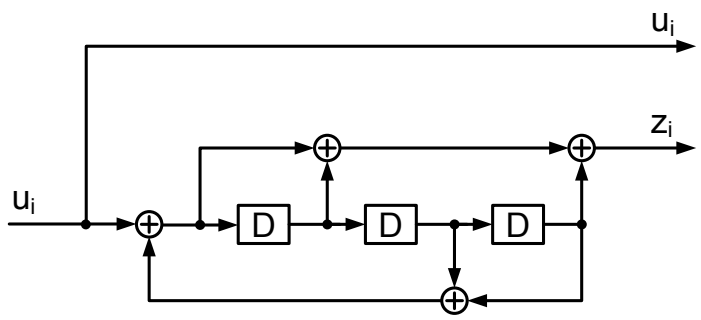


Figure 2.9: Example of a rate 1/2 recursive systematic convolutional (RSC) encoder.

be of the form $\mathbf{c} = [u_0, z_0, z_0', u_2, z_0, z_1', \cdots]$. In order to raise the code rate to $r = 1/2$, the output can drop every odd bit parity bit of the lower encoder and every even bit of the upper encoder. The codeword will then be of the form $\mathbf{c} = [u_0, z_0, u_1, z_1', u_2, z_2, \cdots]$. This process of systematically dropping coded bits to increase code rate is referred to as *puncturing*.

Although the convolutional encoders can encode data in a continuous stream, in practice it is desirable to break up encoded data into discrete chunks called *frames*. At the end of a frame, the output of the encoders is forced back to the all-zero sequence. Unlike NSC encoders, RSC encoders have an infinite impulse response due to their recursive structure. As such, they cannot in general be forced back to the all-zero state by simply inserting a string of zeros at the input. Thus, in order to end a frame and reinitialize the RSC encoders to the all-zero state, a zero-forcing sequence is needed. The feedback data provides such a sequence. Thus, an RSC encoder can be forced back to the all-zero state by connecting the feedback path directly to the input of the encoder for m time shifts.

The interleaver[1] at the input of the second constituent code is an essential component of a turbo code that has a major influence on BER performance. The interleaver rearranges a block of data in a prescribed, but irregular pattern. Because the pattern is prescribed, a corresponding deinterleaver can restore the original order of the data at the receiver. By changing the order of the data bits in a pseudo-random way, the interleaver ensures that the input to each constituent encoder is different, resulting in different parity

[1]It is important to differentiate turbo code interleavers from the rectangular interleavers used in wireless systems to combat fades. The purpose of a rectangular channel interleaver is to space out the data according to a regular pattern. The purpose of a turbo code interleaver, in contrast, is to randomize the ordering of the data in an irregular manner [23].

bits at the output of each. In addition to the property of being pseudo-random, the size of an interleaver has a significant impact on its performance. Longer interleavers result in enhanced BER performance of the turbo code, but this comes at the cost of increased delay in the system, which may not be desirable in some applications [23].

2.3.2 Turbo Decoder

Another element that is essential to the performance of turbo codes is iterative soft-input, soft-output (SISO) decoding. A diagram of a turbo decoder is shown in Figure 2.10. The overall structure of the turbo decoder is built around two processors that share information. One processor operates on information from the parity bits of the upper constituent encoder, while the second processor operates on information from the parity bits of the lower constituent encoder. During each iteration, the upper processor uses *a priori* probabilities of the data sequence to calculate *a posteriori* probabilities of the data sequence. These *a posteriori* probabilities are then used to form the input *a priori* probabilities to the lower processor which, in turn, calculates its own set of *a posteriori* probabilities. The output of the lower processor is then used to form the input to the upper processor for the next iteration. It is this exchange of information back and forth that makes the decoder iterative. Each iteration improves the decoder's ability to estimate the transmitted sequence. After each iteration, the decoder is better able to estimate the decoded sequence, although each iteration yields diminishing returns. Turbo codes derive their name from this iterative decoding structure which resembles the feedback system between the exhaust and the intake compressor in a turbo engine.

To better understand how the turbo decoder works, we now follow the flow of data in Figure 2.10. Let c_i represent a coded data bit, either systematic or parity, at the input of the channel and let y_i represent the corresponding received signal (i.e., the output of a correlator or matched filter receiver). Since c_i can only be a 1 or a 0, it is said to be a *hard value*. However, if the received signal is left unquantized or quantized to more than one bit, the received symbol, y_i, can take on a larger range of values. In this case, y_i is said to be a *soft value*. The input to the turbo decoder is a probability measure in log-likelihood ratio (LLR) form,

$$R(c_i) = \ln \left[\frac{P(y_i|c_i = 1)}{P(y_i|c_i = 0)} \right] , \tag{2.21}$$

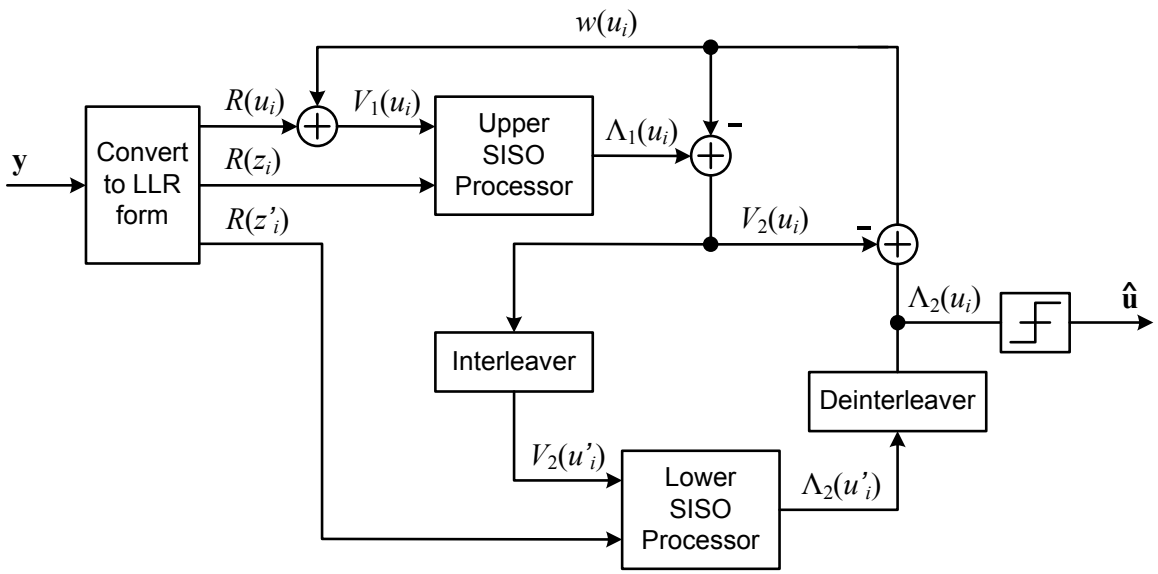


Figure 2.10: Structure of a turbo decoder. Adapted from [23].

where $P(y_i|c_i = b)$ is the conditional probability of receiving y_i given that $c_i = b$ was transmitted. Note that these are soft values. One advantage of using LLRs is that a hard decision can be made by taking the sign of the value. On the other hand, the magnitude of the value reflects the amount of certainty that the value is a 0 or 1. Additionally, calculations of Equation 2.21 require some knowledge of the channel characteristics.

At the input of the turbo decoder, LLRs are calculated separately for the systematic and parity bits, denoted $R(u_i)$, $R(z_i)$, and $R(z_i')$. The upper processor takes $V_1(u_i)$ and $R(z_i)$ as its systematic and parity inputs, respectively. For the first iteration, the systematic input, $V_1(u_i)$, is simply equal to $R(u_i)$. The upper processor then computes *a posteriori* LLR value for each data bit of the form

$$\Lambda_1(u_i) = \ln\left[\frac{P(u_i = 1|\mathbf{y})}{P(u_i = 0|\mathbf{y})}\right] \tag{2.22}$$

where $\mathbf{y} = [y_1, y_2, \cdots, y_n]$ is the received sequence corresponding to the entire encoded frame. For the first iteration, this value $\Lambda_1(u_i)$ is simply interleaved to form the systematic input to the second encoder, denoted $V_2(u_i')$. The second processor then uses $V_2(u_i')$ along with the parity input from the lower constituent encoder, $R(z_i')$, to compute a similar LLR estimate, $\Lambda_2(u_i')$. This value is then deinterleaved and fed back into the first processor for the next iteration. In this way, the turbo decoder is essentially a feedback system. As such,

it is important to avoid positive feedback, which would result in an unstable system. To ensure that only information which is unique to a given decoder iteration is fed back, the input of each processor is subtracted from its output prior to feeding information to the other processor. This difference value is called the *extrinsic information* and is denoted by $w(u_i)$ [23, pp. 387-388]. In subsequent iterations the systematic inputs to the upper and lower encoders are formed as

$$V_1(u_i) = R(u_i) - w(u_i), \tag{2.23}$$

$$V_2(u_i) = \Lambda_1(u_i) - w(u_i). \tag{2.24}$$

Once the desired number of iterations have been completed, a threshold is used to obtain a hard decision on the decoded message, $\hat{\mathbf{u}}$

Both of the processors within the decoder take soft LLR values as inputs and produce similar values at their output. For this reason they are referred to as *soft-input, soft-output* (SISO) processors. The traditional Viterbi algorithm, commonly used to decode convolutional codes, cannot be used in these processors because it generates a hard output. Instead, a SISO algorithm must be used. Two commonly used algorithms are the soft-output Viterbi algorithm (SOVA) [24] and the maximum *a posteriori* (MAP) algorithm (also known as the Bahl-Cocke-Jelinek-Ravive (BCJR) algorithm) [25]. In general, the BCJR algorithm outperforms the SOVA algorithm in iterative decoders, though it is more complex. In order to reduce complexity, the MAP algorithm can be implemented in the log domain, which reduces complexity by changing multiplications to additions. This algorithm is called the log-MAP algorithm [26]. One downside to this approach is that the addition of two variables, x and y, is transformed into the operation

$$\ln(e^x + e^y) \tag{2.25}$$

which is difficult to compute. Simplifications to this algorithm were developed by Viterbi in [26]. Consider the function *max-star* defined as:

$$\max^*(x, y) = \ln(e^x + e^y) \tag{2.26}$$

$$= \max(x, y) + \ln(1 + e^{-|y-x|}) \tag{2.27}$$

$$= \max(x, y) + f_c(|y - x|) \tag{2.28}$$

where $f_c(x) = \ln(1 + e^{-x})$. Rather than calculating the exact value of f_c, an approximation can be used to simplify computation . First, f_c can be approximated as a straight line.

This algorithm is called the *linear-log-MAP* algorithm. Second, f_c can be approximated as a step function (i.e., f_c takes on one of two possible values). This algorithm is called the *constant-log-MAP* algorithm. Third, the simplest approximation is to set $f_c = 0$, such that $max^*(x, y) = max(x, y)$. This algorithm is called the *max-log-MAP* algorithm, and is the least computationally intensive simplification.

2.4 Low-Density Parity-Check Codes

Low-density parity-check (LDPC) codes were originally presented by Gallager in 1962 [28, 29]. Despite their excellent performance, LDPC codes were not considered by the coding community. This was primarily due to the fact that the iterative decoding method proposed by Gallager was too computationally intensive for the technology available at that time. In 1996, after the discovery of turbo codes, LDPC codes were rediscovered by MacKay and Neal [30]. Today LDPC codes are recognized as one of the most powerful state-of-the-art coding schemes known. They are Shannon limit approaching codes capable of achieving performance as good as – and, in some cases, better than – turbo codes.

2.4.1 Structure and Classification of LDPC Codes

LDPC codes are a class of linear block codes, so-called because their parity-check matrix, H, has a low density of 1's relative to 0's. As with all block codes, LDPC codes are uniquely defined by their parity-check matrix. In Gallager's original construction, the H matrix was defined in non-systematic form such that each column had a small number, $j \geq 3$, of 1's and each row had another small number, $k > j$, of 1s. The H matrix was then constructed at random subject to these constraints [28]. Today, LDPC codes constructed in this manner are referred to as *regular* LDPC codes. LDPC codes can also be constructed such that the number of 1's in each row and column of H vary. These codes are referred to as *irregular* LDPC codes. In general, the BER performance of irregular LDPC codes is better than that of regular LDPC codes [22, pp. 280-281]. Since the H matrix is sparse, the corresponding G matrix tends to be dense. This combined with the large block sizes of practical LDPC codes makes encoding computationally intensive. Specifically, if calculated directly, the time to create the generator matrix scales as n^3 and the time to encode scales as n^2, where n is the block length [31, p. 562]. Standardized LDPC codes utilize block

lengths on the order of 10^4 bits. For these codes, certain structures must necessarily be imposed on the H matrix to both facilitate the description of codes and reduce encoding complexity. These structures will be discussed in a standard-specific context in Chapter 6.

2.4.2 Decoding LDPC Codes: The Sum-Product Algorithm

The decoding of LDPC codes is done using an iterative algorithm that uses soft-bit information similar to that of the turbo decoder. The best known algorithm for decoding LDPC codes is the *sum-product algorithm*, also known as *iterative probabilistic decoding* or *belief propagation* [32]. In order to demonstrate this algorithm, it is helpful to first define a graphical representation for linear block codes called a *Tanner graph* [33]. An example parity-check matrix, its parity-check equations, and its corresponding Tanner graph are shown in Figure 2.11. The Tanner graph consists of two types of nodes, *bit nodes* and *check nodes*, which are connected by *edges*. The bit nodes represent the n bits of the codeword. The check nodes represent the $n-k$ parity-check equations, $\mathbf{h}_i$. A bit node is connected to a check node by an edge if and only if the corresponding code bit is contained in the corresponding parity-check equation. Since edges between any two nodes of the same type are not allowed, the Tanner graph is said to be bipartite. If there are d edges emanating from a node, the node is said to be of degree d. Let us also define a *cycle* as a series of edges, each traversed only once, that start and end at the same bit node. In decoding via the sum-product algorithm, it is desirable to avoid short cycles of length 4 and, to a lesser extent, length 6. This is because short cycles limit decoding performance and can prevent the algorithm from converging to the maximum likelihood decision value [15, p. 858].

With an understanding of Tanner graphs, it is now possible to summarize the iterative decoding of LDPC codes based on the sum-product algorithm. The objective is to find the most likely received codeword, $\hat{\mathbf{c}}$, such that $\hat{\mathbf{c}}H^T = \mathbf{0}$. To do so, we will employ two values, Q_{ij}^x and R_{ij}^x, where:

- Q_{ij}^x is the probability that the j-th bit of $\hat{\mathbf{c}}$ has value x, where $x \in 0, 1$, given information obtained from all parity-check nodes connected to c_j other than $\mathbf{h}_i$. This value is passed from bit node to c_j to check node $\mathbf{h}_i$.

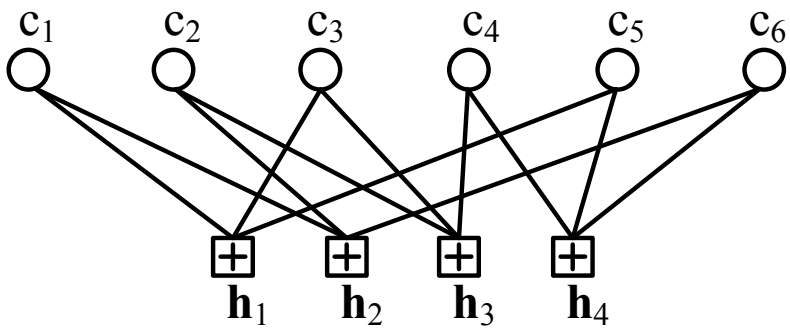
- R_{ij}^x is the probability that the i-th parity-check equation, $\mathbf{h}_i$, is satisfied if c_j is fixed at value x, i.e. R_{ij}^0 assumes that $c_j = 0$, and the other bits of $\hat{\mathbf{c}}$ can be either 0 or 1,

$$
\begin{array}{cc}
 & \begin{array}{cccccc} c_1 & c_2 & c_3 & c_4 & c_5 & c_6 \\ \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \end{array} \\
H = & \begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix} \begin{array}{l} \leftarrow \mathbf{h}_1 \\ \leftarrow \mathbf{h}_2 \\ \leftarrow \mathbf{h}_3 \\ \leftarrow \mathbf{h}_4 \end{array}
\end{array}
$$

(a)

$$
\begin{array}{ll}
\mathbf{h}_1: & c_1 \oplus c_3 \oplus c_5 = 0 \\
\mathbf{h}_2: & c_1 \oplus c_2 \oplus c_6 = 0 \\
\mathbf{h}_3: & c_2 \oplus c_3 \oplus c_4 = 0 \\
\mathbf{h}_4: & c_4 \oplus c_5 \oplus c_6 = 0
\end{array}
$$

(b)



(c)

Figure 2.11: The Tanner graph is a way of representing a parity-check matrix graphically. Depicted are (a) an example parity-check matrix, (b) its parity-check equations, and (c) the corresponding Tanner graph.

given by the probabilities Q_{ij}^x. This value is passed from check node $\mathbf{h}_i$ to bit node c_j.

The algorithm begins with an initialization to determine the initial values of Q_{ij}^x. In this step, Q_{ij}^x are set to the estimates of the received symbols, denoted f_j^x, the probability that the j-th symbol is x given the received symbol y_j. For the AWGN channel with on-off keying (OOK), these values are:

$$f_j^0 = \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{-y_j^2}{2\sigma^2}} \tag{2.29}$$

$$f_j^1 = \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{-(y_j-1)^2}{2\sigma^2}}. \tag{2.30}$$

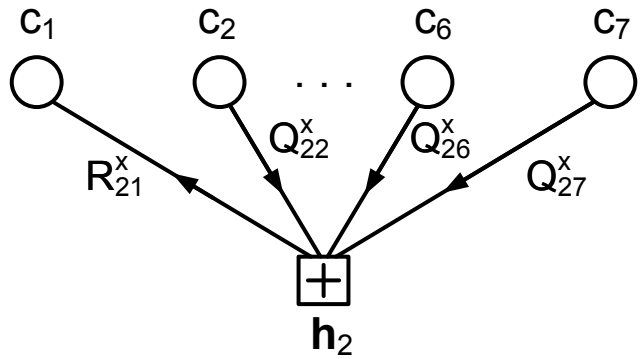


Figure 2.12: Horizontal step of the sum-product algorithm.

Horizontal Step

The next step is referred to as the *horizontal step*, because it operates across each row of the H matrix. In this step, each check node, $\mathbf{h}_i$, is inspected and, for each edge connected to $\mathbf{h}_i$, two values are computed: R^0_{ij} and R^1_{ij}. The probability that $\mathbf{h}_i$ is satisfied given $c_j = x$ is given by

$$P(\mathbf{h}_i|c_j = x) = \sum_{\hat{\mathbf{c}}:c_j=x} P(\mathbf{h}_i|\hat{\mathbf{c}})P(\hat{\mathbf{c}}|c_j = x) \tag{2.31}$$

where $P(\mathbf{h}_i|\hat{\mathbf{c}})$ is equal to 1 if the parity-check equation $\mathbf{h}_i$ is satisfied or 0 if it is not satisfied. The values of R^x_{ij} are now calculated as

$$R^x_{ij} = \sum_{\hat{\mathbf{c}}:c_j=x} P(\mathbf{h}_i|\hat{\mathbf{c}}) \prod_{k\in N(i)|j} Q^{c_k}_{ik} \tag{2.32}$$

where $N(i)|j$ represents all the indexes of the bit nodes connected to check node $\mathbf{h}_i$ with the exclusion of c_j. As an example, consider an LDPC code with parity-check equation $\mathbf{h}_2$ defined as $c_1 \oplus c_2 \oplus c_6 \oplus c_7 = 0$. The corresponding values of R^0_{21} and R^1_{21} passed from check node $\mathbf{h}_2$ to bit node c_1 are thus given by:

$$R^0_{21} = Q^0_{22}Q^0_{26}Q^0_{27} + Q^1_{22}Q^1_{26}Q^0_{27} + Q^1_{22}Q^0_{26}Q^1_{27} + Q^0_{22}Q^1_{26}Q^1_{27} \tag{2.33}$$

$$R^1_{21} = Q^1_{22}Q^0_{26}Q^0_{27} + Q^0_{22}Q^1_{26}Q^0_{27} + Q^0_{22}Q^0_{26}Q^1_{27} + Q^1_{22}Q^1_{26}Q^1_{27} \tag{2.34}$$

This process of updating an edge as part of the horizontal step is illustrated in Figure 2.12.

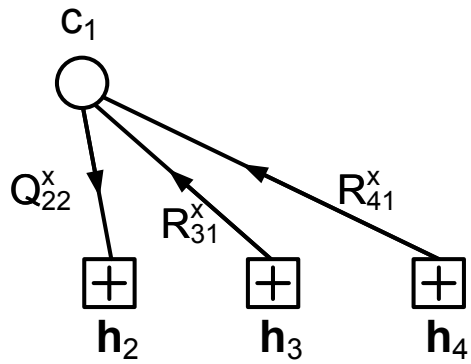


Figure 2.13: Vertical step of the sum-product algorithm.

Vertical Step

The next step is referred to as the *vertical step*, because it operates across each column of the H matrix. In this step, each bit node receives the values of R_{ij}^x from its adjacent check nodes and then updates the values of Q_{ij}^0 and Q_{ij}^1. This is done using the equation

$$Q_{ij}^x = \alpha_{ij} f_j^x \prod_{k \in M(j)|i} R_{kj}^x \tag{2.35}$$

where $M(j)|i$ represents the indexes of all check nodes connected to bit node c_j with the exclusion of $\mathbf{h}_i$. The variable α_{ij} is a normalization factor chosen such that $Q_{ij}^0 + Q_{ij}^1 = 1$. As an example, consider an LDPC code in which the code bit, c_1, participates in 3 parity-check equations, denoted $\mathbf{h}_2$, $\mathbf{h}_3$, and $\mathbf{h}_4$. In this case, the updated values of Q_{21}^0 and Q_{21}^1 to be passed from bit node c_1 check node $\mathbf{h}_2$ in the next iteration are given by

$$Q_{21}^0 = \alpha_{21} f_1^0 R_{31}^0 R_{41}^0 \tag{2.36}$$

$$Q_{21}^1 = \alpha_{21} f_1^1 R_{31}^1 R_{41}^1 \tag{2.37}$$

and the correction factor is

$$\alpha_{21} = \frac{1}{f_1^0 R_{31}^0 R_{41}^0 + f_1^1 R_{31}^1 R_{41}^1}. \tag{2.38}$$

This process of updating an edge as part of the vertical step is illustrated in Figure 2.13.

After each horizontal step, an estimate of $\hat{\mathbf{c}}$ can be calculated for that iteration. Each bit estimate, $\hat{c}_j$, is chosen based on the larger *a posteriori* probability measure, i.e.

$$\hat{c}_j = \arg\max_x f_j^x \prod_{k \in M(j)} R_{kj}^x \ , \tag{2.39}$$

where $M(j)$ are the indexes of all the parity nodes connected to bit node c_j. At this point the algorithm computes the value $\hat{\mathbf{c}}H^T$ and does one of three things:

1. If the algorithm has converged upon a codeword, i.e. $\hat{\mathbf{c}}H^T = \mathbf{0}$, then the algorithm stops and decodes $\hat{\mathbf{u}}$ from $\hat{\mathbf{c}}$.

2. If the algorithm has not converged upon a codeword, i.e. $\hat{\mathbf{c}}H^T \neq \mathbf{0}$, and the maximum number of iterations have not been met, then the algorithm proceeds to the vertical step for another iteration.

3. If the algorithm has not converged upon a codeword, and the maximum number of iterations have been met, then a decoding failure has occurred. In this case the algorithm returns the error codeword $\hat{\mathbf{c}}$ and flags it as a decoding error.

For a detailed example of the sum-product algorithm for decoding LDPC codes, the reader is directed to [22, pp. 284-297].

Chapter 3

Channel Characterization and Estimation

In this chapter we will explore light propagation and noise statistics in an underwater free-space optical (FSO) communication channel testbed in order to establish a channel model for theoretical studies. Since signal-to-noise ratio (SNR) estimation is important in practical analysis, an SNR estimator is established as well. This chapter is structured as follows. In Section 3.1 a brief discussion of the propagation of light through water and Beer's Law is presented. In Section 3.2, the underwater free-space optical communication testbed and experimental procedure are described. Then, in Section 3.3, experimental results obtained on the experimental testbed are used to characterize the channel noise and a mathematical model for the underwater FSO channel is developed. A maximum-likelihood SNR estimator is established in Section 3.4. Finally, in Section 3.5, Beer's Law is used to establish a link between coding gains in electrical SNR and range extension of a FEC-coded system.

3.1 Propagation of Light through Water

Underwater wireless communication is limited by the fact that much of the electromagnetic spectrum is highly attenuated in seawater. A diagram of electromagnetic attenuation in seawater as a function of frequency is shown in Figure 3.1. Radio frequency (RF) waves and microwaves (∼30 kHz - 300 GHz), so ubiquitous in terrestrial wireless sys-

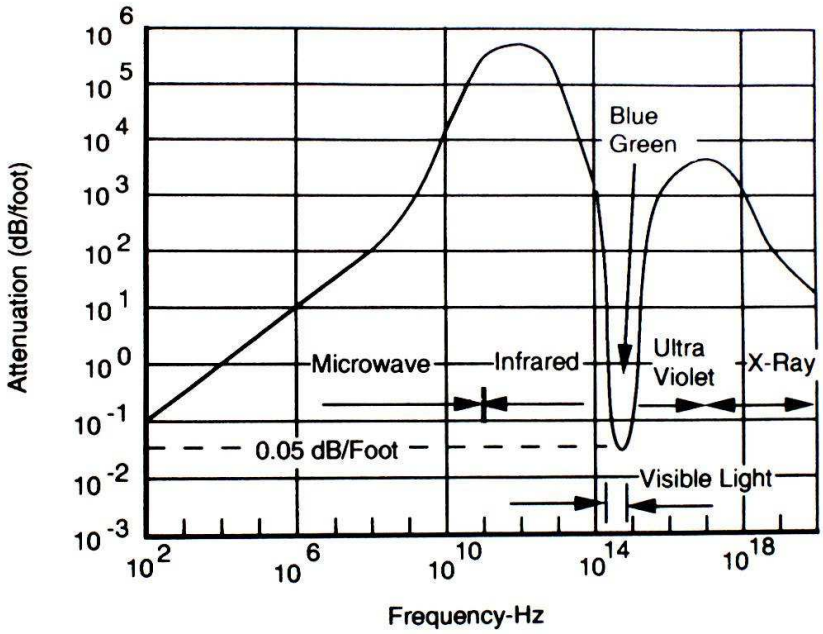


Figure 3.1: Attenuation of electromagnetic radiation in seawater. Reproduced from [34].

tems, suffer a large attenuation allowing them to propagate only a few feet underwater. At infrared frequencies (∼300 GHz - 400 THz), which are typically used in fiber optic systems, attenuation is even greater. There is however a "window" of transmission through seawater in approximately the blue-green region of the optical spectrum. This corresponds to a frequency of around 625 THz or a wavelength of around 480 nm. The low attenuation in this region makes underwater free-space optical communication in the blue-green spectrum a good option for wireless communication underwater.

Since the area of underwater free-space optical (FSO) communication is relatively new, there is little information available in the literature concerning the use of seawater as an optical communication medium. However, the properties of light propagating through water have been studied extensively in a wide range of other disciplines such as underwater imaging, oceanography, physics, and the marine sciences. There also exists a substantial body of research conducted above the water on terrestrial FSO communication. In this section, we will draw upon this body of research to better understand the underwater FSO channel. A few of the most relevant properties, including the absorption and scattering of light, are discussed.

The optical properties of seawater can be divided into two categories: *inherent optical properties* and *apparent optical properties* [35, 36]. The inherent optical properties of water are those which are a function only of the material properties of the medium itself, independent of the properties of the light source. The apparent optical properties, on the other hand, are those which depend on both the geometrical structure of the light field and the inherent optical properties of the medium. Inherent optical properties include, most notably, absorption, scattering, and attenuation, while apparent optical properties include radiometric quantities such as irradiance reflectance, radiance reflectance, and attenuation coefficients for upwelling and downwelling irradiance [36].

To understand the underwater FSO communication channel, the inherent optical properties of spectral absorption and spectral scattering are the most significant, since they are the main source of signal attenuation [37]. Seawater is a complex mixture of dissolved substances, organic matter, living organisms, and water molecules themselves, all of which contribute to its optical properties. Table 3.1 summarizes the contribution of the main components of seawater to spectral absorption and spectral scattering and their corresponding dependence on wavelength (denoted by λ).

3.1.1 Absorption

Absorption is the process in which light energy in the form of photons is lost from the light field and is either converted into another form of energy (such as heat or chemical energy produced during photosynthesis), or else is lost in an electronic transition (such as fluorescence) [35]. The spectral absorption of seawater is due to both inorganic and organic contributors. The main inorganic contributors are water molecules and sea salt. The absorption of pure seawater, in which no organic matter is present, can be considered as the sum of the absorption due to pure water and the absorption due to sea salts present in pure seawater. In the visible spectrum, absorption due to sea salts is negligible [38]. The absorption of pure seawater as a function of wavelength in the visible spectrum is shown in Figure 3.2. This data has been compiled from various studies. The various data sets agree that there is a window in the blue-green region of the spectrum, around $\lambda \approx 480$ nm, where pure seawater is least absorptive.

Table 3.1: Summary of absorption and scattering characteristics of seawater. Adapted from [38].

	Absorption		**Scattering**	
	Characteristic	**λ dependence**	**Characteristic**	**λ dependence**
Water	Invariant at constant temp. and pressure	strong	Invariant, small compared to absorption	λ^{-4}
Sea salts (inorganic)	Negligible in visible spectrum	Some increase towards short λ	Appreciable	None
Colored dissolved organic matter (yellow substance)	Variable	Increase towards short λ	None	None
Particulate matter (including phyto-plankton and detritus)	Variable	Increase towards short λ	Variable	Variable

The main organic contributors to spectral absorption are colored dissolved organic matter (CDOM) and organic particulate matter, including detritus and phytoplankton [36, 38]. CDOM, also referred to as *yellow substance* or *Gelbstoff*, is a rather loosely defined combination of chemical compounds resulting from the decomposition of marine organisms. According to Shifrin [39], about 150 different organic substances have been identified and there are likely more that have not been identified. Absorption by CDOM decreases with wavelength and usually dominates seawater absorption for $\lambda < 400$ nm [36]. Another major organic contributor to spectral absorption is phytoplankton, microscopic plants that exist in suspension in many types of seawater. Absorption by phytoplankton is primarily due to pigments within phytoplankton cells. These pigments, generally referred to as *chlorophyll*, include chlorophyll *a* and accessory pigments such as pheophytin *a*. The absorption spectrum of phytoplankton cells includes a strong absorption band in the blue ($\lambda = 440$ nm) attributed to accessory pigments, and a weaker absorption band in the red ($\lambda = 675$ nm) attributed to chlorophyll *a* [36]. Absorption due to phytoplankton varies based on concentration, with higher concentrations near the surface. The third major organic contributor to spectral absorption is detritus, organic waste formed by dead plants and animals and fecal material. Having similar origins to CDOM, detritus displays a similar absorption spectrum to CDOM with absorption decreasing with wavelength at the lower boundary of the blue-green window [36].

The absorption of seawater is given by the spectral absorption coefficient, $a(\lambda)$. This value is calculated as the sum of the above factors, i.e.

$$a(\lambda) = a_w(\lambda) + a_{CDOM}(\lambda) + a_{phy}(\lambda) + a_{det}(\lambda) \tag{3.1}$$

where $a_w(\lambda)$ is absorption due to pure seawater, $a_{CDOM}(\lambda)$ is absorption due to CDOM, $a_{phy}(\lambda)$ is absorption due to phytoplankton, and $a_{det}(\lambda)$ is absorption due to detritus (cf. [36, p. 143]). Absorption at the upper boundary of the blue-green window, in terms of wavelength, is primarily due to the absorption of pure seawater, which varies with changes in temperature and pressure. Absorption at the lower boundary, however, is primarily caused by CDOM and particulate matter. Thus, as turbidity increases, the lower boundary of the blue-green window is shifted from the blue to the green or even green-yellow portion of the spectrum.

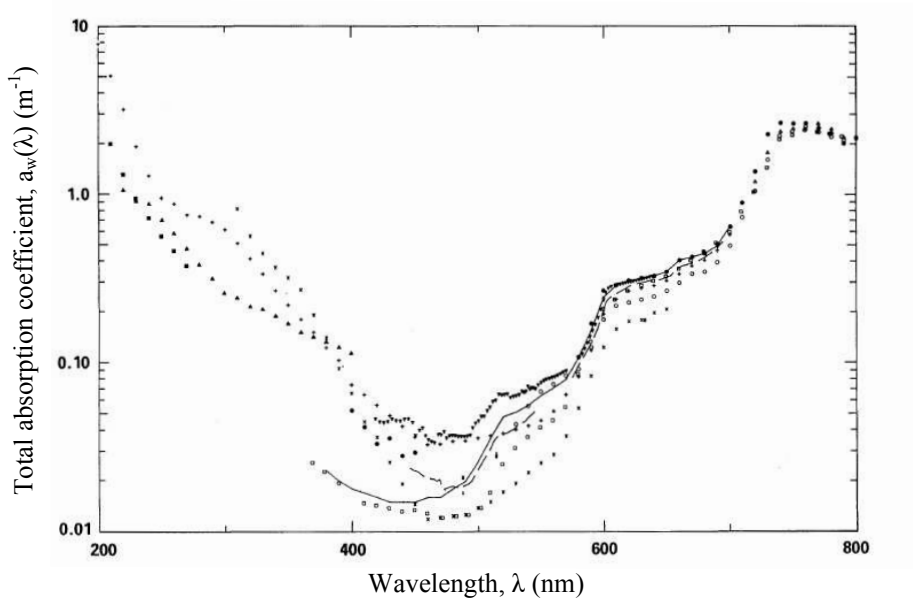


Figure 3.2: Absorption of pure seawater as a function of wavelength as given by multiple studies. Adapted from [40].

3.1.2 Scattering

Scattering is the process in which light energy in the form of a photon is redirected spatially from the forward propagating path due to interaction with molecules or particles. Unlike absorption, the photon is not converted to another form of energy in a scattering event [35]. Scattering can be divided into two types: Rayleigh scattering, which occurs for small particles with radius $r << \lambda$, and Mie scattering, which occurs for larger particles with radius $r \approx \lambda$ and larger. A fundamental difference between these two types of scattering is that Rayleigh scattering tends to scatter uniformly in all directions while Mie scattering is biased in the forward direction [41].

As indicated in Table 3.1, the scattering of light is primarily caused by water molecules, sea salt, and particulate matter such as phytoplankton and detritus. The scattering of pure seawater can be considered the sum of scattering due to water molecules and sea salts. This scattering is small as compared to the absorption of pure seawater. Since both water molecules and salt ions are small relative to optical wavelengths, Rayleigh scattering occurs. The spectral scattering due to particulate matter like phytoplankton and detritus can vary widely depending on the composition of the water. Since these particles

range in size, both Raleigh scattering and Mie scattering may result [36].

The scattering of seawater is given by the spectral scattering coefficient, $b(\lambda)$. This value is calculated similarly to the spectral absorption coefficient:

$$b(\lambda) = b_w(\lambda) + b_{phy}(\lambda) + b_{det}(\lambda) \tag{3.2}$$

where $b_w(\lambda)$ is scattering due to pure seawater, $b_{phy}(\lambda)$ is scattering due to phytoplankton, and $b_{det}(\lambda)$ is scattering due to detritus (cf. [36, p. 143]).

3.1.3 Attenuation

Attenuation of an optical signal transmitted through water is primarily caused by absorption and scattering, as described above. The attenuation of seawater is quantified by the spectral attenuation coefficient, $c(\lambda)$. This value is defined as as the sum of the spectral absorption and spectral scattering coefficients, i.e.

$$c(\lambda) = a(\lambda) + b(\lambda) \ . \tag{3.3}$$

In practice, $c(\lambda)$ is typically measured with a transmissometer consisting of a calibrated, stable light source and a narrow field-of-view detector separated by a small distance. The attenuation of an optical signal as a function of $c(\lambda)$ and distance, d, is described by Beer's Law:

$$I = I_0 e^{-c(\lambda)d} \tag{3.4}$$

where I is the detected intensity at the receiver, I_0 is the transmitted intensity, $c(\lambda)$ is the spectral attenuation coefficient as a function of wavelength, and d is optical path length in meters [41].

In practice, there are other aspects of light propagation through water that are not captured by Beer's Law. For example, Beer's law assumes that once a photon is scattered, it is lost from the received intensity. In very turbid waters, multiple scattering events may cause a scattered photon to resume propagation in the forward direction. This type of multiple scattering can result in spatial and temporal dispersion and is explored in [42] and [43]. The field-of-view of the receiver is another consideration. For example, a larger field of view could be used to capture more photons when spatial dispersion occurs. In general, these considerations become more significant in turbid water where dispersion is

more prevalent, while in pure seawater Beer's Law is more directly applicable. In this work, Beer's Law is assumed to hold for simplicity. However, it should be noted that it is unclear whether results measured in the lab can be directly extrapolated to longer distances and a wider range of water conditions.

3.2 Experimental Setup and Procedure

3.2.1 System Architecture

The experimental results in this thesis were obtained using an underwater testbed developed by Cox and Simpson in [12] and [13]. The system consists of a laser-diode-based transmitter and a photodiode based receiver operating through a custom-built 3,800 liter indoor water tank. The purpose of the underwater testbed is to simulate a practical underwater communication environment by emulating a range of controlled underwater scenarios. An overview of the system architecture is included for completeness but is not a novel contribution of this thesis. A more detailed description of the system architecture can be found in [44].

The laser-diode-based transmitter employs intensity modulation (IM) using on-off keying (OOK) and return-to-zero (RZ) line coding with a 50% duty cycle to transmit at a baud rate of 500 kbps. The light source is a 405 nm (blue) indium-gallium-nitride laser-diode operating at a laser power of 100 mW. This setup offers moderately high optical power and falls within the blue-green window discussed in Section 3.1. No optical attenuator is used at the transmitter. Rather, the attenuation of the transmitted signal is entirely due to the channel medium itself. The data packets to be transmitted originate from the transmitter personal computer (PC). There the data are generated, encoded, and packetized using MATLAB. The details of the packet structure are discussed below. The data packet is then streamed via USB to an FPGA board which buffers the data and sends them to the laser-diode driver.

The photodiode-based receiver is a direct-detection (DD) system consisting of a silicon PIN photodiode, transimpedance amplifier, variable gain amplifier, and an analog-to-digital converter. Light is collected with a 2-inch converging lens with a field-of-view of 40 degrees. The output of the photodiode is pre-amplified by a transimpedence amplifier. The output of the preamp is then amplified by a variable-gain amplifier and digitized by an

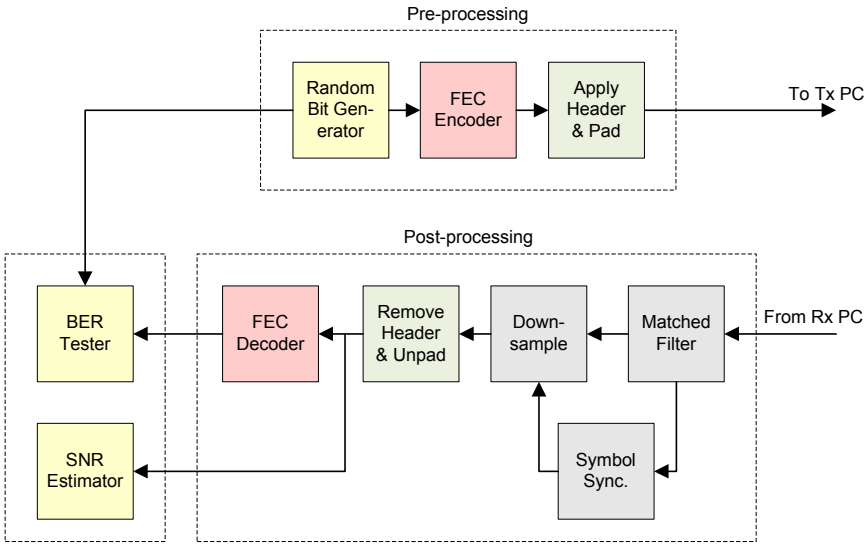


Figure 3.3: MATLAB processing architecture.

8-bit analog-to-digital converter (ADC). The ADC samples the signal at a rate of 10 MSps, resulting in 20 times oversampling of the OOK modulated signal. An FPGA board buffers the data until the entire packet is received and then transmits the received data to the PC via USB for post-processing in MATLAB.

The underwater FSO channel is emulated by means of an indoor water tank constructed out of wood and fiberglass. The tank is 3.66 m long, 1.2 m wide, and 1.2 m tall and is capable of holding 3,800 liters of water. Polycarbonate viewing windows are installed on the long ends of the tank for the transmitter and receiver. The transmitter window is circular, 20 cm in diameter, and the receiver window is rectangular, 53 cm high and 38 cm wide. Different water conditions are simulated by varying the concentration of a scattering agent (Maalox) consisting of suspended particles.

Pre- and post-processing of experimental data is executed in MATLAB, as shown in Figure 3.3. Prior to experimentation, source data is generated, encoded, and packetized using MATLAB and saved to the transmitter PC. The data packet structure used in the experiments is illustrated in Figure 3.4. The first 1024 bits of the packet comprise the header. The header contains a 512-bit synchronization sequence of all 1s, which aids the synchronizer in acquiring the symbol-timing phase, and a 512-bit flag sequence (known to the receiver) that marks the beginning of the data. Due to limitations in buffer size, the

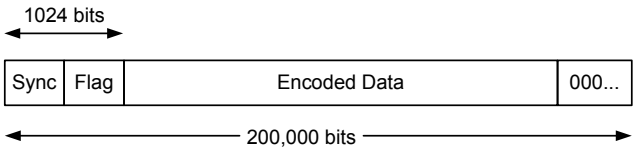


Figure 3.4: Transmitted packet structure.

data packet is limited in size to $200,000$ bits, including header. To maximize the number of information bits sent per packet, the largest multiple of the coded frame size that is below this limit is used. Since frame size depends on the specific FEC code, the number of coded bits per packet varies. For experimental results presented in this thesis, the number coded bits per packet range from $160,632$ to $198,648$ bits. Finally, zeros are padded to the end of the packet to make it a uniform $200,000$ bits.

At the receiver, demodulation, synchronization, detection, and decoding of the collected data is implemented digitally in MATLAB. The received signal is first sent to a matched filter demodulator (cf., [51, pp. 238]), which consists of a finite-impulse response (FIR) digital filter matched to the RZ rectangular pulse shape followed by a down-sampler. A symbol synchronizer is used to determine the optimal sampling times. During our experiments, we observed a deterministic drift in the sampling times. Based on observation of the entire data set post-experiment, it was possible to estimate and compensate for this drift, thus making a conventional synchronizer unnecessary. However, in a practical system, symbol-by-symbol synchronization would need to be implemented using an algorithm such as an early-late gate (cf., [51, pp. 363]) or a maximum-likelihood timing algorithm (cf., [51, pp. 359]).

Next the packet header and zero padding are removed from the demodulator output. The coded data are then fed into the FEC decoder. For RS coded data, hard decision detection of the data stream is performed immediately, resulting in an estimate of 1 or 0 for each received pulse. This data stream is then passed to the RS decoder. For turbo and LDPC coded data, the soft received values are used to form log-likelihood ratios (LLRs), which are then sent to the appropriate decoder. Finally, with knowledge of the original transmitted data, the decoded data, and an estimate of the received signal-to-noise ratio (SNR), bit-error-rate (BER) computations are performed. The algorithm used to compute the SNR is discussed in Section 3.4.

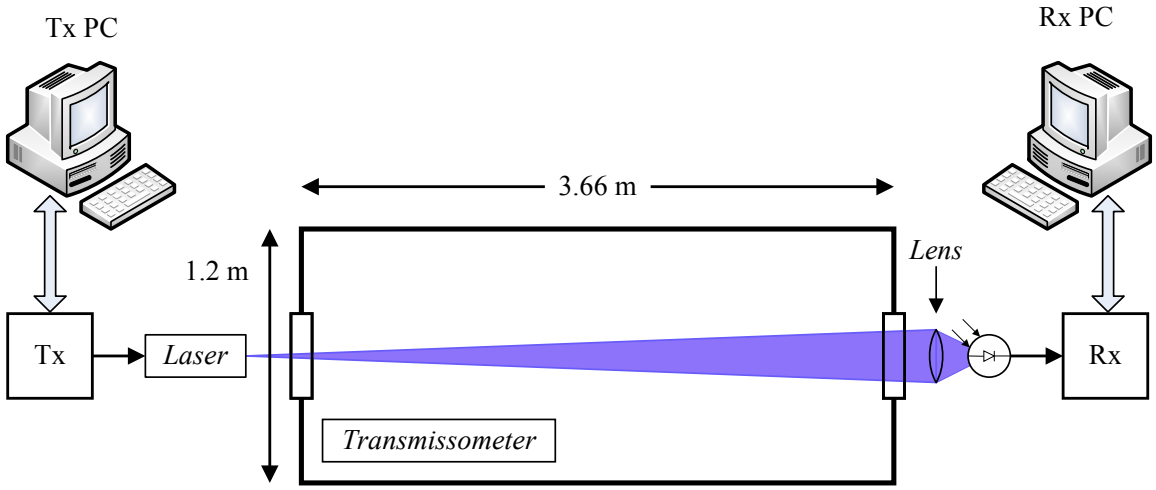


Figure 3.5: Experimental Setup.

3.2.2 Experimental procedure

The experimental setup is shown in Figure 3.5. Prior to experimentation, the tank is filled with approximately 3,600 liters of municipal water that is filtered through a 10 μm cartridge filter to remove large particles. To further filter out impurities, an 18 inch diatomaceous earth pool filter is then used to filter out particles larger than 3 μm. The transmitter and receiver are set up at either end of the tank outside the Polycarbonate windows as shown in Figure 3.5. The transmitter-receiver pair are aligned to achieve maximum signal strength in clear water. Once the setup is complete, the transmitter and receiver software are configured to automatically transmit and receive a packet of data approximately once every thirty seconds throughout the duration of the experiment.

Liquid Maalox, a pharmaceutical antacid, is added to the water as a scattering agent. Pharmaceutical antacids have long been used in laboratory environments to simulate the scattering particles found in natural waters [45], [46]. The use of Maalox in this capacity has been thoroughly measured and validated by [47] as having scattering characteristics similar to that of natural waters. To gradually increase water turbidity over the course of the experiment, a standard medical syringe pump is used to dispense the Maalox at a controlled rate of 50 mL/hr. The output of the syringe pump is fed via a small tube directly to the intake of a sump pump at the bottom of the tank. The sump pump runs continuously to mix the Maalox, thereby making the distribution of Maalox within the

tank relatively homogeneous. To minimize the effects of turbulence on the optical link, the sump pump is placed away from the beam path. A WETLabs C-Star transmissometer with a 25 cm path length operating at 530 nm is used to measure the attenuation coefficient, c, corresponding to each experimental data packet transmission [48]. We note that this wavelength is different from the operating wavelength of the system, which is 405 nm. The impact of this difference will be discussed later in this chapter. After each experiment, the tank is drained and any accumulated Maalox is cleaned off the viewing windows to maintain controlled experimental conditions between experiments.

Although all MATLAB processing and data analysis contained in this thesis is the work of the author, it should be noted that raw experimental data from the underwater testbed was not collected by the author. Credit is given to Jim Simpson and William Cox, members of Dr. John Muth's optics lab, for conducting all underwater experiments used in this thesis.

3.3 Channel Characterization

In order to establish a theoretical model for the underwater FSO communication channel, we first investigate the statistical properties of the channel noise. After an experiment is conducted using the experimental testbed and procedure described in Section 3.2, a received data packet corresponding to a high concentration of the scattering agent is analyzed. The data presented was taken after >50 mL of Maalox had been dispensed into the tank, resulting in an estimated SNR of <-20 dB[1] and a measured attenuation coefficient value of approximately 2.0 m^{-1}. We conclude that, at high concentration of the scattering agent, the noise component is much larger than the signal component and the signal component is thus negligible. By using the approximation that the received data is made up only of noise, we can investigate the statistical properties of the channel noise using this data. In order to avoid altering the statistical properties of the signal, the unprocessed received data prior to matched filtering is used.

The empirical cumulative distribution function (CDF) of the channel noise was plotted using MATLAB. For the purpose of comparison, the empirical noise CDF has been normalized to mean zero and variance one. The result is plotted in Figure 3.6 along with

[1]The algorithm used for SNR estimation is described in Section 3.4.

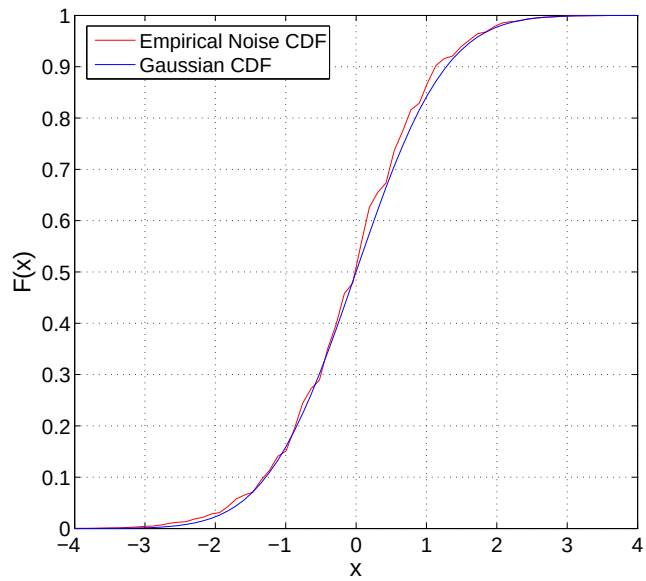


Figure 3.6: Empirical noise CDF of the channel (normalized to mean 0, variance 1) plotted against the $N(0,1)$ Gaussian CDF.

the $\mathcal{N}(0,1)$ Gaussian CDF for reference. Some deviation is to be expected; however the empirical CDF closely matches the Gaussian CDF. Since SNR calculations are largely influenced by the tail probabilities of the noise CDF, these are investigated here in further detail. Figure 3.7 shows the upper and lower empirical tail probabilities plotted as a function of the Gaussian tail probabilities. For the upper tail probability, $(1 - F(x))$ is plotted as a function of the q-function, $Q(x)$, (cf.,[51, p. 40]). For the lower tail probability, $F(x)$ is plotted as a function of $(1 - Q(x))$. The plot uses logarithmic axes to better show detail. For a perfect Gaussian distribution, the resulting plot would be a straight line of slope 1. There is some deviation from the Gaussian distribution, especially in the lower tail probability. However, both tails of the empirical CDF are relatively close to the Gaussian tail. Figures 3.6 and 3.7 suggest that the noise is well modeled by Gaussian noise. The autocorrelation of the channel noise is plotted in Figure 3.8. The plot demonstrates some small amount of correlation, however the large component at 0 lags suggests that the noise can be reasonably approximated as being spectrally white.

Based on the above analysis of the noise, the additive white Gaussian noise (AWGN) channel model (cf.,[51, ch. 5]) is chosen as a good approximation for our exper-

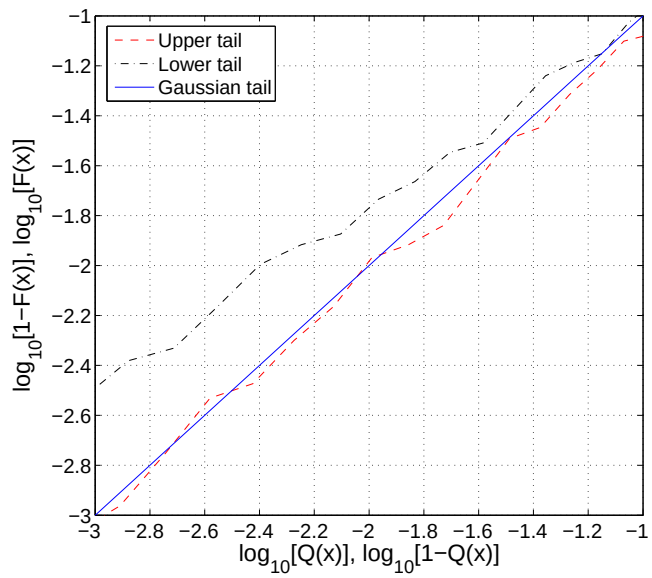


Figure 3.7: Tail probabilities of the empirical noise CDF, $F(x)$, of the channel plotted as a function of the Gaussian tail probabilities, logarithmic axes are used to show greater detail for small x.

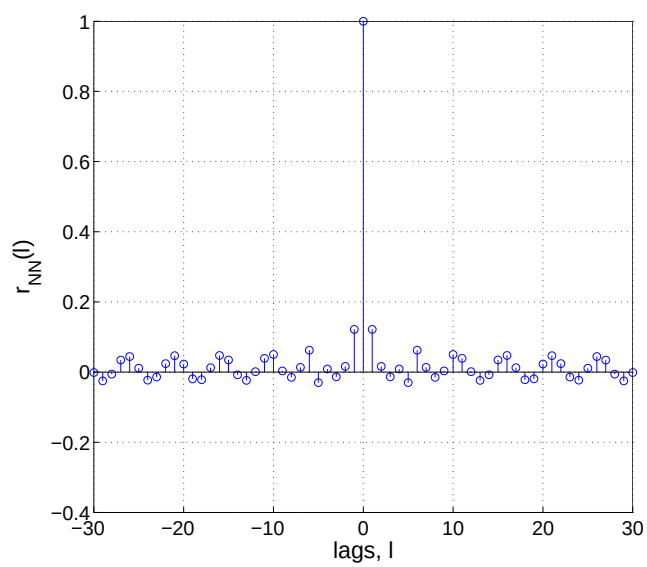


Figure 3.8: Autocorrelation of the channel noise, the large component at 0 lags suggests that the noise is approximately white with some small amount of correlation.

imental channel. We follow the mathematical description of the channel model developed in [49] for an IM/DD terrestrial FSO link using OOK modulation[2]. Let T_b denote the bit interval of the OOK system. If matched filtering is used at the receiver, the resulting electrical signal sampled at the output of the matched filter can be expressed as

$$y_i = \eta I x_i + n_i \ , \tag{3.5}$$

where η is the optical-to-electrical conversion coefficient, I is the light intensity of the received signal, $x_i \in \{0, 1\}$ is the OOK modulated signal, and n_i is the additive white Gaussian noise. The optical-to-electrical conversion coefficient is given by

$$\eta = \gamma T_b \cdot \frac{e\lambda}{hc} \ , \tag{3.6}$$

where γ is the quantum efficiency of the photodetector, e is the electron charge, λ is the signal wavelength, h is Plank's constant, and c is the speed of light (cf. ,[49, p. 1296]). The additive noise term, n_i, has mean zero and variance $\sigma_n^2 = N_0/2$, where $N_0/2$ is the noise power spectral density for real-valued AWGN. It is assumed that the noise is statistically independent from the signal component.

From (3.5) above, we define the signal component to be $s_i = \eta I x_i$. We thus define the SNR, ρ, as

$$\begin{aligned} \rho &= \frac{E\left[s_i^2\right]}{E\left[n_i^2\right]} && (3.7) \\ &= \frac{2E_s}{N_0} && (3.8) \\ &= \frac{2E\left[\eta^2 I^2 x_i^2\right]}{N_0} && (3.9) \\ &= \frac{\eta^2 I^2}{N_0} \ , && (3.10) \end{aligned}$$

where E_s is the average energy of the signal component. The factor of 2 in (3.8) arises from the fact that real signaling is used. The factor of 2 cancels in (3.10) due to the fact that $E\left[x_i^2\right] = 1/2$ for OOK modulation with equiprobable symbols. In order to compare codes with different code rates, the power efficiency of an FEC code is typically expressed

[2]The channel model developed in [49] is generalized for turbulent channels with log-normal fading. Here we use the specific case in which the variance of the log-normal fading term is 0, which is simply the AWGN channel.

in bit-energy-to-noise-density ratio, E_b/N_0, rather than SNR. This value is defined as

$$E_b/N_0 = \frac{1}{r} \cdot \frac{E_s}{N_0} \tag{3.11}$$

$$= \frac{\rho}{2r}, \tag{3.12}$$

where r is the code rate in units of information bits/coded bit.

We note that if BPSK was used instead of OOK, i.e. $x_i \in \{+1, -1\}$, then the average SNR would be $\rho_{BPSK} = 2\rho_{OOK}$. It will be useful to express an alternate form of the channel model from (3.5) in which the mean has been subtracted to produce antipodal signaling and the noise is normalized to variance one. We thus define the normalized received sequence, $\bar{y}_i$, as follows:

$$\bar{y}_i = \frac{y_i - \mu_i}{\sigma_n} = \sqrt{\frac{2}{N_0}}\left(y_i - \mu_i\right), \tag{3.13}$$

where μ_i is the mean of the received sequence, y_i, given by

$$\begin{aligned} \mu_i &= E\left[y_i\right] \\ &= \frac{\eta I}{2} \\ &= \frac{\sqrt{\rho N_0}}{2}\ . \end{aligned} \tag{3.14}$$

This results in an alternate expression of the channel model given by

$$\bar{y}_i = \sqrt{\frac{\rho}{2}}\bar{x}_i + \bar{n}_i, \tag{3.15}$$

where $\bar{x}_i \in \{+1, -1\}$ and $\bar{n}_i$ is additive white Gaussian noise with mean zero and variance one.

The constellation-constrained capacity is a fundamental limit that gives the optimum trade-off between channel utilization and power efficiency for a specific modulation scheme. It provides a basis for comparing the performance of the coded system with the best possible performance predicted by information theory. A derivation of the constellation-constrained capacity for OOK over the AWGN channel is presented below. For convenience, the equivalent channel model given by (3.15) is used. The probability density function of the received sequence, $\bar{y}_i$ is given by

$$\begin{aligned} p(\bar{y}_i) &= \frac{1}{2}\left[\frac{1}{\sqrt{2\pi}}e^{-\frac{1}{2}(\bar{y}_i - \sqrt{\frac{\rho}{2}})^2} + \frac{1}{\sqrt{2\pi}}e^{-\frac{1}{2}(\bar{y}_i + \sqrt{\frac{\rho}{2}})^2}\right] \\ &= \frac{1}{\sqrt{2\pi}}e^{\frac{-\bar{y}_i^2 - \frac{\rho}{2}}{2}} \cosh\left(\bar{y}_i\sqrt{\frac{\rho}{2}}\right), \end{aligned} \tag{3.16}$$

where ρ is the SNR as defined above. The differential entropy of $\bar{y}_i$ in nats is then given by

$$\begin{aligned} h(\bar{y}_i) &= -\int p(\bar{y}_i)\ln p(\bar{y}_i)d\bar{y}_i \\ &= \int p(\bar{y}_i)\left\{\frac{\bar{y}_i^2}{2}+\frac{\rho}{4}+\frac{1}{2}\ln(2\pi)-\ln\left[\cosh\left(\bar{y}_i\sqrt{\frac{\rho}{2}}\right)\right]\right\}d\bar{y}_i \\ &= \frac{\rho}{2}+\frac{1}{2}\ln(2\pi e)-\frac{e^{\frac{-\rho}{4}}}{\sqrt{2\pi}}\int e^{\frac{-\bar{y}_i^2}{2}}\cosh\left(\bar{y}_i\sqrt{\frac{\rho}{2}}\right)\ln\left[\cosh\left(\bar{y}_i\sqrt{\frac{\rho}{2}}\right)\right]d\bar{y}_i \\ &= \frac{\rho}{2}+\frac{1}{2}\ln(2\pi e)-\frac{e^{\frac{-\rho}{4}}}{\sqrt{2\pi}}\int e^{\frac{-t^2}{2}}\cosh\left(t\sqrt{\frac{\rho}{2}}\right)\ln\left[\cosh\left(t\sqrt{\frac{\rho}{2}}\right)\right]dt \end{aligned} \tag{3.17}$$

Now, the constellation-constrained capacity for OOK in units of data bits/pulse is given by

$$\begin{aligned} C_{OOK}(\rho) &= h(\bar{y}_i)-h(\bar{y}_i|\bar{x}_i) \\ &= h(\bar{y}_i)-\frac{1}{2}\log_2(2\pi e) \\ &= \frac{\rho}{2}\log_2(e)-\frac{e^{\frac{-\rho}{4}}}{\sqrt{2\pi}}\int e^{\frac{-t^2}{2}}\cosh\left(t\sqrt{\frac{\rho}{2}}\right)\log_2\left[\cosh\left(t\sqrt{\frac{\rho}{2}}\right)\right]dt \end{aligned} \tag{3.18}$$

From (3.12), we apply the substitution

$$\frac{E_b}{N_0}=\frac{\rho}{2C_{OOK}} \tag{3.19}$$

to (3.18) to obtain an implicit relationship between the constellation-constrained capacity, C_{OOK} (in bits/pulse), and E_b/N_0. Thus, by evaluating both C_{OOK} and E_b/N_0 for different values of the SNR, ρ, it is possible to determine the best possible E_b/N_0 for any given code rate, $C_{OOK}=r$.

3.4 SNR Estimation

Before we can assess the performance of the experimental system, it is necessary to estimate the SNR of the received signal. In general, the SNR will depend on many aspects of the electronics, the channel, and the receiver front end. For example, the transmitter power can be changed at the laser or by attenuating the beam prior to entering the water. The signal will also be attenuated by the water channel through absorption and scattering, which is a function of the water turbidity. Additional attenuation may be introduced by the polycarbonate windows through which the light propagates, which may depend on how well they are cleaned. At the receiver, the SNR can depend on the gain setting, field of view, and quantum efficiency of the photodetector.

To provide a consistent point of reference, the SNR was measured directly from the received signal at the output of the matched filter. Again, the AWGN channel model is used as described in Section 3.3. Using this theoretical model, the received SNR can be estimated using a maximum-likelihood (ML) estimator as described in [50]. Since this SNR estimator relies on knowledge of the transmitted data, it is designated as a *data-aided* (DA) estimator. In order to obtain accurate estimates of SNR, particularly at the very low SNRs where turbo and LDPC codes operate, perfect knowledge of the transmitted sequence is used. Although this is useful in a laboratory setting, it should be noted that a practical system would need to utilize either a blind SNR estimator or an DA SNR estimator based on receiver decisions. A derivation of the ML SNR estimator is presented below. For these calculations, we assume that the signal and noise are statistically independent. We begin by calculating the normalized received sequence as defined in (3.15). The second moment of $\bar{y}_i$ is given by

$$E\left[\bar{y}_i^2\right] = \frac{\rho}{2} + 1 \ , \tag{3.20}$$

and the correlation of $\bar{y}_i$ with the transmitted sequence, $\bar{x}_i$, is given by

$$\begin{aligned} E\left[\bar{y}_i\bar{x}_i\right] &= E\left[\left(\sqrt{\frac{\rho}{2}}\bar{x}_i + \bar{n}_i\right)\bar{x}_i\right] \\ &= \sqrt{\frac{\rho}{2}} \end{aligned} \tag{3.21}$$

where perfect knowledge of the transmitted sequence, $\bar{x}_i$, is used in (3.21). By combining (3.20) and (3.21), we obtain the following equation:

$$\frac{E\left[\bar{y}_i\bar{x}_i\right]^2}{E\left[\bar{y}_i^2\right]} = \frac{\frac{\rho}{2}}{\frac{\rho}{2}+1} \ . \tag{3.22}$$

Solving (3.22) for ρ, we have

$$\rho = \frac{2E\left[\bar{y}_i\bar{x}_i\right]^2}{E\left[\bar{y}_i^2\right] - E\left[\bar{y}_i\bar{x}_i\right]^2} \ . \tag{3.23}$$

It is assumed that x_i and n_i are ergodic and so the mean and statistics in (3.20) and (3.21) can be estimated by their respective time averages as

$$\mu \approx \frac{1}{N}\sum_{i=1}^{N} y_i \ , \tag{3.24}$$

$$E\left[\bar{y}_i^2\right] \approx \frac{1}{N}\sum_{i=1}^{N} \bar{y}_i^2 \ , \tag{3.25}$$

and

$$E\left[\bar{y}_i\bar{x}_i\right] \approx \frac{1}{N}\sum_{i=1}^{N}\bar{y}_i\bar{x}_i \ . \tag{3.26}$$

The estimated E_b/N_0 of the system can be calculated by applying (3.12) to the estimated SNR obtained from (3.23).

3.5 Relation of Coding Gain to Range Extension

The coding gain of an FEC system is a measure of how much power can be saved relative to uncoded OOK by using error-control coding. Since SNR decreases with distance, these savings in power can also be used to extend the range of the optical link. Recall from Section 3.1 that the relationship between optical intensity, turbidity, and distance is given by Beer's Law:

$$I = I_0 e^{-c(\lambda)d} \ , \tag{3.27}$$

where I is the detected intensity at the receiver, I_0 is the transmitted intensity, $c(\lambda)$ is the spectral attenuation coefficient as a function of wavelength, and d is optical path length in meters. A useful water metric in the measurement of range extension is the attenuation length, $c(\lambda)^{-1}$. The attenuation length is the distance at which optical intensity has been decreased by a factor of e^{-1}. For example, if $c(\lambda) = 0.5$ m^{-1}, then according to (3.27), the optical intensity is reduced by a factor of e^{-1} when $d = 2$ m. Furthermore, we define the range of an underwater FSO link expressed in units of attenuation lengths as

$$d_a = c(\lambda)d \ , \tag{3.28}$$

where d_a is a unitless measure. Substituting (3.27) and (3.28) into (3.10), we obtain the relationship

$$\rho = \frac{\eta^2 I_0^2 e^{-2d_a}}{N_0} \ . \tag{3.29}$$

Thus, (3.29) suggests that the SNR at the receiver is a decreasing function of d_a. Measuring the range of the system in attenuation lengths, d_a, has the advantage that the coding gain can easily be extrapolated to various distances and water turbidities.

As a basis for comparing experimental range extension with that predicted by theory, it is useful to calculate the expected range extension, Δd_a, based on the coding gain in SNR using (3.10) and (3.27). To achieve a given target BER, the coded system requires

a lower received optical intensity, say I_{coded}, than that of the uncoded system, $I_{uncoded}$. Assuming that the noise power in both systems is equal, we can relate the coding gain in SNR to the received optical intensities by

$$\begin{aligned}
\Delta\rho_{dB} &= \rho_{uncoded}(\text{dB}) - \rho_{coded}(\text{dB}) && (3.30)\\
&= 10\log_{10}\left(\frac{\eta^2 I_{uncoded}^2}{N_0}\right) - 10\log_{10}\left(\frac{\eta^2 I_{coded}^2}{N_0}\right) && (3.31)\\
&= 20\log_{10}\left(\frac{I_{uncoded}}{I_{coded}}\right), && (3.32)
\end{aligned}$$

where $\Delta\rho_{dB}$ is the coding gain in terms of SNR in dB. The uncoded and coded received optical intensities are related to d_a according to (3.27) as

$$I_{uncoded} = I_0 e^{-d_{a,uncoded}} \tag{3.33}$$

and

$$I_{coded} = I_0 e^{-d_{a,coded}}\ , \tag{3.34}$$

where I_0 is the optical power at the transmitter, which is set to 100 mW. The coding gain in terms of range extension is defined as $\Delta d_a = d_{a,coded} - d_{a,uncoded}$. By substituting (3.33) and (3.34) into (3.32) and changing the logarithm, we get

$$\begin{aligned}
\Delta\rho(\text{dB}) &= 20\log_{10}(e)\cdot\ln\left(\frac{e^{-d_{a,uncoded}}}{e^{-d_{a,coded}}}\right)\\
&= 20\log_{10}(e)\left[(-d_{a,uncoded}) - (-d_{a,coded})\right]\\
\Delta d_a &= \frac{\Delta\rho(\text{dB})}{20\log_{10}(e)}. && (3.35)
\end{aligned}$$

For the system used in this work, the optical path length d is equal to the length of the water tank, 3.66 m. Using the relationship $d_a = c(\lambda)d$, we can express (3.35) terms of change in the attenuation coefficient as

$$\Delta c(\lambda) = \frac{1}{3.66\ \text{m}}\cdot\frac{\Delta\rho(\text{dB})}{20\log_{10}(e)}\ . \tag{3.36}$$

For each underwater experiment, the SNR is estimated using the algorithm in Section 3.4 and the attenuation coefficient is measured using a transmissometer for each transmitted packet. However, as mentioned in Section 3.2, the transmissometer operates at a wavelength of 530 nm, whereas the system operates at a wavelength of 405 nm. Since the attenuation coefficient is a function of wavelength, the values of $c(\lambda)$ measured by the

transmissometer will be slightly different than those of the system. For this reason, we would expect that measured values of $c(530\text{ nm})$ taken by the transmissometer will not necessarily follow the relationship given by (3.36). Based on the discussion of the propagation of light through water in Section 3.1, light propagating through water at 405 nm (the blue-violet edge of the blue-green window) will suffer greater attenuation than light propagating at 530 nm (green light) when there is a significant amount of scattering, as is the case for the turbidities emulated in the tank. Thus, we would expect the relationship

$$c(405\text{ nm}) > c(530\text{ nm})\ . \tag{3.37}$$

In general, the relationship between $c(405\text{ nm})$ and $c(530\text{ nm})$ will depend on many aspects of the water conditions, including the size of scattering particles, the amount of scattering, and the amount of absorption. Thus, reliably converting gains in $d_a = c(530\text{ nm})d$ into gains in $d_a = c(405\text{ nm})d$ would require further experimental investigation and is beyond the scope of this thesis. For this reason, experimental gains in d_a presented in this thesis are in terms of measured attenuation coefficient values from the transmissometer at a wavelength $\lambda = 530$ nm. However, based on (3.37), we note that these values are likely to be conservative.

In order to observe the relationship between empirical attenuation coefficient values, $c(530\text{ nm})$, and estimated SNR, plots of $c(530\text{ nm})$ vs. SNR were generated for each set of experimental results. These plots are shown in Appendix A. From these plots, we observe that the empirical relationship between $c(530\text{ nm})$ and SNR is approximately linear. To facilitate comparison, a linear least squares regression was fitted to each set of data. Outliers at very low and very high SNR were omitted. The resulting regression lines for each experiment are shown in Figure 3.9. Each line is of the form

$$c(530\text{ nm}) = m\rho(\text{dB}) + b\ , \tag{3.38}$$

where $\rho(\text{dB})$ is the estimated SNR in dB, m is the slope, and b is the y-intercept (i.e., the value of $c(530\text{ nm})$ for $\rho = 0$ dB). The values of m and b for each regression, along with lower and upper bounds on the 95% confidence interval, are summarized in Table 3.2.

From these data, we observe a consistent slope of $m \approx 0.020$ but a variation in y-intercept, b, between experiments. Thus, we conclude that the relationship between $c(530\text{ nm})$ and $\rho(\text{dB})$ is approximately linear across the range of $c(\lambda)$ values and SNRs emulated in the tank and that the relationship between $\Delta c(530\text{ nm})$ and $\Delta\rho(\text{dB})$ remains

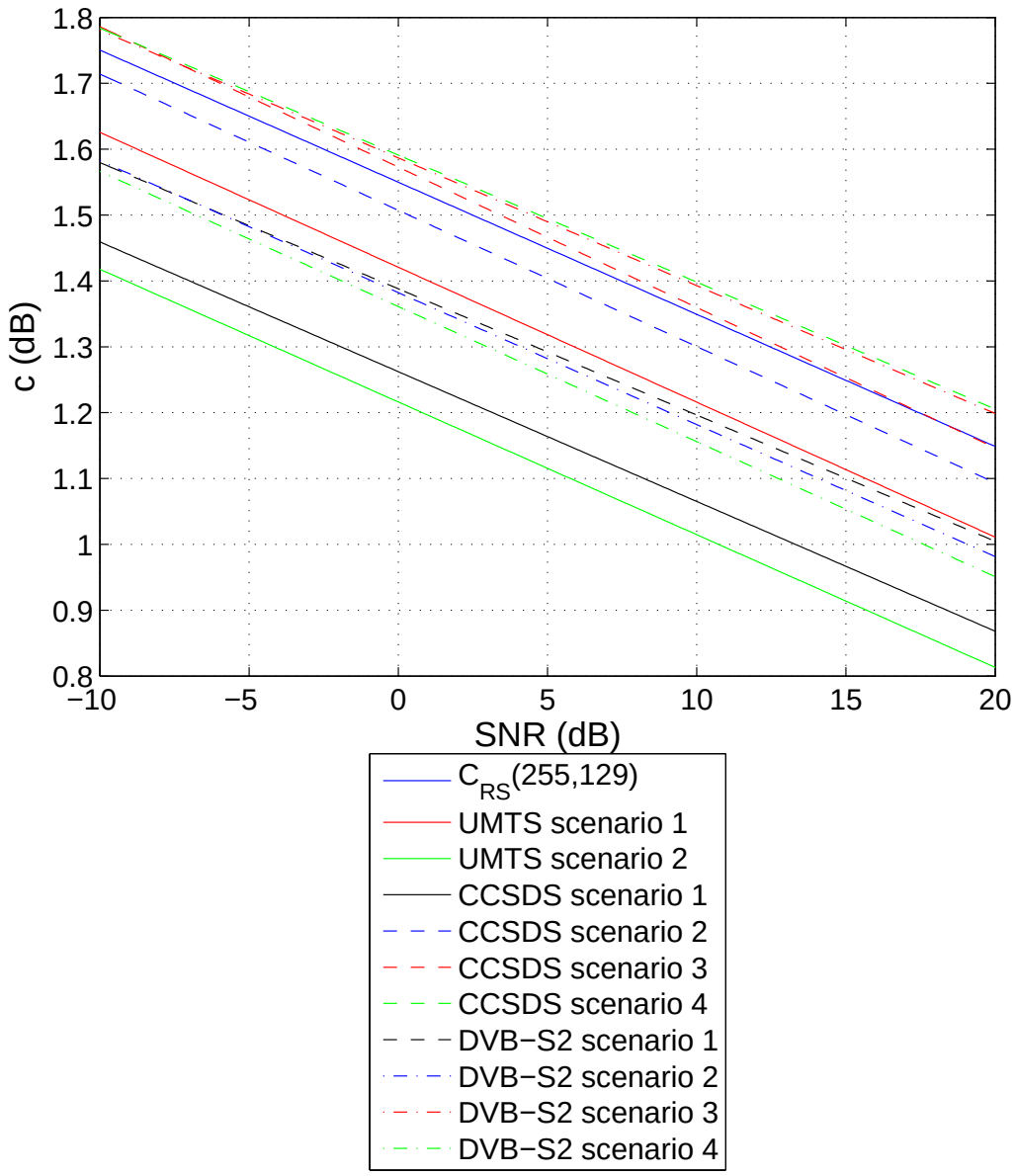


Figure 3.9: Empirical relationship between SNR and measured $c(530\text{ nm})$ based on linear regression.

Table 3.2: Summary of linear regression data.

Experiment	m	m (upper)	m (lower)	b	b (upper)	b (lower)
$\mathcal{C}_{RS}(255, 129)$	-0.0201	-0.0203	-0.0199	1.55	1.55	1.56
UMTS scenario 1	-0.0205	-0.0207	-0.0203	1.42	1.42	1.42
UMTS scenario 2	-0.0202	-0.0204	-0.0199	1.22	1.21	1.22
CCSDS scenario 1	-0.0197	-0.0200	-0.0194	1.26	1.26	1.26
CCSDS scenario 2	-0.0207	-0.0209	-0.0205	1.51	1.51	1.51
CCSDS scenario 3	-0.0213	-0.0215	-0.0211	1.57	1.57	1.58
CCSDS scenario 4	-0.0193	-0.0195	-0.0191	1.59	1.59	1.59
DVB-S2 scenario 1	-0.0192	-0.0196	-0.0188	1.39	1.38	1.39
DVB-S2 scenario 2	-0.0200	-0.0203	-0.0197	1.38	1.38	1.39
DVB-S2 scenario 3	-0.0194	-0.0196	-0.0192	1.59	1.58	1.59
DVB-S2 scenario 4	-0.0205	-0.0208	-0.0203	1.36	1.36	1.36

constant between experiments. The y-intercepts, however, vary between experiments. This is to be expected, as this value depends on many aspects of the electronics, the channel, and the overall system. The consistent linear relationship between c(530 nm) and ρ(dB) may indicate that Beer's Law is applicable across the water conditions observed in the tank for these experiments, but with a different slope than that predicted by (3.36) due to the difference in wavelength between the transmissometer and the underwater FSO link. However, this is a topic that requires further investigation and is beyond the scope of this thesis.

Chapter 4

Reed-Solomon Coding

In this chapter we investigate the application of Reed-Solomon (RS) forward-error correction codes to the underwater optical communication channel. A RS code was initially implemented for its simplicity and robustness. RS codes offer a good starting point because, unlike turbo and LDPC codes, their implementation does not require prior knowledge of the channel characteristics. In Section 4.1, we begin by comparing theoretical bounds with simulated performance of various RS codes over $GF(256)$ and $GF(512)$. Then in Section 4.2, an implementation of the $\mathcal{C}_{RS}(255, 129)$ code is presented along with an analysis of experimental results. Finally in Section 4.3, significant results are summarized and concluding remarks are made.

4.1 Reed-Solomon Theory and Simulation

As discussed in Chapter 2, Reed-Solomon codes are a class of non-binary cyclic block codes operating over $GF(q)$ where, for practical codes, $q = 2^m$. An RS code has a block length $n = q - 1$ symbols and dimension $k = q - 1 - 2t$ symbols, where t is the maximum number of symbol errors the code is designed to correct. We will denote this code by $\mathcal{C}_{RS}(n, k)$. Furthermore, RS codes are maximum distance separable, which means they have minimum distance $d_{min} = n - k + 1$ and are capable of correcting up to $t = 0.5(n - k)$ symbol errors in a single codeword. The $\mathcal{C}_{RS}(n, k)$ encoder maps k m-bit information symbols onto n m-bit code symbols. Thus, one q-ary symbol is represented by m OOK bits. For this investigation, the Berlekamp-Massey-Forney algorithm is used for

decoding, which operates on hard-decision received data [22, pp. 128-136].

In Chapter 3 we presented experimental results that suggest the underwater optical communication channel may be well modeled by the AWGN channel model. The uncoded probability of bit error for this channel as a function of SNR, ρ, when OOK modulation is used is given by

$$P_b = Q\left(\sqrt{\frac{\rho}{2}}\right) \ , \tag{4.1}$$

where Q is the q-function (cf., [51, p. 40]). If the source data is encoded by a $\mathcal{C}_{RS}(n,k)$ code, the probability of an m-bit symbol error is

$$P_s = \sum_{i=1}^{m} \binom{m}{i} P_b^i \left(1 - P_b\right)^{m-i} \ . \tag{4.2}$$

Assuming all the bits in an uncorrectable error are decoded in error results in the following upper bound for the bit-error rate of the RS coded data:

$$P_{b,coded} \leq \sum_{i=t+1}^{n} \binom{n}{i} P_s^i \left(1 - P_s\right)^{n-i} \ , \tag{4.3}$$

where t is again the designed error-correction capability of the code (cf., [51, p. 465]). From (3.12), we can apply the relation

$$\rho = 2r\frac{E_b}{N_0} \tag{4.4}$$

to express the upper bound in (4.3) as a function of E_b/N_0.

Simulations were run in MATLAB to investigate the BER of RS codes for OOK modulation and two typical Galois fields, $GF(256)$ and $GF(512)$. Code rates of $r = 1/2$, $r = 1/3$, $r = 1/4$, and $r = 1/6$ were chosen to allow direct comparison with the turbo and LDPC codes discussed in subsequent chapters. The simulated codes and their corresponding parameters are summarized in Table 4.1. Simulation results are shown in Figures 4.1 and 4.2 for codes over $GF(256)$ and $GF(512)$, respectively. The coding gains that can potentially be achieved by each code in terms of E_b/N_0 are summarized in column 4 of Table 4.1.

From the simulation results, we note that the simulated performance agrees with the theoretical upper bounds predicted by (4.3) for small BERs. We also make two other observations: First, we notice that the codes with the best performance in terms of E_b/N_0 from this set of RS codes are the rate 1/2 codes. These codes also provide better bandwidth efficiency than the lower rate codes. Second, we notice that, at a BER of 10^{-4} (the regime

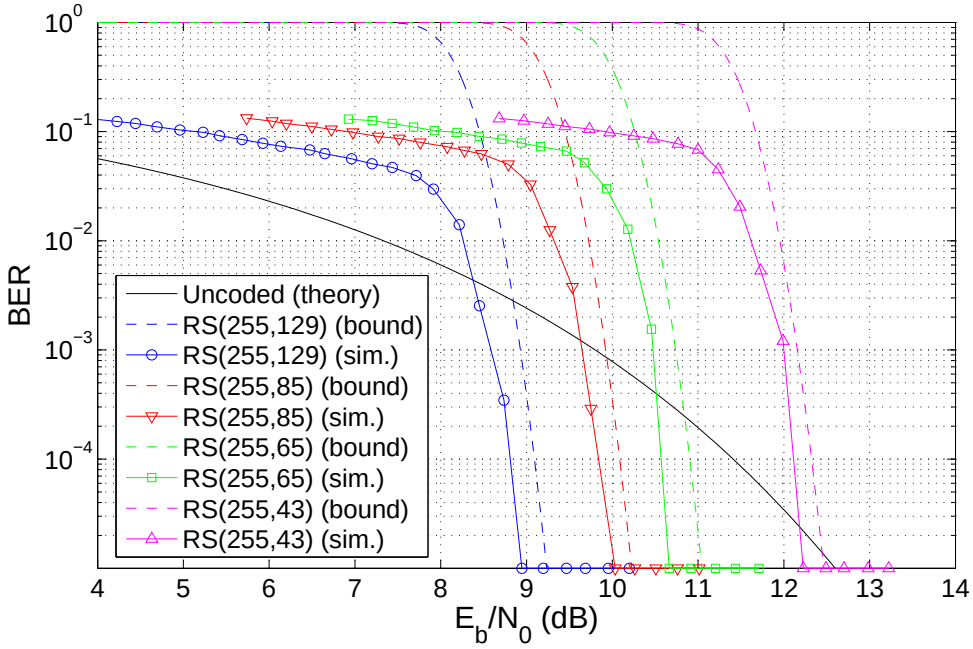


Figure 4.1: RS simulation results for codes over $GF(256)$.

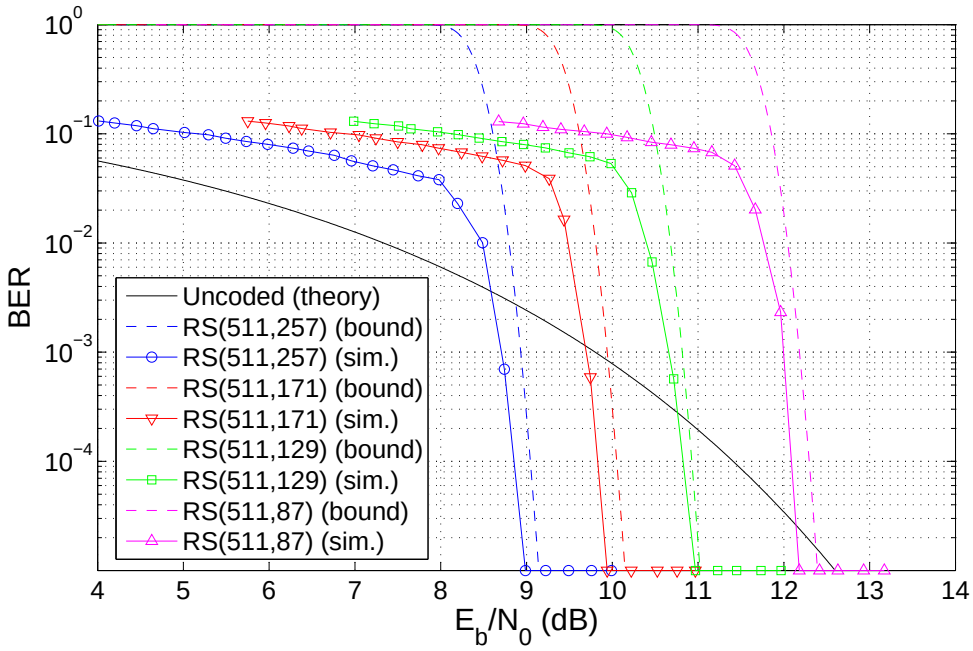


Figure 4.2: RS simulation results for codes over $GF(512)$.

Table 4.1: Summary of RS code parameters and corresponding gains at BER= 10^{-4}. Code rates are approximate. Coding gains are in terms of E_b/N_0.

Code	**Rate**, r	**Error correction**, t	**Coding Gain (dB)**
$\mathcal{C}_{RS}(255, 129)$	1/2	63	2.28
$\mathcal{C}_{RS}(255, 85)$	1/3	85	1.48
$\mathcal{C}_{RS}(255, 65)$	1/4	95	0.75
$\mathcal{C}_{RS}(255, 43)$	1/6	106	−0.79
$\mathcal{C}_{RS}(512, 257)$	1/2	127	2.47
$\mathcal{C}_{RS}(512, 171)$	1/3	170	1.46
$\mathcal{C}_{RS}(512, 129)$	1/4	191	0.48
$\mathcal{C}_{RS}(512, 87)$	1/6	212	−0.76

of interest for this system), the performance of the $\mathcal{C}_{RS}(255, 129)$ and the $\mathcal{C}_{RS}(512, 257)$ codes are comparable. Thus, without significant loss of power efficiency, the $\mathcal{C}_{RS}(255, 129)$ code offers lower latency at the receiver due to smaller block length. Additionally, it has the advantage that 8-bit symbols are used instead of 9-bit symbols, which makes implementation in a practical system simpler. Both of these aspects make the $\mathcal{C}_{RS}(255, 129)$ code a better option for future implementation in a real-time system.

4.2 Experimental Results

Based on simulated performance in the previous section, the $\mathcal{C}_{RS}(255, 129)$ code was selected for experimentation. This code has a rate of approximately 1/2 and is capable of correcting up to 63 symbol errors in each codeword. Experimental results were obtained using the underwater testbed and procedure described in Section 3.2. The RS encoder and decoder were implemented in MATLAB using the communications toolbox. For the $\mathcal{C}_{RS}(255, 129)$ code, the encoder maps 129 8-bit information symbols onto 255 8-bit coded symbols. This is equivalent to a binary dimension of $km = 1032$ information bits and block length $nm = 2040$ coded bits per block.

The experimental packet is generated and saved prior to experimentation. In order to obtain a BER estimate approaching 10^{-4}, on the order of 100,000 information bits should be sent per packet. Due to limitations in buffer size, the data packets are limited in size to 200,000 bits, including a 1024 bit packet header. To maximize the number of information bits sent per packet, the largest multiple of the block length that is below this limit is used.

This results in 197,880 coded bits (approx. 24 KB) and 100,104 information bits per packet. In order to have an accurate basis for calculating empirical coding gains, it is necessary to measure the performance of the uncoded system. To avoid sending two packets for the same water condition, the coded packet is used to generate an estimate of the uncoded BER by simply calculating the BER of the coded data prior to decoding.

The accuracy achieved using the packet size given above can be quantified in terms of confidence intervals. The 95% confidence interval, denoted (y_+, y_-), is defined as the interval about the BER estimate, $\hat{p}$, such that

$$\text{Prob}\left[y_+ \leq p \leq y_-\right] = 0.95, \tag{4.5}$$

where y_+ is the upper bound on the confidence interval, y_- is the lower bound on the confidence interval, and p is the actual BER. For a target BER estimate, $\hat{p} = 10^{-k}$, and packet size, $N = b \cdot 10^k$ bits, the bounds on the 95% confidence interval are closely approximated using the following expression [52]:

$$y_\pm = 10^{-k}\left\{1 + \frac{3.84}{2b}\left[1 \pm \sqrt{\frac{4b}{3.84} + 1}\right]\right\} . \tag{4.6}$$

Thus, for a BER estimate of $\hat{p} = 10^{-4}$ and the packet size above, the 95% confidence interval is $(1.84\hat{p}, 0.543\hat{p})$ for the coded system and $(1.55\hat{p}, 0.65\hat{p})$ for the uncoded system.

4.2.1 BER Performance

Experimental measurements of BER vs. E_b/N_0 are shown in Figure 4.3. The uncoded data are represented by blue circles and the coded data are represented by red triangles. Points plotted on the x axis have a BER $\leq 10^{-5}$. Also shown for comparison are the theoretical BER for the uncoded system given by (4.1) and the upper bound on the coded system given by (4.3), both of which assume an ideal AWGN channel. From the figures, we see that the experimental results for the coded system agree with theory, but perform roughly 0.4 dB worse than simulation at a BER of 10^{-4}. Likewise, the experimental results for the uncoded system perform close to theory but deviate slightly at high SNR, performing a little less than 1 dB worse than theory at a BER of 10^{-4}. The resulting coding gain in E_b/N_0 is approximately 2.5 dB. Since both coded and uncoded curves perform slightly worse than theory, the experimental coding gain is approximately the same as that predicted by

simulation. The fact that the experimental data agrees closely with theory suggests that the AWGN channel model may be a good model of the underwater optical communications link. However, the small deviation of both the coded and uncoded experimental data from the theory curve at high SNR suggests a deviation from the AWGN channel model. This could be caused by a number of factors. It could be due to non-Gaussian sources of noise or the presence of weak fading due to turbulence. It could also be caused by the experimental setup itself, possibly due to quantization error at high signal levels or imprecise synchronization. Based on these experimental results, the cause is yet unclear but is beyond the scope of this thesis.

4.2.2 Range Extension

In the previous section, we saw that the $\mathcal{C}_{RS}(255, 129)$ provides a moderate improvement in the power efficiency, E_b/N_0, for an underwater optical communication link. We now examine how this improvement in power efficiency can extend the range of reliable underwater communication. To do so, we make use of Beer's Law given by (3.4), which relates optical intensity to water turbidity and link distance. In Section 3.5, we established the link distance expressed in units of attenuation lengths, $d_a = c(\lambda)d$, as a water metric for quantifying the range extension of a FEC-coded system. Expressing the range of the system in units of attenuation lengths has the advantage that the coding gain can easily be extrapolated to various distances and water turbidities.

In Section 3.5 we showed that an expected coding gain in d_a can be calculated based on coding gain in SNR using Beer's Law and the theoretical channel model from Section 3.3. This relationship is given by (3.35). Substituting the empirical coding gain in SNR observed in Section 6.3.1 into (3.35), we calculate the expected coding gain in d_a to be approximately 0.63. We reiterate, that this expected gain is in terms of $d_a = c(405 \text{ nm})d$, i.e. the range extension in attenuation lengths is evaluated at the wavelength of the experimental system, $\lambda = 405$ nm. These values will differ from the empirical gains calculated using measured attenuation coefficient values from the transmissometer, which operates at $\lambda = 530$ nm (cf., Section 3.5). Experimental results are presented here in terms of $d_a = c(530 \text{ nm})d$ using measured attenuation coefficient values from the transmissometer. However, these values are likely to be conservative relative to range extension of the actual system.

Experimental results for BER vs. d_a are shown in Figure 4.4. The coding gain

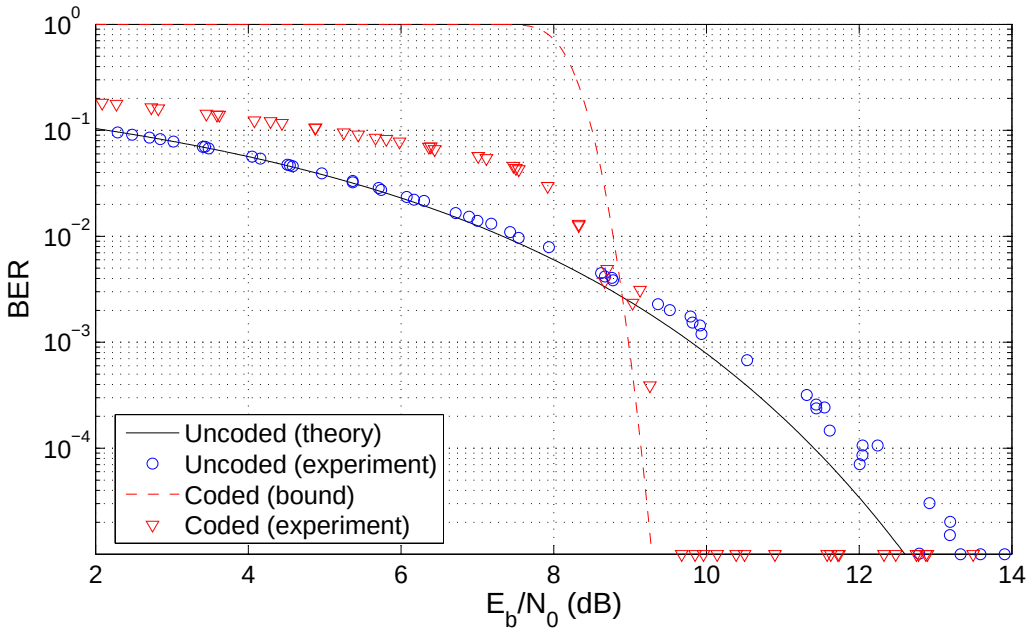


Figure 4.3: Experimental BER vs. E_b/N_0 results for the $\mathcal{C}_{RS}(255, 129)$ code.

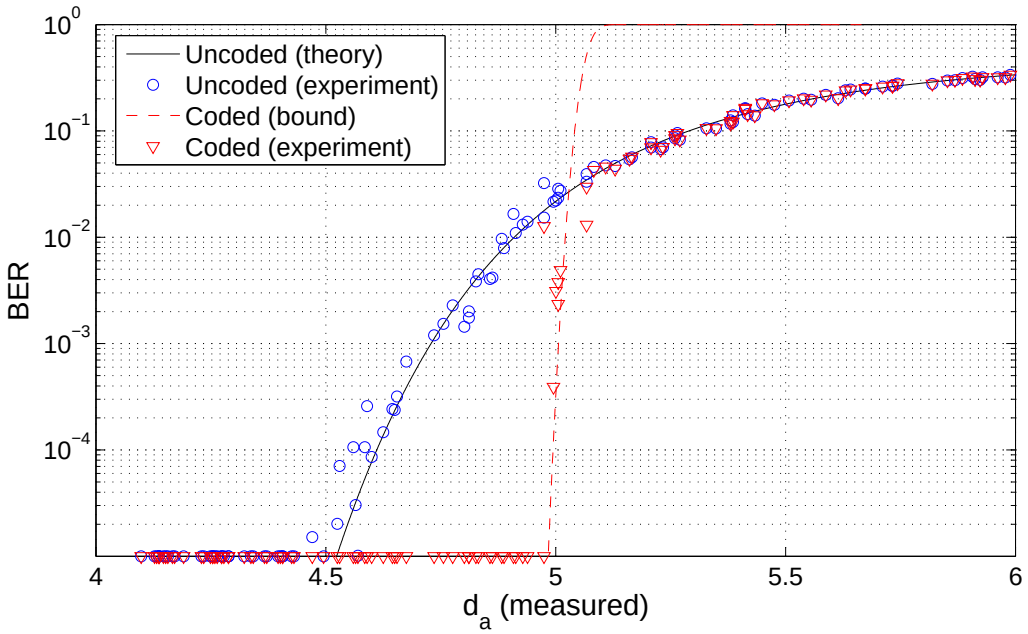


Figure 4.4: Experimental BER vs. d_a results for the $\mathcal{C}_{RS}(255, 129)$ code.

Table 4.2: Summary of range extension for the $\mathcal{C}_{RS}(255, 129)$ RS.

Water Type	$c(\lambda)$	**Range Extension (m)**
Clear ocean	0.1514	2.61
Coastal ocean	0.399	0.99
Harbor water	2.195	0.18

in terms of attenuation lengths is approximately 0.40. Theory and simulation curves as a function of d_a were calculated as follows. From the experimental data, the measured $c(530 \text{ nm})$ values from the transmissometer were first plotted as a function of estimated SNR. This plot, shown in Appendix A, demonstrates a linear relationship between ρ in dB and $c(530 \text{ nm})$. To establish this relationship explicitly, a linear, least-square regression was used to determine a line of best fit. Using this line, it is then possible to map SNR for the theory curves into d_a for each experiment.

Using the experimental gain in d_a, we can extrapolate the potential range extension that could be attained using these FEC codes. We examine the potential range extension for three different water conditions: clear ocean water ($c(\lambda) = 0.1514 \text{ m}^{-1}$), coastal water ($c(\lambda) = 0.399 \text{ m}^{-1}$), and turbid harbor water ($c(\lambda) = 2.195 \text{ m}^{-1}$). These attenuation coefficient values are based on measurements in [46] that were made in a spectral band centered at $\lambda \approx 530$ nm. Using the relationship

$$d = \frac{d_a}{c(530 \text{ nm})} \tag{4.7}$$

and assuming that Beer's Law holds at each value of $c(\lambda)$, we are able to predict the range extension achieved by each code scenario in each type of water. These results are summarized in Table 5.9. We note that since the measured attenuation coefficients from the transmissometer were used, the range extension capabilities presented are likely to be conservative.

4.3 Conclusion

In this chapter we demonstrated the use of Reed-Solomon forward-error correction coding on a 500 kbps OOK RZ underwater optical communication link operating at 405 nm. The performance of rate $r = 1/2$, $r = 1/3$, $r = 1/4$, and $r = 1/6$ codes over $GF(256)$ and $GF(512)$ was investigated in simulation. Based on simulated performance, the $\mathcal{C}_{RS}(255, 129)$

code was chosen for implementation for its good power and bandwidth efficiency relative to the other RS codes investigated. This code is also a good choice for practical implementation because it operates on 8-bit symbols.

Experimental results were collected using the underwater optical testbed and experimental procedure described in Chapter 3. The experimental coding gain for the $\mathcal{C}_{RS}(255, 129)$ code was approximately 2.5 dB in E_b/N_0 at a BER of 10^{-4}, which agrees with theory and simulation. The fact that the experimental data agrees closely with theoretical performance suggests that the underwater optical channel is well modeled by the AWGN channel model. This knowledge will be useful in the implementation of turbo and LDPC codes in subsequent chapters. Using measured attenuation coefficient values, we also calculated experimental coding gain in terms of range extension, d_a, to be approximately 0.40 attenuation lengths. While the coding gains provided by this RS code are substantial, these gains are still more than 6 dB E_b/N_0 away from the information-theoretic limits of the AWGN channel. In the next two chapters we shall see that much larger coding gains are attainable through the use of newer, state-of-the-art coding schemes.

Chapter 5

Turbo Coding

In this chapter we investigate the application of turbo codes to the underwater optical communication channel. These state-of-the-art codes were chosen for their large coding gains and ability to operate at very low SNR. Two standardized turbo codes are investigated and implemented. The first code is the Universal Mobile Telecommunications System (UMTS) turbo code used in third generation (3G) mobile telecommunications. This code was chosen because it is widely used and offers a broad range of frame sizes which provide trade-offs between power efficiency and latency at the decoder. The second code is the Consultative Committee for Space Data Systems (CCSDS) turbo code used in space telemetry. This code was chosen because it provides multiple code rates which provide trade-offs between power efficiency and throughput. In particular, a rate 1/6 code is specified which allows for significant improvement in power efficiency as compared to the UMTS turbo codes. This chapter is primarily divided into two sections. Section 5.1 is dedicated to the UMTS turbo code, while Section 5.2 is dedicated to the CCSDS turbo code. In each section, the coding algorithm is presented, followed by simulation and experimental results. The chapter ends with a conclusion and summary for both codes in Section 5.3

5.1 UMTS Turbo Code

5.1.1 UMTS Turbo Code Specification

The first turbo code we consider is the Universal Mobile Telecommunications System (UMTS) turbo code used in third generation mobile telecommunications [53], [54].

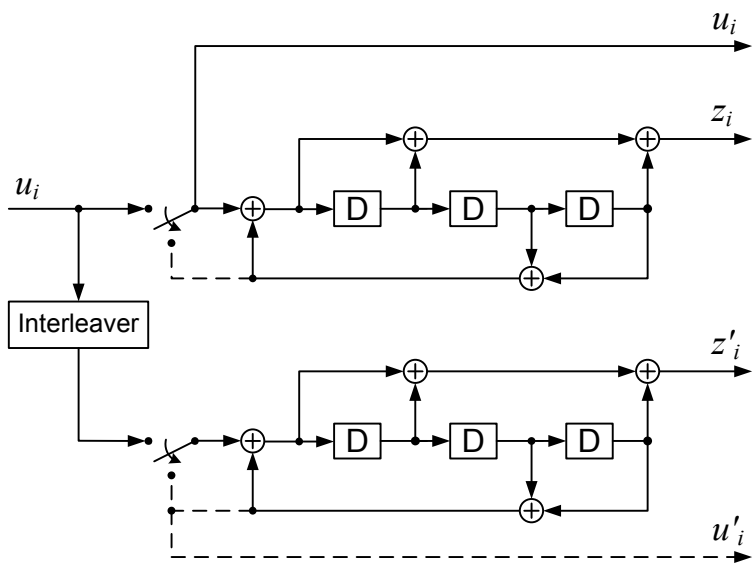


Figure 5.1: Turbo encoder for the UMTS turbo code. Adapted from [53].

The UMTS turbo encoder consists of two identical, constraint length 4 recursive systematic convolutional (RSC) encoders concatenated in parallel. For each RSC encoder, the feed-forward generator is 15 and the feedback generator is 13, both in octal. A diagram of the UMTS turbo encoder is given in Figure 5.1. Initially, the two switches are in the up position as the information bits, u_i, are read into the encoder . The output from the encoder is then multiplexed in the form:

$$\{u_1, z_1, z'_1, u_2, z_2, z'_2, \cdots, u_k, z_k, z'_k\} \tag{5.1}$$

where $\mathbf{u} = \{u_1, u_2, \cdots, u_k\}$ represents the original data sequence, $\mathbf{z} = \{z_1, z_2, \cdots, z_k\}$ represents the parity bits due to the upper RSC encoder, and $\mathbf{z}' = \{z'_1, z'_2, \cdots, z'_k\}$ represents the parity bits due to the lower RSC encoder. Thus, each input bit produces three output bits–one systematic bit and two parity bits. This results in a nominal code rate of $r = 1/3$ (not counting tail bits).

The information bits at the input to the encoder are broken up into frames of k information bits. According to the UMTS standard, any frame size between 40 and 5114 bits can be used. However, to limit our investigation to a reasonable parameter space, we will focus on eight typical frame sizes (cf. [23]). These frame sizes are presented in Table 5.1. To terminate transmission at the end of a frame, the two switches in Figure 5.1

Table 5.1: UMTS turbo code frame sizes.

Information bits, k	**Coded bits,** n
40	132
190	582
530	1602
640	1932
1060	3192
2020	6072
3460	10392
5114	15354

are thrown into the down position after all k information bits have been encoded. This forces the trellises of both constituent encoders back to the all-zero state after three bit shifts. Since the decoder performs better if both the initial state and final state of the encoders are known [23], the tail bits are then transmitted at the end of the encoded frame according to

$$\left\{u_{k+1},\ z_{k+1},\ u_{k+2},\ z_{k+2},\ u_{k+3},\ z_{k+3},\ u'_{k+1},\ z'_{k+1},\ u'_{k+2},\ z'_{k+2},\ u'_{k+3},\ z'_{k+3}\right\}\ , \tag{5.2}$$

where $\{u_{k+1}, \cdots, u_{k+3}\}$ and $\{u'_{k+1}, \cdots, u'_{k+3}\}$ represent the the feedback bits of the upper and lower constituent encoders, respectively, (i.e., the feedback bits are used as the input to each constituent encoder for three bit shifts). Note that this differs from the form of the encoded data. The first six bits alternate between the feedback and feed-forward bits of the upper constituent encoder, and the last six bits alternate between the feedback and feed-forward bits of the lower constituent encoder. Counting the tail bits, each frame of k information bits results in $n = 3(k + 4)$ coded bits. As a result, the actual code rate, $r = k/n$, is slightly lower than the nominal code rate of $r = 1/3$. However, for large frame sizes, this difference is negligible.

The UMTS turbo interleaver is an $R \times C$ matrix, where R is the number of rows and C is the number of columns. The values of R and C are chosen based on the length of the input sequence, k, as defined in [53, p. 19]. In general, R is equal to either 5, 10, or 20, and C takes on a value between 8 and 256 such that the $R \times C$ matrix is large enough to accommodate all k input bits. Data is read row-wise into the matrix, with the first data bit placed in the upper leftmost position of the matrix (i.e. row 1, column 1). If $R \times C > k$, then dummy bits are padded at the end of the input sequence. Next, each row undergoes intra-

row permutations that reorganize the column positions of the bits in a pseudo-random way. Since the intra-row permutation algorithm is rather long, the interested reader is directed to the specification for a full description [53, pp. 20-21]. After intra-row permutations, the rows are rearranged by inter-row permutations. For $R = 5$ and $R = 10$, the row ordering is simply reversed (e.g, for $R = 5$ the ordering becomes $\{5, 4, 3, 2, 1\}$). For $R = 20$, the rows take on a pseudo-random reordering that depends on the value of k. After both intra-row and inter-row permutations have been performed, the bits of the permuted matrix are read out column-wise starting with the upper leftmost bit and ending with the lower rightmost bit. Finally, dummy bits are deleted to form the interleaved sequence, $\mathbf{u}'$.

Decoding of the UMTS turbo code is done using the turbo decoder described in Section 2.3.2. The decoder operates in the log domain using log-likelihood ratios. The SISO processors decode using the log-MAP algorithm or one of its approximations, i.e., linear-log-MAP, constant-log-MAP, max-log-MAP [26]. The maximum number of decoding iterations is not specified by the UMTS technical specification, leaving this parameter as another design decision.

5.1.2 Simulation Results

To provide a theoretical baseline for the underwater experiments, we simulated the performance of the UMTS turbo code for the AWGN channel. Simulations were run in MATLAB using the Coded Modulation Library (CML) software developed by Iterative Solutions [55]. In order to compare the potential trade-offs between power efficiency and decoder complexity, each of the four decoding algorithms described in Section 2.3.2 (i.e., the log-domain BCJR algorithm and its three approximations) were simulated. Simulation results are shown in Figures 5.2 through 5.5 and are presented in order of decoder complexity as follows: log-MAP, linear-log-MAP, constant-log-MAP, and max-log-MAP. Based on the channel model developed in Section 3.3, the bit-error rate (BER) for uncoded OOK modulation in AWGN is given by

$$P_b = Q\left(\sqrt{\frac{E_b}{N_0}}\right), \tag{5.3}$$

where E_b/N_0 is the bit-energy-to-noise-density-ratio, and Q is the q-function (cf., [51, p. 40]). Uncoded OOK achieves a BER of 10^{-4} (the regime of interest for this system) at an E_b/N_0

of 11.4086 dB. The coding gain in E_b/N_0 that can potentially be obtained at this BER varies by frame size and decoding algorithm, ranging from 8.05 dB for log-MAP decoding with frame size $k = 5114$ to 4.77 dB for max-log-MAP decoding with frame size $k = 40$. These gains are summarized in Tables 5.2 through 5.5. We observe that, in general the potential coding gains tend to increase with decoder complexity.

As discussed in Section 3.3, the constellation-constrained capacity is a fundamental limit that gives the optimum trade-off between channel utilization and power efficiency for a specific modulation scheme. By comparing the above simulation results to the constellation-constrained capacity, we can observe how well these codes perform relative to the best possible performance predicted by information theory. It was shown in Section 3.3 that the constellation-constrained capacity for OOK modulation over the AWGN channel is given by

$$C_{OOK} = \frac{\rho}{2}\log_2(e) - \frac{e^{\frac{-\rho}{4}}}{\sqrt{2\pi}} \int e^{\frac{-t^2}{2}} \cosh\left(t\sqrt{\frac{\rho}{2}}\right) \log_2\left[\cosh\left(t\sqrt{\frac{\rho}{2}}\right)\right] dt \tag{5.4}$$

where C_{OOK} is the constellation-constrained capacity in units of data bits/pulse and ρ is the SNR. From (3.12), we apply the substitution

$$\frac{E_b}{N_0} = \frac{\rho}{2C_{OOK}} \tag{5.5}$$

to (5.4) to obtain an implicit relationship between the constellation-constrained capacity, C_{OOK} (in bits/pulse), and E_b/N_0. By evaluating this expression for $C_{OOK} = 1/3$ bits/pulse, we observe that the best possible performance of a $r = 1/3$ code is $E_b/N_0 = 2.52$ dB. Since the UMTS turbo code is a state-of-the-art, capacity-approaching code, it is designed to operate at an E_b/N_0 very close to this value. The distance of each code at a BER of 10^{-4} from C_{OOK} is summarized in Tables 5.2 through 5.5. We note that for large frame size, the simulated performance is within 1 dB of the constellation-constrained capacity at a BER of 10^{-4}.

The UMTS turbo codes presented here offer a number of trade-offs between power efficiency, latency, and decoder complexity. In particular, they offer a wide range of frame sizes. The best performing codes in terms of power efficiency are those that utilize large frame sizes and the log-MAP decoding algorithm. However, this performance comes at a cost of increased decoding complexity and increased latency due to the large frame size. In exchange for decreased power efficiency, it is possible to decrease system latency and/or decoder complexity. For example, decoder complexity can be decreased by using one of the

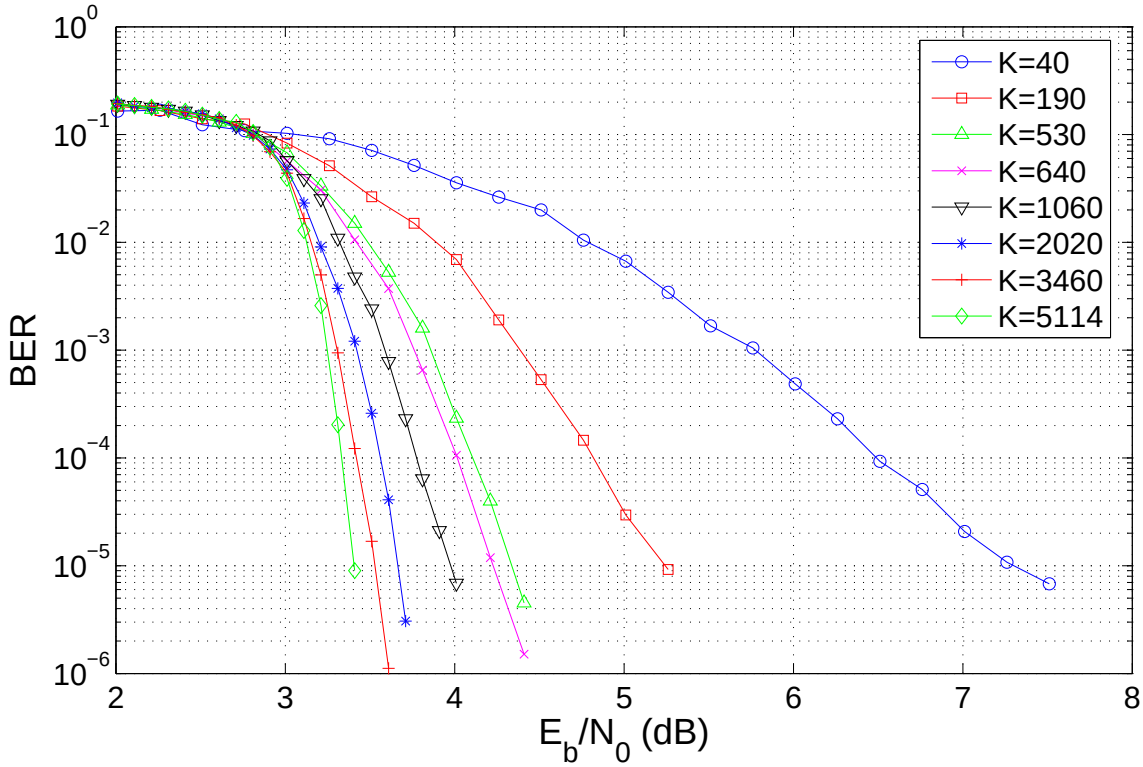


Figure 5.2: Simulated performance of the UMTS turbo code in AWGN using the log-MAP decoding algorithm, 16 decoding iterations, and variable frame size, k (BER vs. E_b/N_0).

Table 5.2: Summary of simulated performance of the UMTS turbo code in AWGN using the log-MAP decoding algorithm and 16 decoding iterations. Frame sizes are in bits. Coding gains and distance from constellation-constrained capacity, C_{OOK}, are in E_b/N_0 in dB.

Frame size (Uncoded), k	**Simulated Coding Gain (dB)**	**Distance to C_{OOK} (dB)**
40	4.91	3.98
190	6.55	2.34
530	7.26	1.63
640	7.39	1.51
1060	7.62	1.27
2020	7.83	1.07
3460	7.98	0.92
5114	8.05	0.85

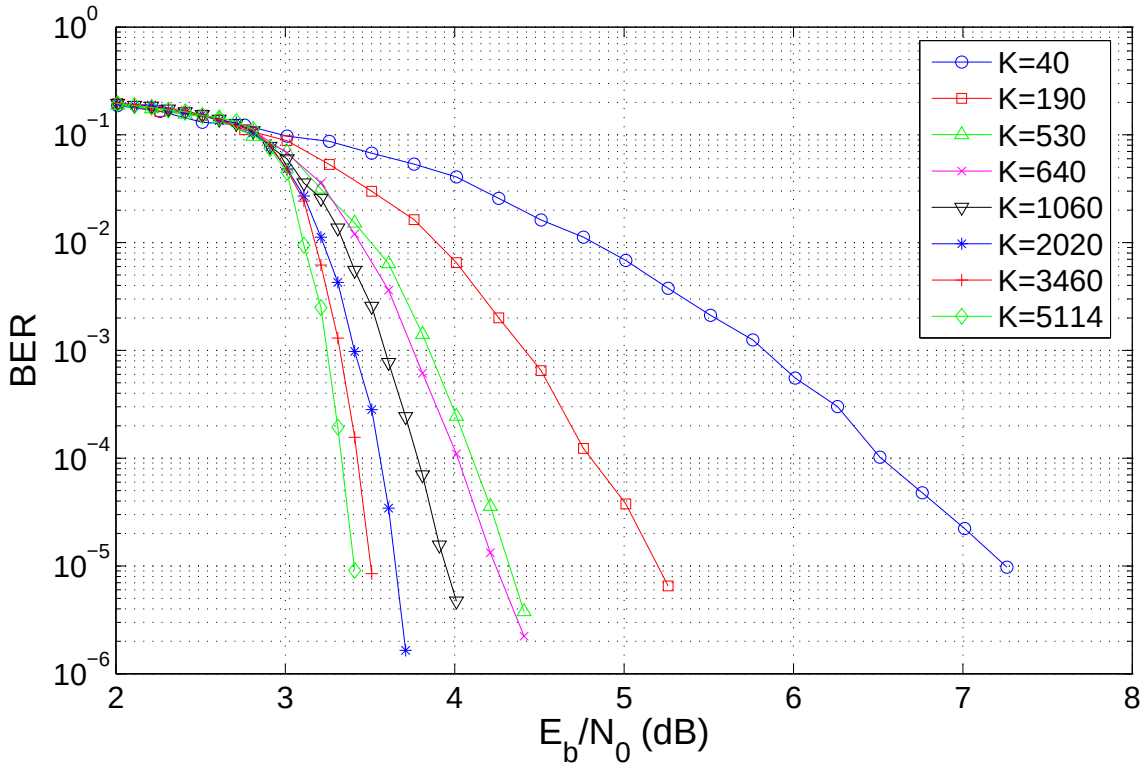


Figure 5.3: Simulated performance of the UMTS turbo code in AWGN using the linear-log-MAP decoding algorithm, 16 decoding iterations, and variable frame size, k (BER vs. E_b/N_0).

Table 5.3: Summary of simulated performance of the UMTS turbo code in AWGN using the linear-log-MAP decoding algorithm and 16 decoding iterations. Frame sizes are in bits. Coding gains and distance from constellation-constrained capacity, C_{OOK}, are in E_b/N_0 in dB.

Frame size (Uncoded), k	**Simulated Coding Gain (dB)**	**Distance to C_{OOK} (dB)**
40	4.89	4.01
190	6.58	2.31
530	7.26	1.63
640	7.38	1.51
1060	7.62	1.28
2020	7.82	1.07
3460	7.96	0.93
5114	8.05	0.85

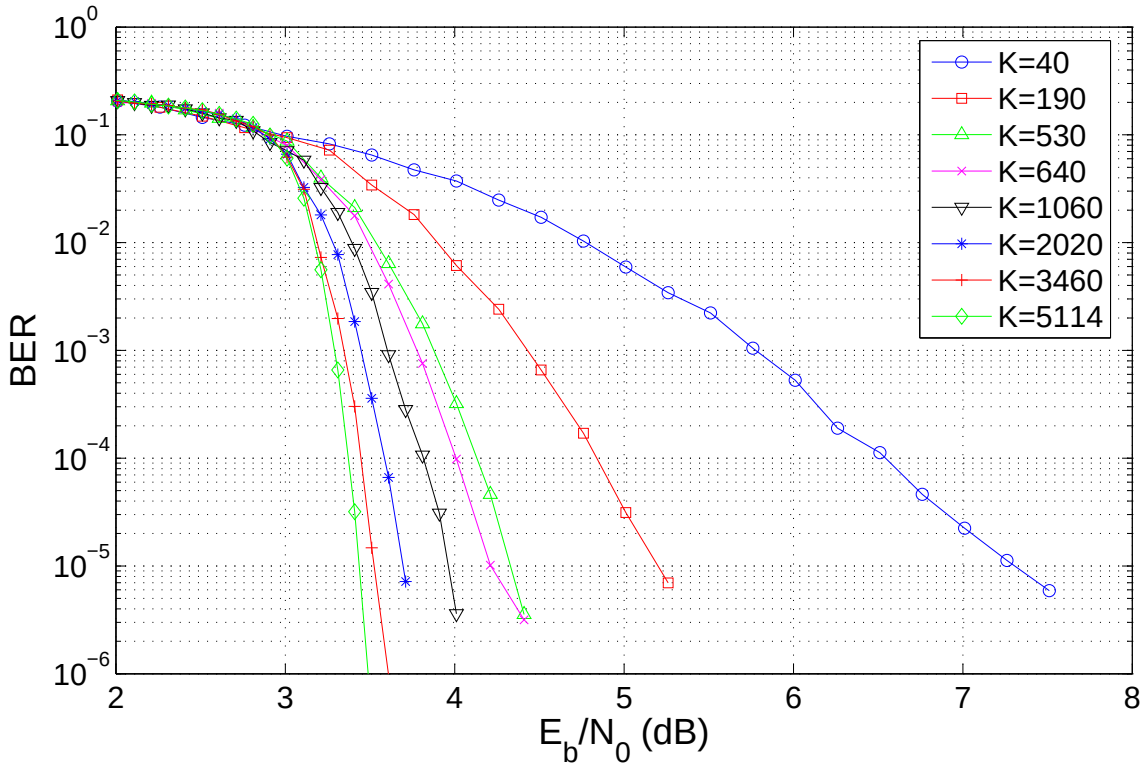


Figure 5.4: Simulated performance of the UMTS turbo code in AWGN using the constant-log-MAP decoding algorithm, 16 decoding iterations, and variable frame size, k (BER vs. E_b/N_0).

Table 5.4: Summary of simulated performance of the UMTS turbo code in AWGN using the constant-log-MAP decoding algorithm and 16 decoding iterations. Frame sizes are in bits. Coding gains and distance from constellation-constrained capacity, C_{OOK}, are in E_b/N_0 in dB.

Frame size (Uncoded), k	**Simulated Coding Gain (dB)**	**Distance to C_{OOK} (dB)**
40	4.85	4.04
190	6.52	2.37
530	7.24	1.66
640	7.40	1.49
1060	7.59	1.30
2020	7.81	1.08
3460	7.93	0.97
5114	8.01	0.88

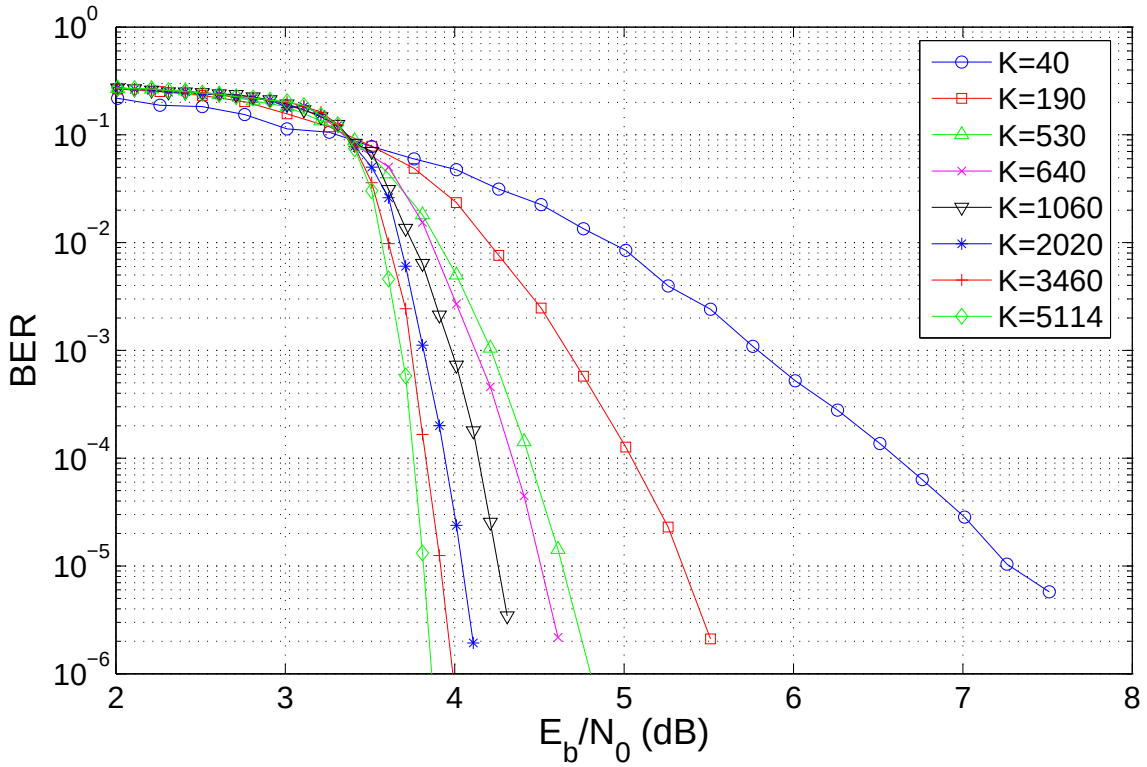


Figure 5.5: Simulated performance of the UMTS turbo code in AWGN using the max-log-MAP decoding algorithm, 16 decoding iterations, and variable frame size, k (BER vs. E_b/N_0).

Table 5.5: Summary of simulated performance of the UMTS turbo code in AWGN using the max-log-MAP decoding algorithm and 16 decoding iterations. Frame sizes are in bits. Coding gains and distance from constellation-constrained capacity, C_{OOK}, are in E_b/N_0 in dB.

Frame size (Uncoded), k	**Simulated Coding Gain (dB)**	**Distance to C_{OOK} (dB)**
40	4.77	4.12
190	6.33	2.56
530	6.93	1.96
640	7.03	1.87
1060	7.25	1.65
2020	7.44	1.45
3460	7.56	1.34
5114	7.61	1.28

Table 5.6: UMTS experimental scenarios.

Scenario	**Rate**	**Frame size (uncoded), k**	**Decoding algorithm**	**Decoding Iterations**
1	1/3	5114	log-MAP	16
2	1/3	530	max-log-MAP	10

approximations to the log-MAP algorithm. Presumably, assuming fixed processing power, this would also decrease the latency of the system as decoding could be performed faster. Alternatively, the latency of the system can be reduced by choosing a smaller frame size. Although we haven't explored it in depth in this chapter, the number of decoding iterations can also be decreased to achieve decreased system latency at the cost of power efficiency. Furthermore, since multiple decoding algorithms, frame sizes, and numbers of decoding iterations are available, it is possible to specify the degree to which we are willing to trade power efficiency for decreased latency and/or decoding complexity. This ability to fine-tune power efficiency, latency, and decoder complexity makes the UMTS turbo code a potentially attractive option for practical systems, particularly if real-time applications are used.

5.1.3 Experimental Results

We now present experimental measurements of the BER performance of the UMTS turbo code when used on the underwater optical communication channel. Based on simulated performance in the previous section, two sets of parameters, designated scenarios 1 and 2, were selected for experimentation. These parameters are summarized in Table 5.6. The parameters for scenario 1 were selected to maximize power efficiency. It uses an uncoded frame size of $k = 5114$ bits, the largest frame size specified by the UMTS standard. The decoder uses the log-MAP decoding algorithm with 16 decoding iterations. The parameters for scenario 2 were chosen as an example of a UMTS code that offers slightly lower power efficiency in exchange for lower latency and decreased decoding complexity. It uses an uncoded frame size of $k = 530$ bits, and the decoder uses the max-log-MAP decoding algorithm. To further reduce the latency of the system, only 10 decoding iterations are used. Based on simulation, this comes at a cost of less than 0.1 dB in power efficiency and in return the decoding time is reduced by roughly 38%. These aspects make this scenario a potentially attractive option for future implementation in a real-time optical system.

Table 5.7: Summary of 95% confidence intervals for UMTS packets at BER $\hat{p} = 10^{-4}$.

Scenario	**Coded System**	**Uncoded System**
1	$(2.16\hat{p},\ 0.46\hat{p})$	$(1.57\hat{p},\ 0.64\hat{p})$
2	$(2.11\hat{p},\ 0.47\hat{p})$	$(1.55\hat{p},\ 0.65\hat{p})$

Experimental results were obtained using the experimental testbed and procedure described in Section 3.2. The encoder and decoder were implemented in MATLAB using functions from the CML [55] in addition to software written by the author. Experimental packets are generated prior to experimentation. As mentioned in Section 4.2, limitations in buffer size constrain the data packet length to at most 200,000 bits, including a 1024 bit packet header. To maximize the number of information bits sent per packet, the largest multiple of the coded frame size that is below this limit is used. For scenario 1, this results in 184,248 coded bits (approx. 22.5 KB) and 61,368 information bits (approx. 7.5 KB) per packet. For scenario 2, this results in 198,648 coded bits (approx. 24 KB) and 65,720 information bits (approx. 8 KB) per packet. It is necessary to measure the performance of the system both with and without coding in order to calculate empirical coding gains. Following the same approach used in the previous chapter, a single transmitted packet is used to estimate both the coded BER and the uncoded BER, where the uncoded BER is determined by performing a hard decision on the received data and calculating the BER prior to decoding. This approach eliminates the need to send two packets for the same water condition.

The accuracy achieved using the packet sizes given above can be quantified in terms of confidence intervals. Recall from Section 4.2 that the 95% confidence interval, denoted (y_+, y_-), is defined as the interval about the BER estimate, $\hat{p}$, such that

$$\text{Prob}\left[y_+ \leq p \leq y_-\right] = 0.95, \tag{5.6}$$

where y_+ is the upper bound on the confidence interval, y_- is the lower bound on the confidence interval, and p is the actual BER. Furthermore, the bounds on the 95% confidence interval can be closely approximated using Equation (4.6). Substituting into this equation the UMTS packet sizes listed above, it is possible to calculate the 95% confidence intervals for a BER estimate of $\hat{p} = 10^{-4}$, the BER of interest for this system. These confidence intervals are summarized in Table 5.7.

At the receiver, the decoding algorithm is initialized by calculating the LLRs of the received sequence prior to decoding. Recall from Section 2.3.2 that the LLR is equal to

$$R(c_i) = \ln\left[\frac{P(y_i|c_i=1)}{P(y_i|c_i=0)}\right] \tag{5.7}$$

where $P(y_i|c_i = b)$ is the conditional probability of receiving y_i given that $c_i = b$ was transmitted. For BPSK modulation with equiprobable symbols $\{+1, -1\}$, the LLR is simply

$$R(c_i) = 2\rho y_i \tag{5.8}$$

where ρ is the SNR (cf., [15, pp. 780-781]). Based on this equation, the LLRs of the received data for the OOK modulated system can be calculated similarly as

$$R(c_i) = \rho \bar{y}_i \tag{5.9}$$

where $\bar{y}_i$ is the OOK received sequence normalized to zero mean and noise variance one, as defined in Section 3.3[1]. In this equation the factor of 2 goes away when changing from BPSK to OOK modulation due to the fact OOK modulation with an SNR ρ_{OOK} is equivalent to BPSK modulation with an SNR $\rho_{OOK}/2$.

BER Performance

The experimentally measured BERs for each scenario are plotted versus E_b/N_0 in Figures 5.6 and 5.7 along with the performance of uncoded OOK modulation. In each plot, the uncoded data are represented by blue circles and the coded data are represented by red triangles. Points plotted on the x axis have a BER$\leq 10^{-5}$. Also shown for comparison are:

- the theoretical BER for the uncoded system given by (5.3),
- the simulated BER for the coded system from Section 5.1.2,
- the constellation-constrained capacity for OOK at the appropriate code rate given by (5.4), or more precisely, this is the E_b/N_0 at which $C_{OOK} = r$.

[1]Prior to forming the LLR, the received data is normalized to match the statistics expected by the CML decoding algorithm. The received sequence (normalized to mean zero) is modeled according to the CML algorithm as $\tilde{y}_i = a\left(\bar{x}_i + \sqrt{\frac{2}{\rho}}\bar{n}_i\right)$, where $\bar{x}_i \in \{+1, -1\}$, $\bar{n}_i \sim N(0,1)$, and a is a constant scale factor. Note that this definition differs from $\bar{y}_i$. To compensate, the scale factor is estimated for each received packet by taking the variance of $\tilde{y}_i$ and solving for a. The LLR is then formed using the equation $R(c_i) = \frac{\rho}{\hat{a}}\tilde{y}_i$, where $\hat{a}$ is the estimate of a.

For each of these curves, the ideal AWGN channel is assumed. From the figures, we see that the experimental results for the coded system agree very closely with simulation. There is a small deviation from the simulated curve for scenario 2, however this deviation is within 0.5 dB with the exception of an outlier at BER$\approx$ 7e$-$4. The experimental results for the uncoded system follow close to theory but deviate slightly at high SNRs, performing less than 1 dB worse than theory at a BER of 10^{-4}. As a result, the measured coding gain is slightly better than the gains predicted by simulation. The resulting coding gains in terms of E_b/N_0 are approximately 9.0 dB for scenario 1 and 6.8 dB for scenario 2. These results are summarized in column 2 of Table 5.8. The performance of these codes in terms of code rate vs. power efficiency in E_b/N_0 is summarized visually in Figure 5.8 which shows both coded and uncoded rates vs. measured E_b/N_0 at a BER of 10^{-4} and their relation to the constellation-constrained capacity.

The fact that the experimental data agrees so closely with theory and simulation supports the AWGN channel model for the underwater optical communications link. In particular, since the decoding of a turbo code requires knowledge of channel statistics to form the received symbol LLRs, the fact that the experimental coded data agrees so closely with simulation is a strong indicator that the model is reasonably accurate. However, the systematic deviation of the uncoded data from the theory curve at high SNR suggests a deviation from the AWGN channel model. As mentioned in the previous chapter, such a deviation could be caused by a number of factors. It could be due to non-Gaussian sources of noise or the presence of weak fading due to turbulence. It could also be caused by the experimental setup itself, possibly due to quantization error at high signal levels or imprecise synchronization. Based on these experimental results, the cause is yet unclear but is beyond the scope of this thesis. Investigating the behavior of the uncoded system at high SNR is an interesting subject for future work.

Range Extension

In the previous section, we saw that the UMTS turbo code can provide significant gains in power efficiency for an underwater optical communication link. We now examine how these improvements in power efficiency can extend the range of reliable underwater communication. Following the approach described in Section 4.2.2, the link distance expressed in units of attenuation lengths, $d_a = c(\lambda)d$, is used as a water metric to quantify

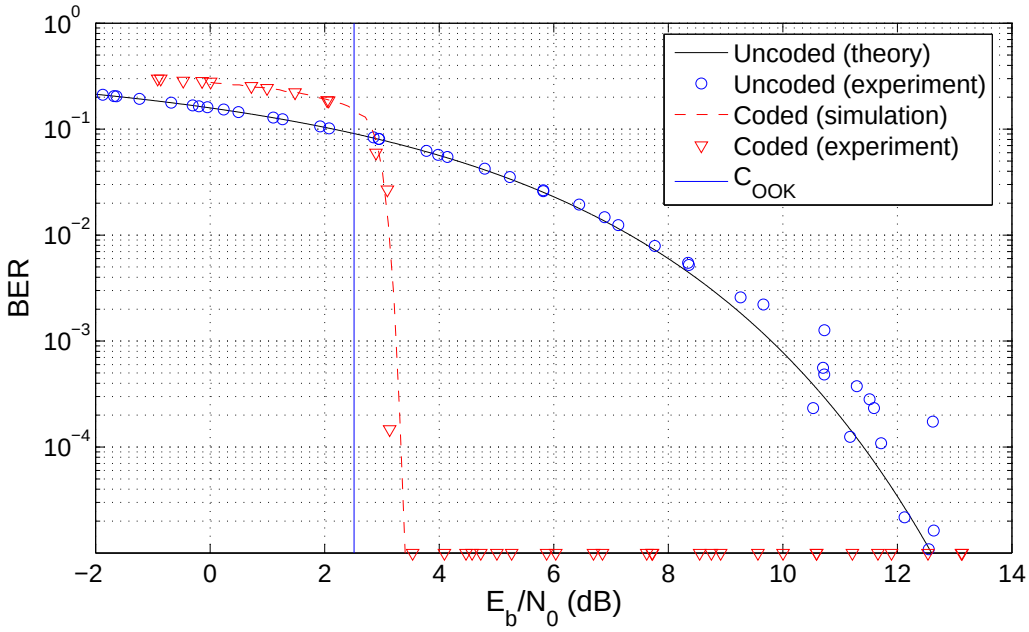


Figure 5.6: Experimental BER vs. E_b/N_0 results for UMTS scenario 1: $k = 5114$, $r = 1/3$, log-MAP decoding, 16 decoding iterations.

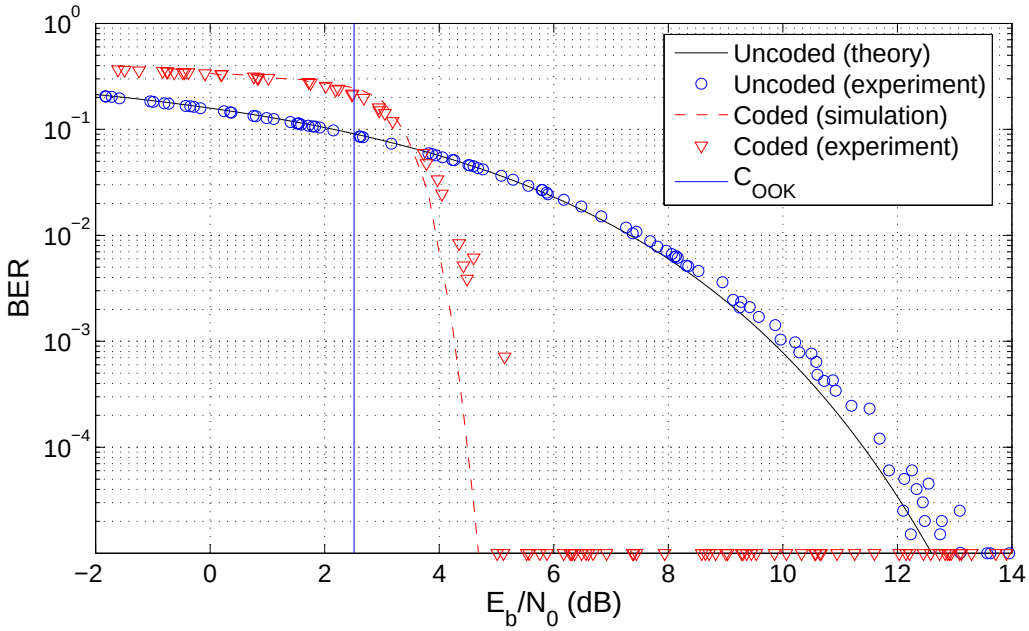


Figure 5.7: Experimental BER vs. E_b/N_0 results for UMTS scenario 2: $k = 530$, $r = 1/3$, max-log-MAP decoding, 10 decoding iterations.

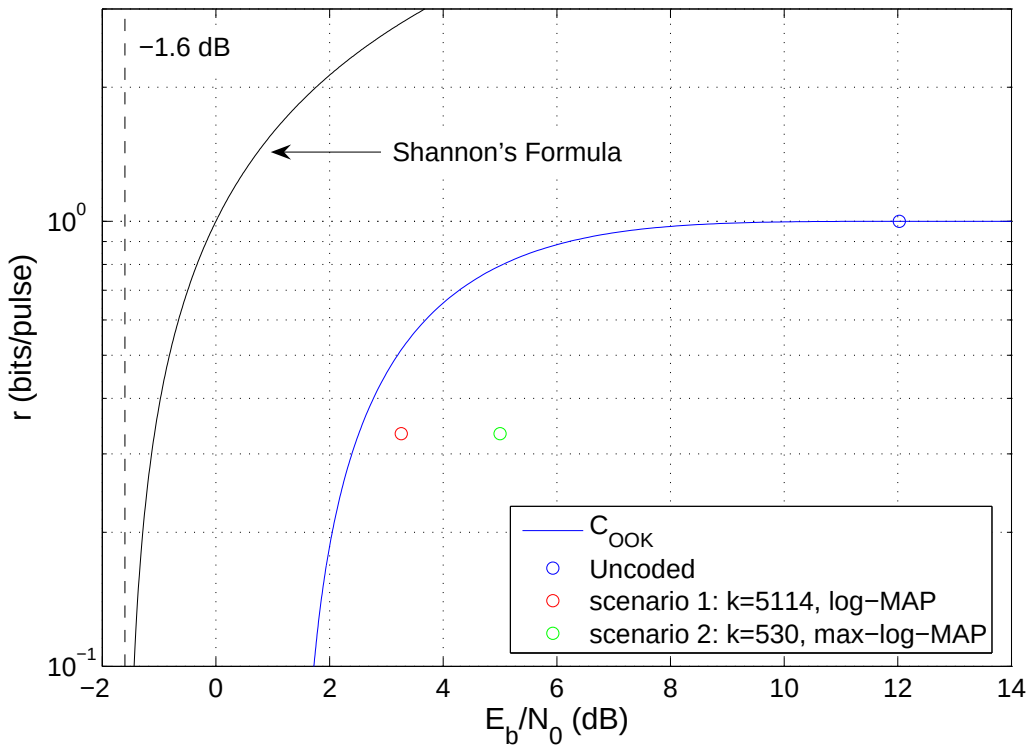


Figure 5.8: Comparison of experimental performance of the UMTS turbo code with information-theoretic limits.

Table 5.8: Summary of experimental FEC coding gains for UMTS turbo code.

Scenario	**Gain in E_b/N_0 (dB)**	**Expected gain in $d_a = c(405\ \text{nm})d$**	**Measured gain in $d_a = c(530\ \text{nm})d$**
1	9.0	1.24	1.15
2	6.8	0.99	0.89

the range extension of a coded system. This approach has the advantage that the coding gains can easily be extrapolated to various distances and water turbidities.

Expected coding gains in d_a are calculated for each scenario based on observed coding gains in SNR from the previous section. This is done using the relationship given by (3.35) (cf., Section 3.5). These expected gains are shown in column 2 of Table 5.8. We once again note, however, that it is implied in (3.35) that the expected gains are in terms of $d_a = c(405 \text{ nm})d$, i.e. the range extension in attenuation lengths is evaluated at the wavelength of the experimental system, $\lambda = 405$ nm. These values will differ from the empirical gains presented below, which are based on measurements obtained with a transmissometer operating at 530 nm.

Experimental results for BER vs. d_a are shown in Figures 5.9 and 5.10 for scenarios 1 and 2, respectively. Note that these values of d_a are based on measured values of $c(530 \text{ nm})$ from the transmissometer. The coding gains in terms of attenuation lengths are approximately 1.15 for scenario 1 and 0.89 for scenario 2. These results are summarized in column 3 of Table 5.8. Theory and simulation curves as a function of $c(530 \text{ nm})$ were calculated as follows. For each set of experimental data, the measured $c(530 \text{ nm})$ values from the transmissometer were first plotted as a function of estimated SNR, as shown in Appendix A. These plots demonstrate a linear relationship between ρ in dB and $c(530 \text{ nm})$. A linear, least-square regression was used to determine a line of best fit for each experiment. We observe that the slopes of these lines are approximately the same, however the y-intercept varies by experiment. Using these lines, it is possible to map SNR from the theory curves onto d_a for each experiment.

Using the experimental gains in d_a, we can extrapolate the potential range extension that could be attained using these FEC codes. Following the method established in Section 4.2.2, we examine the range extension achieved by each UMTS code scenario in three different water conditions: clear ocean water ($c(\lambda) = 0.1514 \text{ m}^{-1}$), coastal water ($c(\lambda) = 0.399 \text{ m}^{-1}$), and turbid harbor water ($c(\lambda) = 2.195 \text{ m}^{-1}$). These attenuation coefficient values are based on measurements in [46] that were made in a spectral band centered at $\lambda \approx 530$ nm. Applying the relationship

$$d = \frac{d_a}{c(530 \text{ nm})} \tag{5.10}$$

and assuming that Beer's Law holds at each value of $c(\lambda)$, we are able to predict the

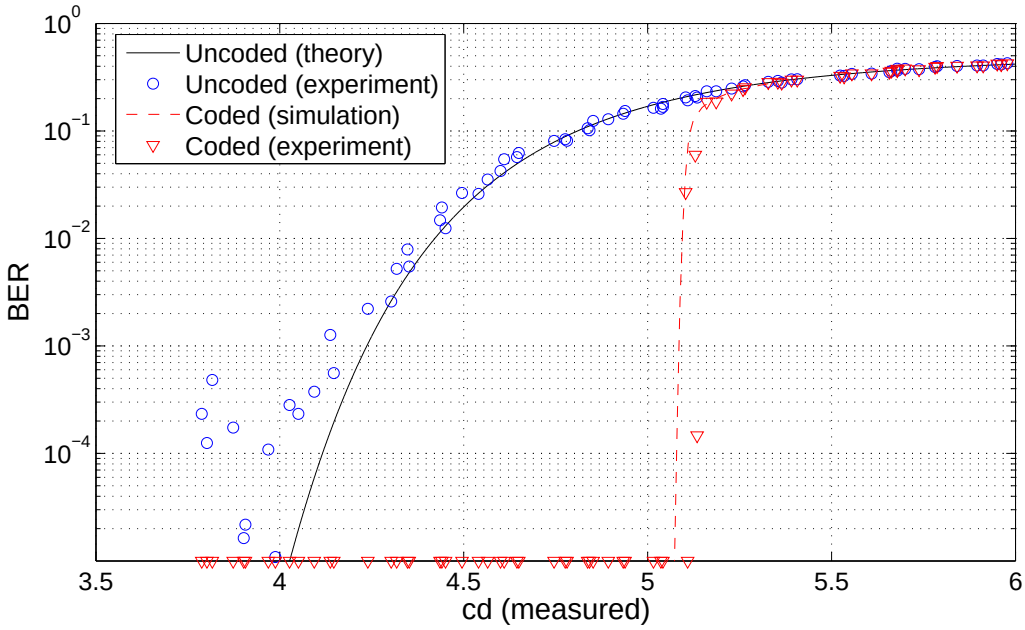


Figure 5.9: Experimental BER vs. d_a results for UMTS scenario 1: $k = 5114$, $r = 1/3$, log-MAP decoding, 16 decoding iterations.

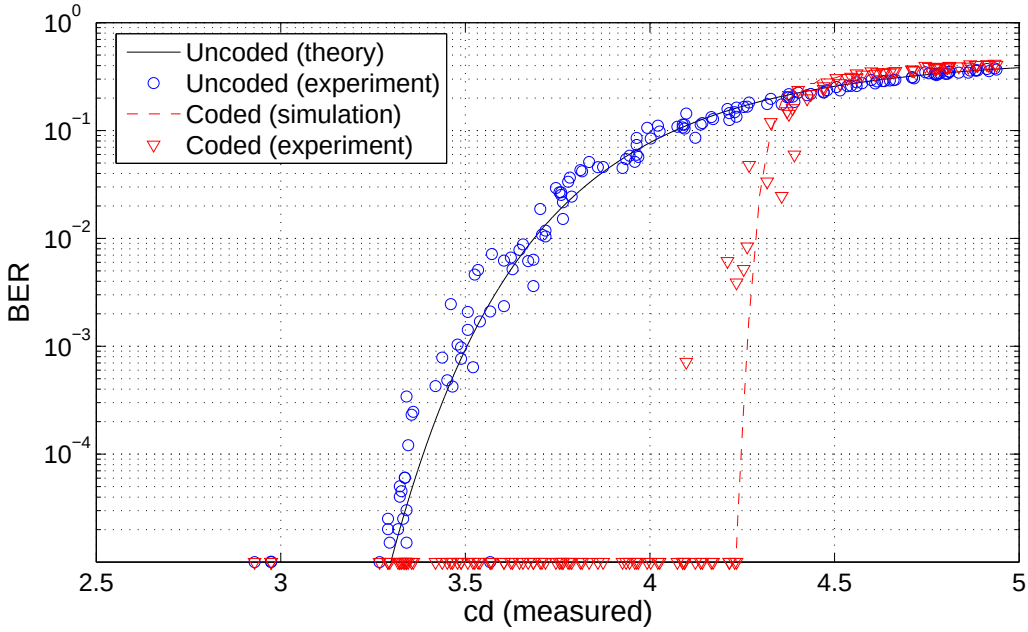


Figure 5.10: Experimental BER vs. d_a results for UMTS scenario 2: $k = 530$, $r = 1/3$, max-log-MAP decoding, 10 decoding iterations.

Table 5.9: Summary of range extension for the UMTS turbo codes.

Water Type	$c(\lambda)$	**Range Extension (m)**	
		Scenario 1	**Scenario 2**
Clear ocean	0.1514	7.57	5.89
Coastal ocean	0.399	2.87	2.24
Harbor water	2.195	0.52	0.41

range extension achieved by each code scenario in each type of water. These results are summarized in Table 5.9. Again, we note that since the measured attenuation coefficients from the transmissometer were used, the measured coding gains in d_a are likely to be lower than the actual gains of the system at $\lambda = 405$ nm. Thus, the range extension capabilities presented are conservative.

5.2 CCSDS Turbo Code

5.2.1 CCSDS Turbo Code Specification

The second turbo code we consider is the Consultative Committee for Space Data Systems (CCSDS) turbo code used in space telemetry [56], [57]. The CCSDS turbo encoder is comprised of two identical constraint length 5 RSC encoders concatenated in parallel. Each RSC encoder has one feedback path and three feed-forward paths. The feedback generator is $\mathbf{g}^{(0)} = 23$ and the feed-forward generators are $\mathbf{g}^{(1)} = 33$, $\mathbf{g}^{(2)} = 25$, and $\mathbf{g}^{(3)} = 37$, all in octal. However, the second feed-forward path corresponding to $\mathbf{g}^{(2)}$ of the lower constituent encoder is not used.

A diagram of the CCSDS turbo encoder is shown in Figure 5.11. Initially the two switches are in the up position as the information bits, u_i, are read into the encoder. The output from the encoder is multiplexed and punctured according to one of four patterns to achieve a nominal code rate of $r = 1/2, r = 1/3, r = 1/4$ or $r = 1/6$.

The rate of 1/3 is achieved by transmitting the parity bits form feed-forward generator $\mathbf{g}^{(1)}$ of each constituent encoder, denoted by $z_{(1),i}$ and $z'_{(1),i}$. The output pattern is thus

$$\left\{u_1, z_{(1),1}, z'_{(1),1}, u_2, z_{(1),2}, z'_{(1),2}, \cdots, u_k, z_{(1),k}, z'_{(1),k}\right\} , \tag{5.11}$$

where $\mathbf{u} = \{u_1, u_2, \cdots, u_k\}$ represents the original data sequence. For rate 1/2, odd-even

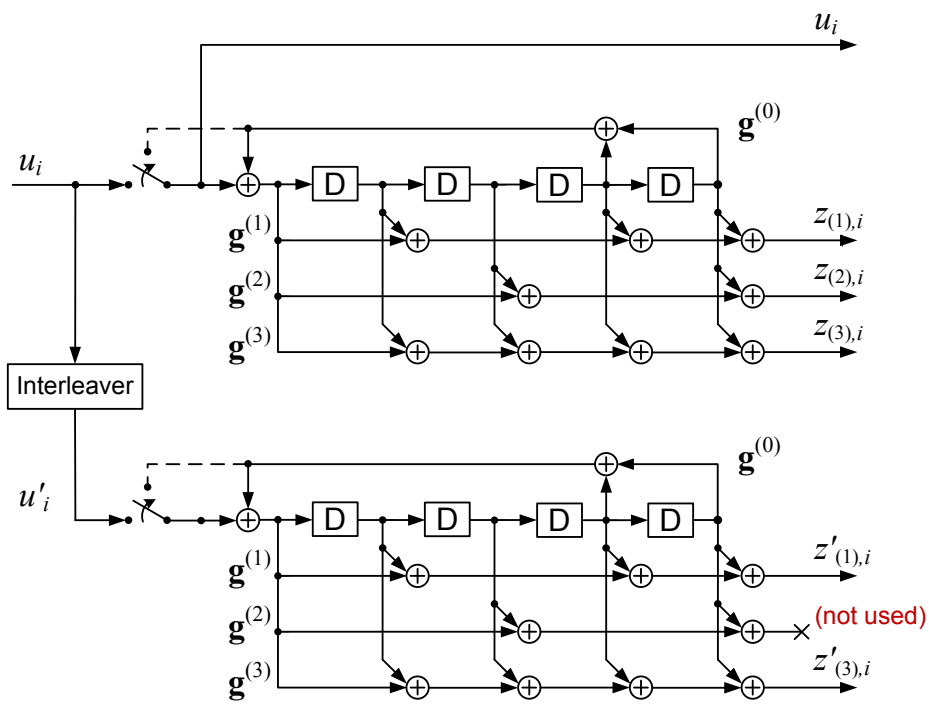


Figure 5.11: Turbo encoder for the CCSDS turbo code. For clarity, the generators $\mathbf{g}^{(0)}$ through $\mathbf{g}^{(3)}$ are labeled. Adapted from [56].

puncturing is applied to the parity bits from the same two generators, resulting in the output pattern

$$\left\{u_1, z_{(1),1}, u_2, z'_{(1),2}, u_3, z_{(1),3}, \cdots, u_k, z'_{(1),k}\right\} . \tag{5.12}$$

For rate 1/4, the parity bits from $\mathbf{g}^{(2)}$ and $\mathbf{g}^{(3)}$ of the upper constituent encoder, denoted by $z_{(2),i}$ and $z_{(3),i}$, and the parity bit from $\mathbf{g}^{(1)}$ of the lower encoder are used. This results in the output pattern

$$\left\{u_1, z_{(2),1}, z_{(3),1}, z'_{(1),1}, u_2, z_{(2),2}, z_{(3),2}, z'_{(1),2}, \cdots, u_k, z_{(2),k}, z_{(3),k}, z'_{(1),k}\right\} . \tag{5.13}$$

No puncturing is used for rate 1/4. For rate 1/6, all three parity bits from the upper constituent encoder are used along with the parity bits from $\mathbf{g}^{(1)}$ and $\mathbf{g}^{(3)}$ of the lower encoder. This results in the output pattern

$$\left\{u_1, z_{(1),1}, z_{(2),1}, z_{(3),1}, z'_{(1),1}, z'_{(3),1}, \cdots, u_k, z_{(1),k}, z_{(2),k}, z_{(3),k}, z'_{(1),k}, z'_{(3),k}\ ,\right\} \tag{5.14}$$

where $z'_{(3),i}$ denotes parity bits from $\mathbf{g}^{(3)}$ of the lower encoder. No puncturing is used for rate 1/6.

Table 5.10: CCSDS turbo code frame sizes.

Information bits, k	**Coded bits,** n			
	$r = 1/2$	$r = 1/3$	$r = 1/4$	$r = 1/6$
1784	3576	5364	7152	10728
3568	7144	10716	14288	21432
7136	14280	21420	28560	42840
8920	17848	26772	35696	53544

Once the entire frame has entered the shift register, the switches move to the upper position for the final 4 bit times to terminate the trellis and return the shift register to the all-zero state. During trellis termination, the encoder continues to output non-zero encoded symbols. These symbols are multiplexed and transmitted using the same output patterns described above for each respective rate. Counting tail bits, each frame of k information bits produces $n = (k + 4)/r$ coded bits, where r is the nominal code rate. The CCSDS turbo code supports four frame sizes, which are summarized in Table 5.10. A fifth, larger frame size of $k = 16,384$ information bits is currently under study and will be incorporated into a future revision of the standard. This larger frame size should afford larger coding gains and may be a topic of interest for future work.

The CCSDS turbo interleaver operates on the entire frame of k information bits according to the following algorithm. First, k is expressed as the product of two variables, k_1 and k_2, which are given in Table 5.11. Next, the permutation numbers $\pi(s)$ are calculated for $s = 1, \cdots, k$ using the following formulas:

$$m = (s-1) \bmod 2 \tag{5.15}$$

$$i = \left\lfloor \frac{s-1}{2k_2} \right\rfloor \tag{5.16}$$

$$j = \left\lfloor \frac{s-1}{2} \right\rfloor - ik_2 \tag{5.17}$$

$$t = (19i + 1) \bmod \frac{k_1}{2} \tag{5.18}$$

$$q = (t \bmod 8) + 1 \tag{5.19}$$

$$c = (jp_q + 21m) \bmod k_2 \tag{5.20}$$

$$\pi(s) = 2\left(t + c\frac{k_1}{k_2} + 1\right) - m \tag{5.21}$$

where $\lfloor x \rfloor$ denotes the largest integer less than or equal to x, and p_q denotes one of the

Table 5.11: CCSDS turbo interleaver variables k_1 and k_2 by frame size.

Information bits, k	k_1	k_2
1784	8	223
3568	8	223×2
7136	8	223×4
8920	8	223×5

following eight prime integers:

$$p_1 = 31, \quad p_2 = 37, \quad p_3 = 43, \quad p_4 = 47, \quad p_5 = 53, \quad p_6 = 59, \quad p_7 = 61, \quad p_8 = 67.$$

Finally, the information sequence is reordered according to the permutation numbers. This is done by moving the bit corresponding to index $\pi(s)$ in the uninterleaved sequence, $\mathbf{u}$, to index s in the interleaved sequence, $\mathbf{u}'$. This mapping between the uninterleaved sequence and the interleaved sequence is shown in Figure 5.12.

Decoding is performed in the log domain using the turbo decoder described in Section 2.3.2. The SISO processors decode using the log-MAP algorithm or one of its approximations, i.e., linear-log-MAP, constant-log-MAP, max-log-MAP [26]. The maximum number of decoding iterations is not explicitly specified in the CCSDS standard, leaving this parameter as another design decision.

5.2.2 Simulation Results

To provide a theoretical baseline for the underwater experiments, the performance of the CCSDS turbo code for the AWGN channel was simulated. Simulations were run in MATLAB using the Coded Modulation Library (CML) software developed by Iterative Solutions [55]. Simulations were run for each of the four block sizes (i.e., $k = 1784$, $k = 3568$, $k = 7136$, and $k = 8920$) and each of the four rates (i.e., $r = 1/2$, $r = 1/3$, $r = 1/4$, and $r = 1/6$) specified by the standard. In order to compare the potential trade-offs between power efficiency and decoder complexity, each of the four decoding algorithms described in Section 2.3.2 (i.e., the log-domain BCJR algorithm and its three approximations) were simulated. Simulation results are shown in Figures 5.13 through 5.16 and are presented in order of decoder complexity as follows: log-MAP, linear-log-MAP, constant-log-MAP, and max-log-MAP. Based on the channel model developed in Section 3.3, the bit-error rate

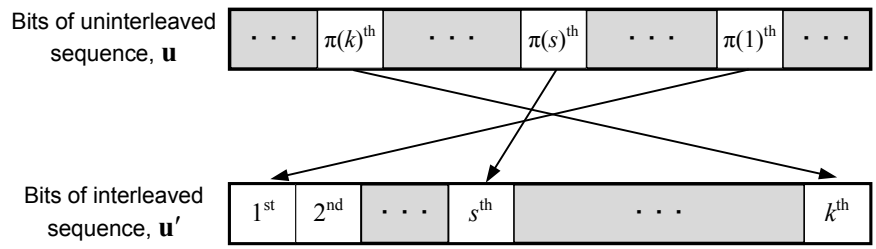


Figure 5.12: CCSDS interleaver mapping of bits in the uninterleaved sequence to bits in the interleaved sequence using the permutation numbers, $\pi(s)$. Adapted from [56].

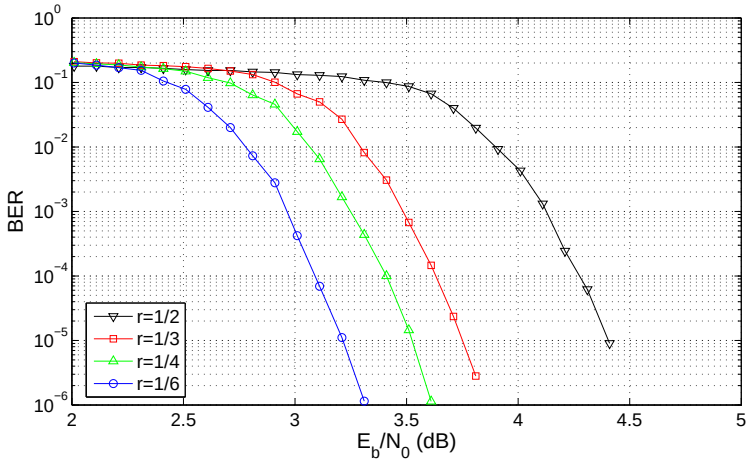
(BER) for OOK modulation in AWGN is given by

$$P_b = Q\left(\sqrt{\frac{E_b}{N_0}}\right), \tag{5.22}$$

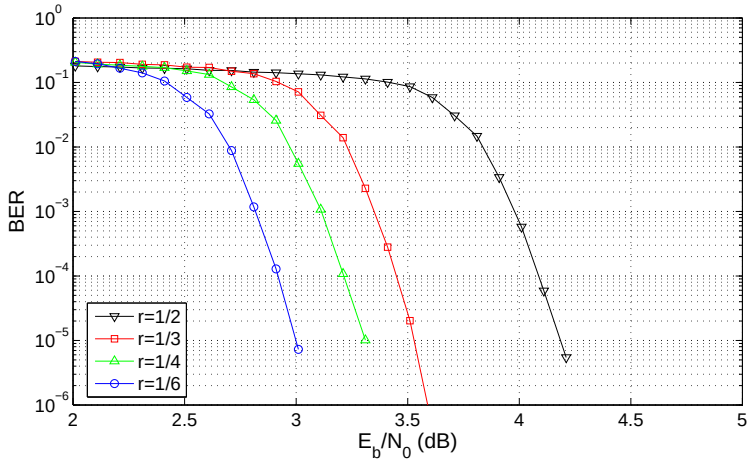
where E_b/N_0 is the bit-energy-to-noise-density-ratio, and Q is the q-function (cf., [51, p. 40]). Uncoded OOK achieves a BER of 10^{-4} (the regime of interest for this system) at an E_b/N_0 of 11.41 dB. The coding gain in E_b/N_0 that can potentially be obtained at this BER varies by rate, frame size and decoding algorithm, ranging from 8.62 dB for log-MAP decoding with rate $r = 1/6$ and frame size $k = 5114$ to 6.68 dB for max-log-MAP decoding with rate $r = 1/2$ and frame size $k = 40$. These gains are summarized in Tables 5.12 through 5.15. Like the UMTS turbo code discussed in Section 5.1, we observe that in general the potential coding gains tend to decrease with decreasing decoder complexity.

The constellation-constrained capacity is used to compare the simulated performance of the coded system with the best possible performance dictated by information theory, as we did for the UMTS turbo code in Section 5.1.2. Like the UMTS turbo code, the CCSDS turbo code is a state-of-the-art, capacity-approaching code that is designed to operate at an E_b/N_0 very close to the information-theoretic limits. The distance of each code at a BER of 10^{-4} from C_{OOK} is summarized in Tables 5.15 through 5.15. We note that for large frame sizes, the simulated performance is within 1 dB of the constellation-constrained capacity at a BER of 10^{-4}.

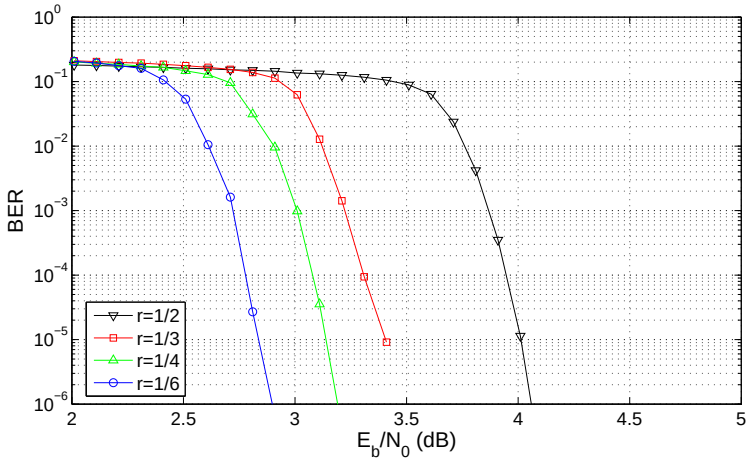
Like the UMTS turbo code, the CCSDS turbo codes presented here offer a number of trade-offs between power efficiency, latency, and decoder complexity. Although they do not offer the same flexibility in terms of frame sizes, the option to choose between different code rates is appealing because it provides an addition trade-off between power efficiency



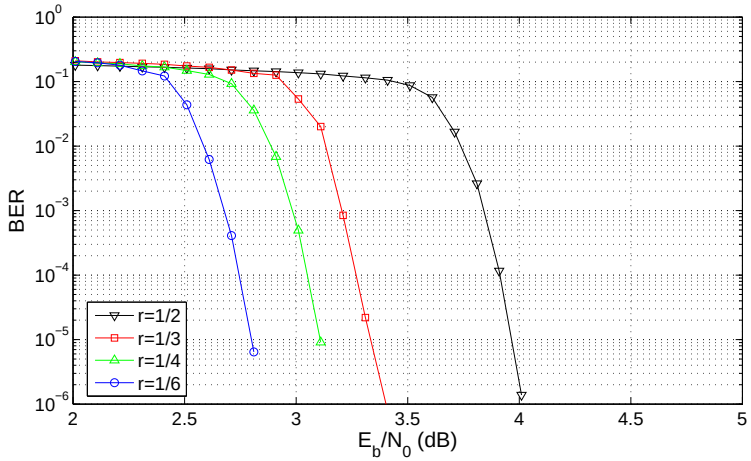
(a) $k = 1784$



(b) $k = 3568$

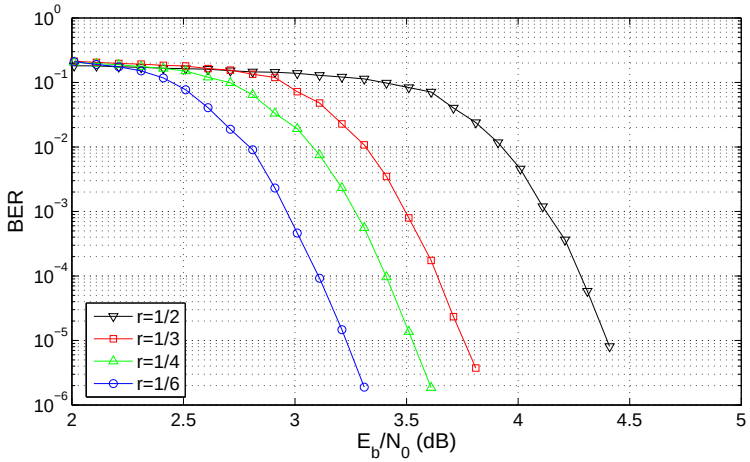


(c) $k = 7136$

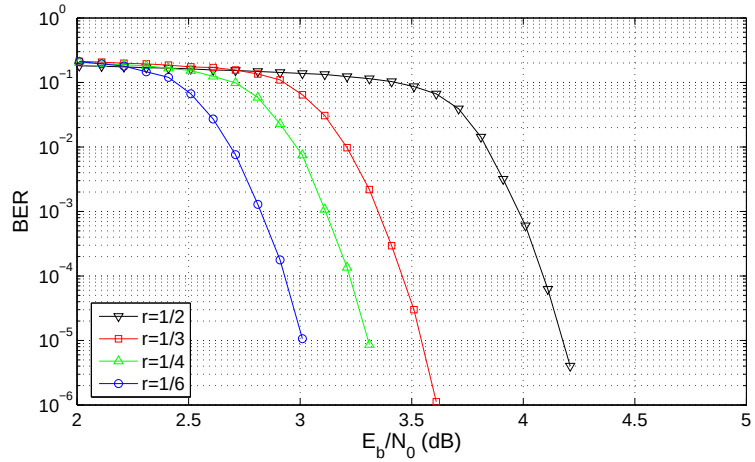


(d) $k = 8920$

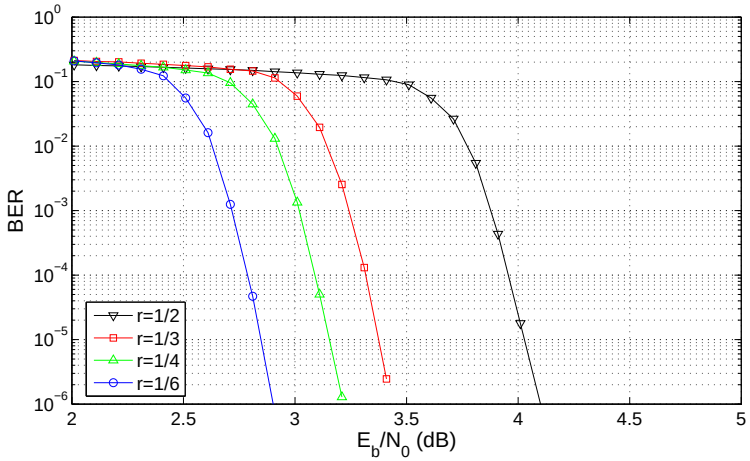
Figure 5.13: Simulation results for the CCSDS turbo code using the log-MAP decoding algorithm, and a frame size of (a) $k = 1784$, (b) $k = 3568$, (c) $k = 7136$, and (d) $k = 8920$.



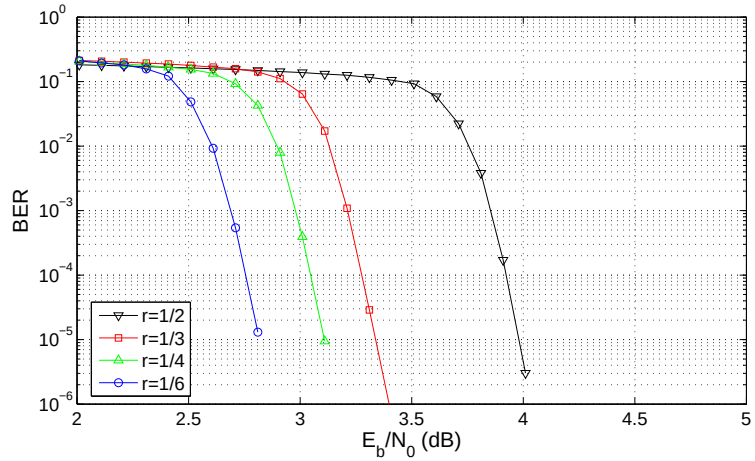
(a) $k = 1784$



(b) $k = 3568$

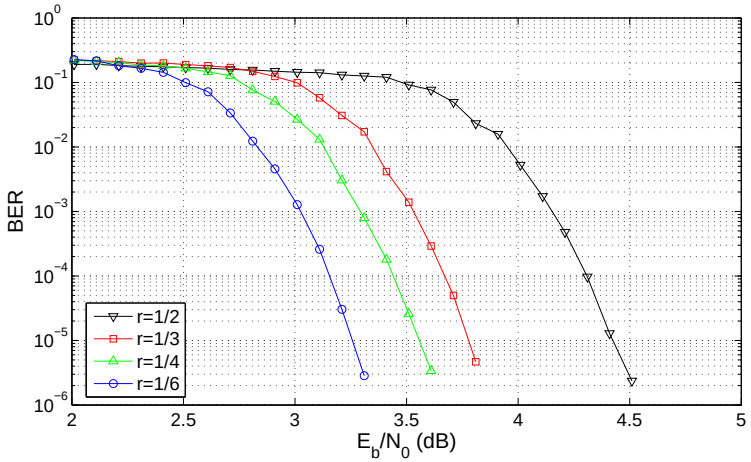


(c) $k = 7136$

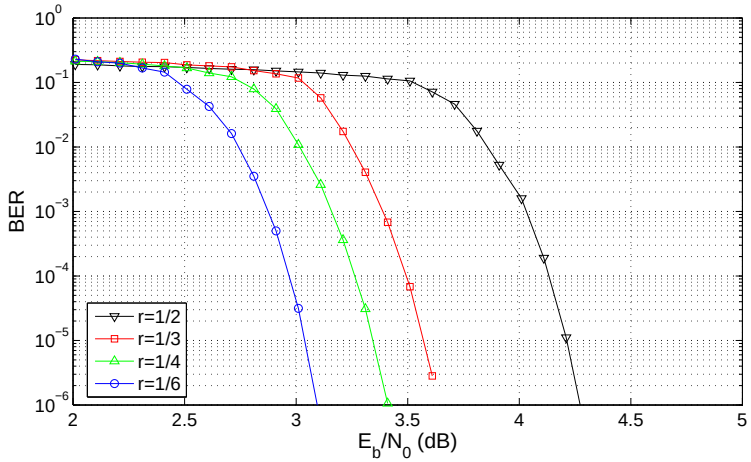


(d) $k = 8920$

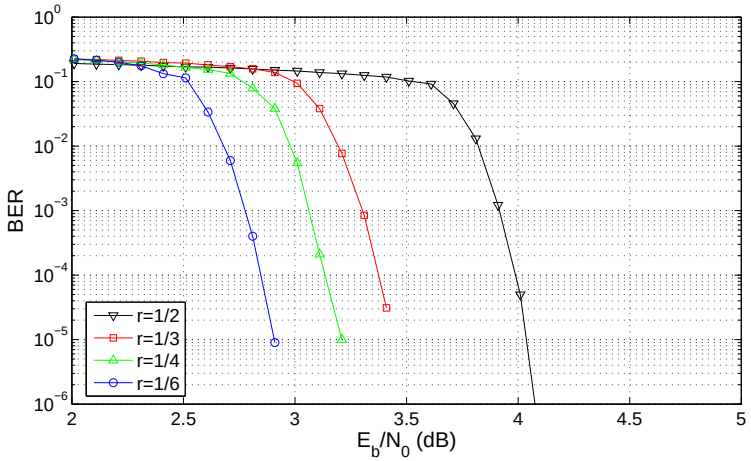
Figure 5.14: Simulation results for the CCSDS turbo code using the linear-log-MAP decoding algorithm, and a frame size of (a) $k = 1784$, (b) $k = 3568$, (c) $k = 7136$, and (d) $k = 8920$.



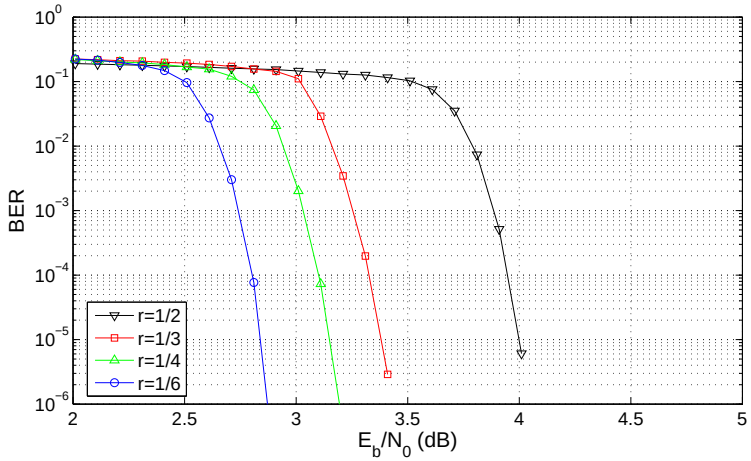
(a) $k = 1784$



(b) $k = 3568$

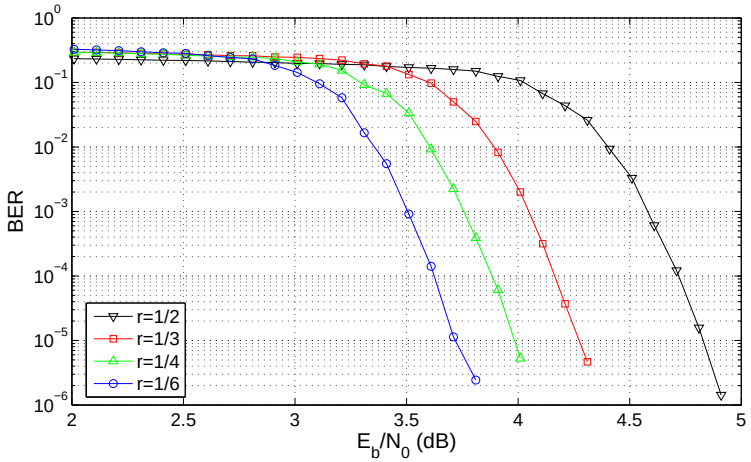


(c) $k = 7136$

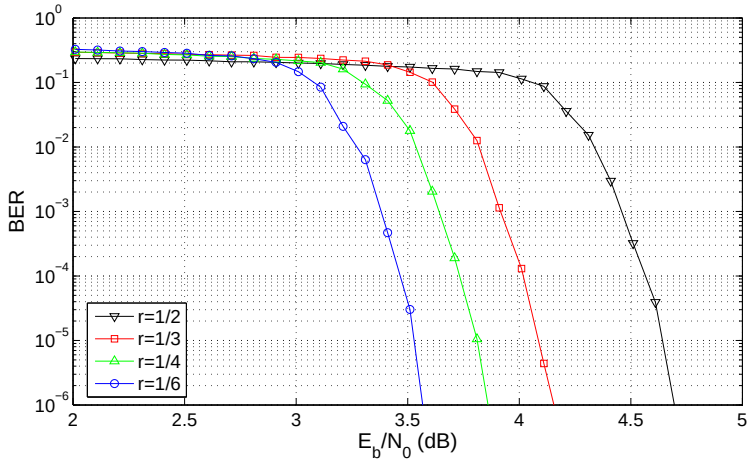


(d) $k = 8920$

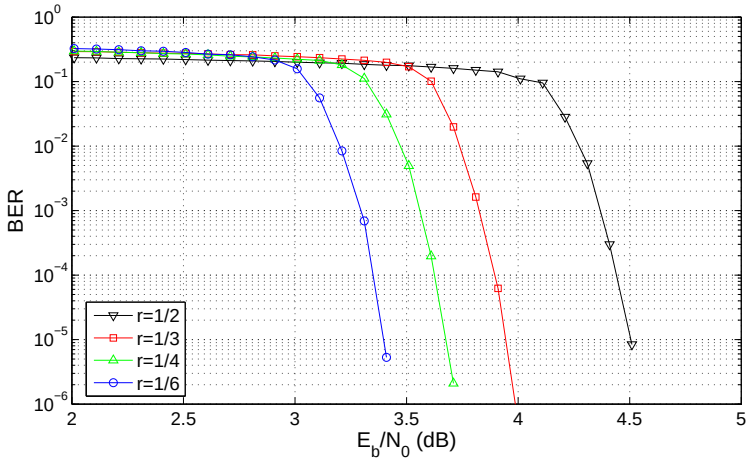
Figure 5.15: Simulation results for the CCSDS turbo code using the constant-log-MAP decoding algorithm, and a frame size of (a) $k = 1784$, (b) $k = 3568$, (c) $k = 7136$, and (d) $k = 8920$.



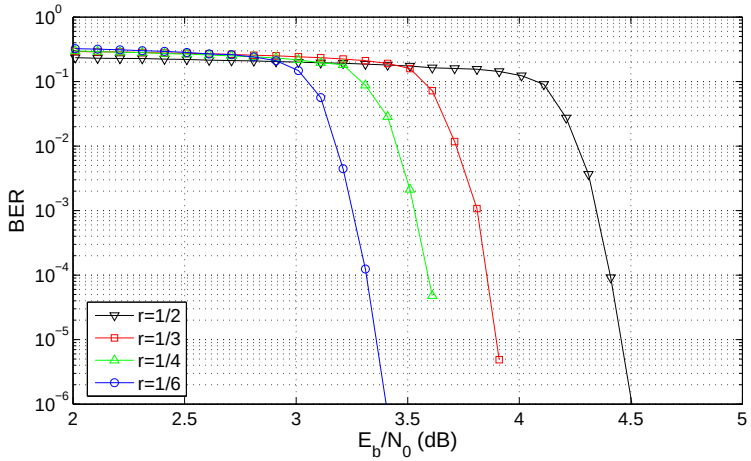
(a) $k = 1784$



(b) $k = 3568$



(c) $k = 7136$



(d) $k = 8920$

Figure 5.16: Simulation results for the CCSDS turbo code using the max-log-MAP decoding algorithm, and a frame size of (a) $k = 1784$, (b) $k = 3568$, (c) $k = 7136$, and (d) $k = 8920$.

Table 5.12: Summary of simulated performance of the CCSDS turbo code in AWGN using the log-MAP decoding algorithm and 16 decoding iterations. Frame sizes are in bits. Coding gains and distance from constellation-constrained capacity, C_{OOK}, are in E_b/N_0 in dB.

Frame Size (Uncoded), k	**Rate, r**	**Simulated Coding Gain (dB)**	**Distance to C_{OOK} (dB)**
1784	1/2	7.12	1.09
1784	1/3	7.76	1.13
1784	1/4	8.00	1.19
1784	1/6	8.31	1.16
3568	1/2	7.31	0.90
3568	1/3	7.93	0.96
3568	1/4	8.19	1.00
3568	1/6	8.47	1.00
7136	1/2	7.42	0.79
7136	1/3	8.10	0.79
7136	1/4	8.31	0.89
7136	1/6	8.60	0.87
8920	1/2	7.48	0.73
8920	1/3	8.11	0.79
8920	1/4	8.32	0.88
8920	1/6	8.62	0.85

Table 5.13: Summary of simulated performance of the CCSDS turbo code in AWGN using the linear-log-MAP decoding algorithm and 16 decoding iterations. Frame sizes are in bits. Coding gains and distance from constellation-constrained capacity, C_{OOK}, are in E_b/N_0 in dB.

Frame Size (Uncoded), k	**Rate, r**	**Simulated Coding Gain (dB)**	**Distance to C_{OOK} (dB)**
1784	1/2	7.11	1.10
1784	1/3	7.75	1.14
1784	1/4	8.00	1.19
1784	1/6	8.30	1.17
3568	1/2	7.31	0.91
3568	1/3	7.92	0.97
3568	1/4	8.17	1.02
3568	1/6	8.45	1.02
7136	1/2	7.42	0.79
7136	1/3	8.07	0.82
7136	1/4	8.30	0.89
7136	1/6	8.60	0.87
8920	1/2	7.46	0.76
8920	1/3	8.11	0.79
8920	1/4	8.32	0.87
8920	1/6	8.61	0.86

Table 5.14: Summary of simulated performance of the CCSDS turbo code in AWGN using the constant-log-MAP decoding algorithm and 16 decoding iterations. Frame sizes are in bits. Coding gains and distance from constellation-constrained capacity, C_{OOK}, are in E_b/N_0 in dB.

Frame Size (Uncoded), k	**Rate, r**	**Simulated Coding Gain (dB)**	**Distance to C_{OOK} (dB)**
1784	1/2	7.10	1.11
1784	1/3	7.72	1.17
1784	1/4	7.95	1.25
1784	1/6	8.23	1.24
3568	1/2	7.25	0.96
3568	1/3	7.90	0.99
3568	1/4	8.12	1.07
3568	1/6	8.41	1.06
7136	1/2	7.40	0.81
7136	1/3	8.01	0.89
7136	1/4	8.24	0.95
7136	1/6	8.52	0.95
8920	1/2	7.42	0.79
8920	1/3	8.05	0.85
8920	1/4	8.30	0.89
8920	1/6	8.60	0.87

Table 5.15: Summary of simulated performance of the CCSDS turbo code in AWGN using the max-log-MAP decoding algorithm and 16 decoding iterations. Frame sizes are in bits. Coding gains and distance from constellation-constrained capacity, C_{OOK}, are in E_b/N_0 in dB.

Frame Size (Uncoded), k	**Rate, r**	**Simulated Coding Gain (dB)**	**Distance to C_{OOK} (dB)**
1784	1/2	6.68	1.53
1784	1/3	7.22	1.67
1784	1/4	7.51	1.68
1784	1/6	7.77	1.70
3568	1/2	6.82	1.39
3568	1/3	7.37	1.52
3568	1/4	7.65	1.54
3568	1/6	7.91	1.56
7136	1/2	6.93	1.28
7136	1/3	7.50	1.39
7136	1/4	7.75	1.44
7136	1/6	8.01	1.46
8920	1/2	7.00	1.21
8920	1/3	7.51	1.38
8920	1/4	7.80	1.39
8920	1/6	8.08	1.39

Table 5.16: CCSDS experimental scenarios.

Scenario	**Rate**	**Frame size (uncoded), k**	**Decoding algorithm**	**Decoding Iterations**
1	1/2	8920	log-MAP	16
2	1/3	8920	log-MAP	16
3	1/4	8920	log-MAP	16
4	1/6	8920	log-MAP	16

and throughput. The codes that utilize smaller code rates, larger block sizes, and higher complexity decoding algorithms, such as log-MAP decoding, tend to perform better in terms of power efficiency. However, they incur a cost in terms of throughput, latency, and decoder complexity. The ability to fine-tune code parameters to achieve this wide range of performance metrics makes the CCSDS turbo code a robust, flexible option for real-time underwater optical communication links optimized to a wide range of applications.

5.2.3 Experimental Results

We now present experimental measurements of the BER performance of the CCSDS turbo code when used on the underwater optical communication channel. Based on simulated performance in the previous section, four sets of parameters, designated scenarios 1 - 4, were selected for experimentation. The parameters used in each scenario are summarized in Table 5.16. The parameters for each scenario have been chosen to demonstrate the largest achievable coding gains at each code rate. They all use an uncoded frame size of $k = 8920$ bits, the largest frame size specified by the CCSDS standard. The decoders each use the log-MAP decoding algorithm with 16 decoding iterations. Each CCSDS code rate (i.e., $r = 1/2$, $r = 1/3$, $r = 1/4$, $r = 1/6$) is implemented. Code rates are in descending order by scenario, beginning with $r = 1/2$ for scenario 1 and ending with $r = 1/6$ for scenario 4.

Experimental results were once again obtained using the experimental testbed and procedure described in Section 3.2. The encoder and decoder were implemented in MATLAB using functions from the CML [55] in addition to software written by the author. Experimental packets are generated prior to experimentation. Following the same approach used for UMTS, a single transmitted packet is used to generate an estimate of the BERs for both the coded and uncoded systems. At the receiver, the decoding algorithm is initialized by calculating LLRs of the received sequence prior to decoding as described in Section 5.1.3.

Table 5.17: Summary of 95% confidence intervals for CCSDS packets at BER $\hat{p} = 10^{-4}$.

Scenario	**Coded System**	**Uncoded System**
1	$(1.85\hat{p},\ 0.54\hat{p})$	$(1.55\hat{p},\ 0.64\hat{p})$
2	$(2.15\hat{p},\ 0.47\hat{p})$	$(1.57\hat{p},\ 0.64\hat{p})$
3	$(2.45\hat{p},\ 0.41\hat{p})$	$(1.58\hat{p},\ 0.63\hat{p})$
4	$(3.11\hat{p},\ 0.32\hat{p})$	$(1.62\hat{p},\ 0.62\hat{p})$

As mentioned previously in this chapter, the data packet length is limited to at most 200,000 bits, including a 1024 bit packet header, due to limitations in buffer size. To maximize the number of information bits sent per packet, the largest multiple of the coded frame size that is below this limit is used. For scenario 1, this results in $196,328$ coded bits (approx. 24 KB) and $98,120$ information bits (approx. 12 KB) per packet. For scenario 2, this results in $187,404$ coded bits (approx. 23 KB) and $62,440$ information bits (approx. 8 KB) per packet. For scenario 3, this results in $178,480$ coded bits (approx. 22 KB) and $44,600$ information bits (approx. 5.5 KB) per packet. For scenario 4, this results in $160,632$ coded bits (approx. 19.5 KB) and $26,760$ information bits (approx. 3.5 KB) per packet. The accuracy achieved using these packet sizes is quantified in terms of confidence intervals. Following the same method described in Section 5.1.3, the 95% confidence intervals are calculated for the CCSDS data packet sizes at a BER estimate of $\hat{p} = 10^{-4}$. These confidence intervals are summarized in Table 5.17

BER Performance

The experimentally measured BERs for each scenario are plotted versus E_b/N_0 in Figures 5.17 through 5.20 along with the performance of uncoded OOK modulation. In each plot, the uncoded data are represented by blue circles and the coded data are represented by red triangles. Points plotted on the x axis have a BER$\leq 10^{-5}$. Also shown for comparison are:

- the theoretical BER for the uncoded system given by (5.22),
- the simulated BER for the coded system from Section 5.2.2,
- the constellation-constrained capacity for OOK at the appropriate code rate given by (5.4), or more precisely, this is the E_b/N_0 at which $C_{OOK} = r$.

For each of these curves, the ideal AWGN channel is assumed. From the figures, we see that the experimental results for the coded systems agree reasonably well with simulation. We observe that there is a deviation from the simulated curve for scenarios 1 - 3. The deviation of the experimental data is within 0.5 dB for scenarios 1 and 3 and within 0.6 dB for scenario 2. The experimental results for the uncoded system follow closely with theory but deviate slightly at high SNRs, performing about 1 dB worse than theory at a BER of 10^{-4}. As a result, the measured coding gain is slightly better than the gains predicted by simulation. The resulting coding gains in terms of E_b/N_0 are approximately 7.7 dB for scenario 1, 8.1 dB for scenario 2, 8.3 dB for scenario 3, and 9.5 dB for scenario 4. These results are summarized in column 2 of Table 5.18. The performance of these codes in terms of code rate vs. power efficiency in E_b/N_0 is summarized visually in Figure 5.21 which shows both coded and uncoded rates vs. measured E_b/N_0 at a BER of 10^{-4} and their relation to the constellation-constrained capacity.

We once again make the observation that the the experimental data presented here agrees very closely with theory and simulation based on the AWGN channel model. This is further evidence that the underlying mathematical model based on the AWGN channel is a good approximation for the underwater optical communication channel emulated in the tank. An unexplained deviation of the uncoded curve at high SNR is again observed. As discussed earlier in this chapter, such a deviation could be caused by a number of factors (cf., Section 5.1.3).

Range Extension

In the previous section, we saw that the CCSDS turbo can provide significant gains in power efficiency for an underwater optical communication link. We now examine how these improvements in power efficiency can extend the range of reliable underwater communication. As in Section 5.1.3, the link distance expressed in units of attenuation lengths, $d_a = c(\lambda)d$, is used as a water metric to quantify the range extension of a coded system. The relationship between coding gain in SNR and coding gain in d_a, given by (3.35), is used to calculate expected coding gains in d_a based on the empirical coding gains in SNR observed in the previous section. These expected gains are shown in column 2 of Table 5.18. Again, it is implied by (3.35) that the expected gains are in terms of $d_a = c(405\text{ nm})$, i.e. the range extension in attenuation lengths is evaluated at the wavelength of the experimental

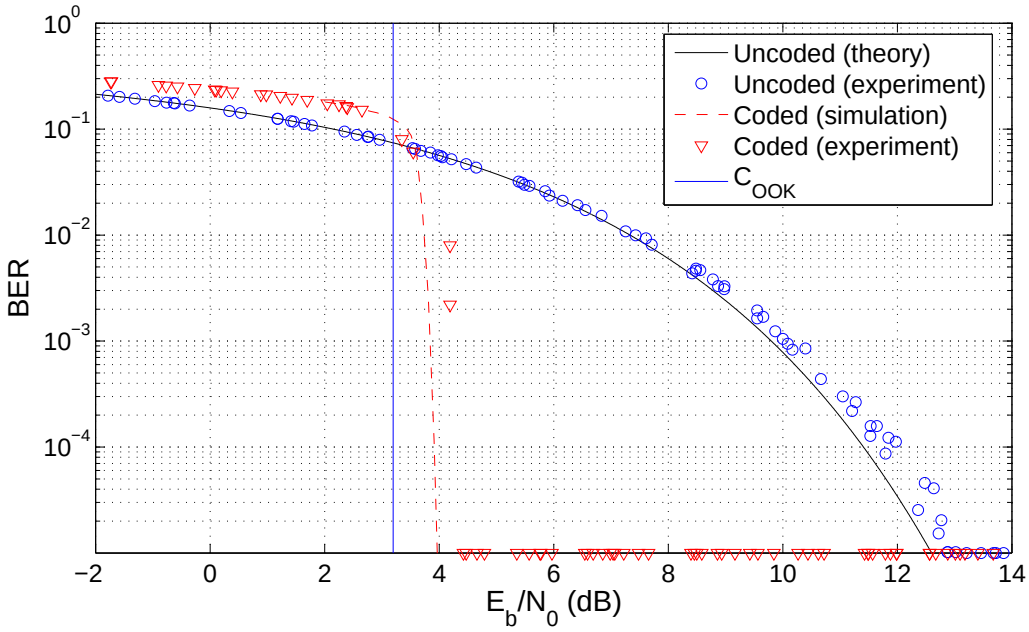


Figure 5.17: Experimental BER vs. E_b/N_0 results for CCSDS scenario 1: $k = 8920$, $r = 1/2$, log-MAP decoding, 16 decoding iterations.

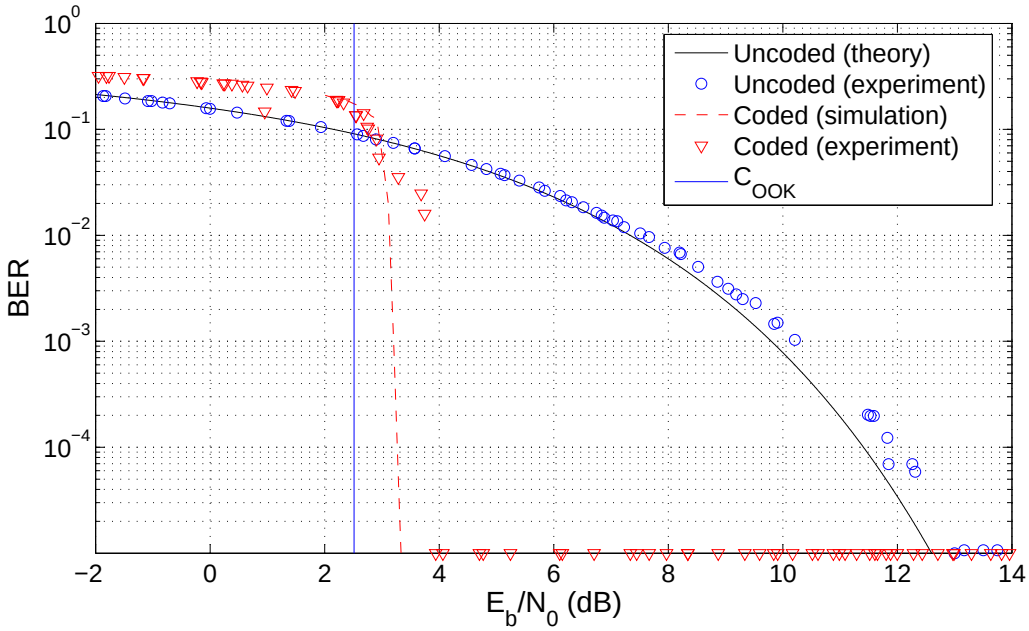


Figure 5.18: Experimental BER vs. E_b/N_0 results for CCSDS scenario 2: $k = 8920$, $r = 1/3$, log-MAP decoding, 16 decoding iterations.

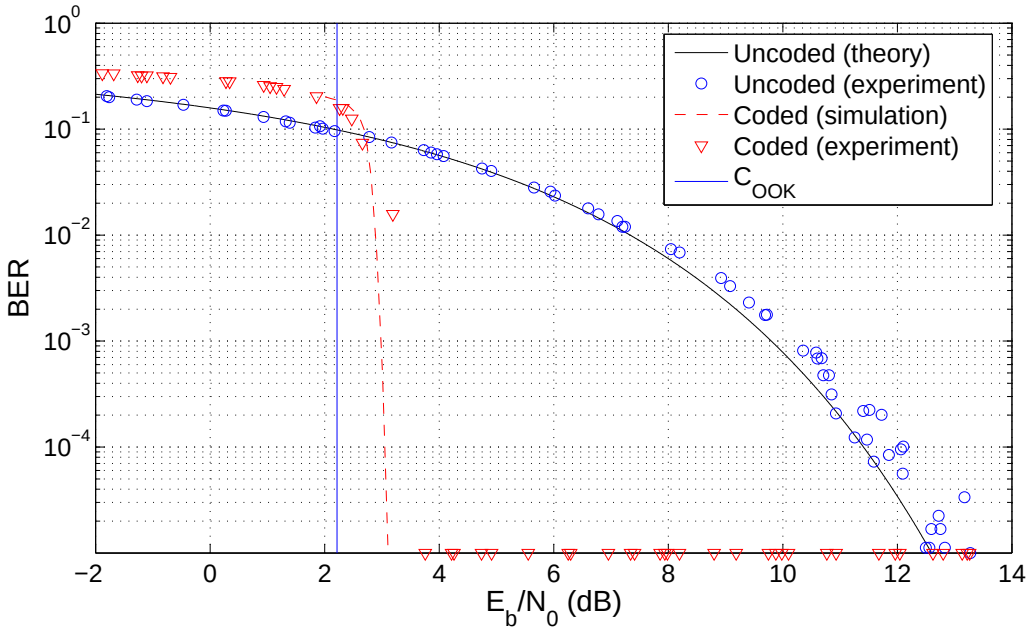


Figure 5.19: Experimental BER vs. E_b/N_0 results for CCSDS scenario 3: $k = 8920$, $r = 1/4$, log-MAP decoding, 16 decoding iterations.

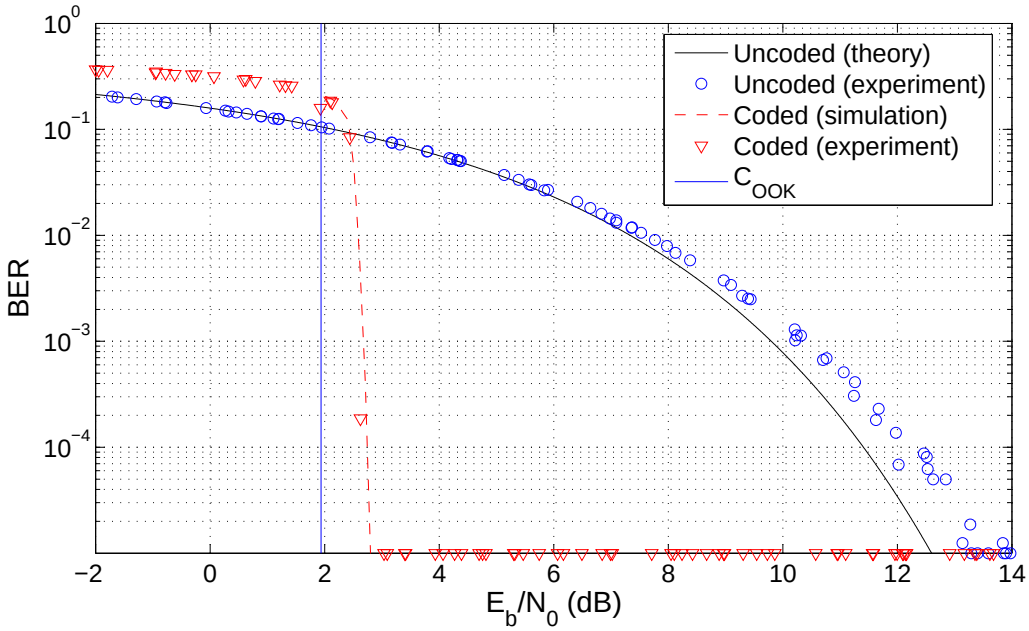


Figure 5.20: Experimental BER vs. E_b/N_0 results for CCSDS scenario 4: $k = 8920$, $r = 1/6$, log-MAP decoding, 16 decoding iterations.

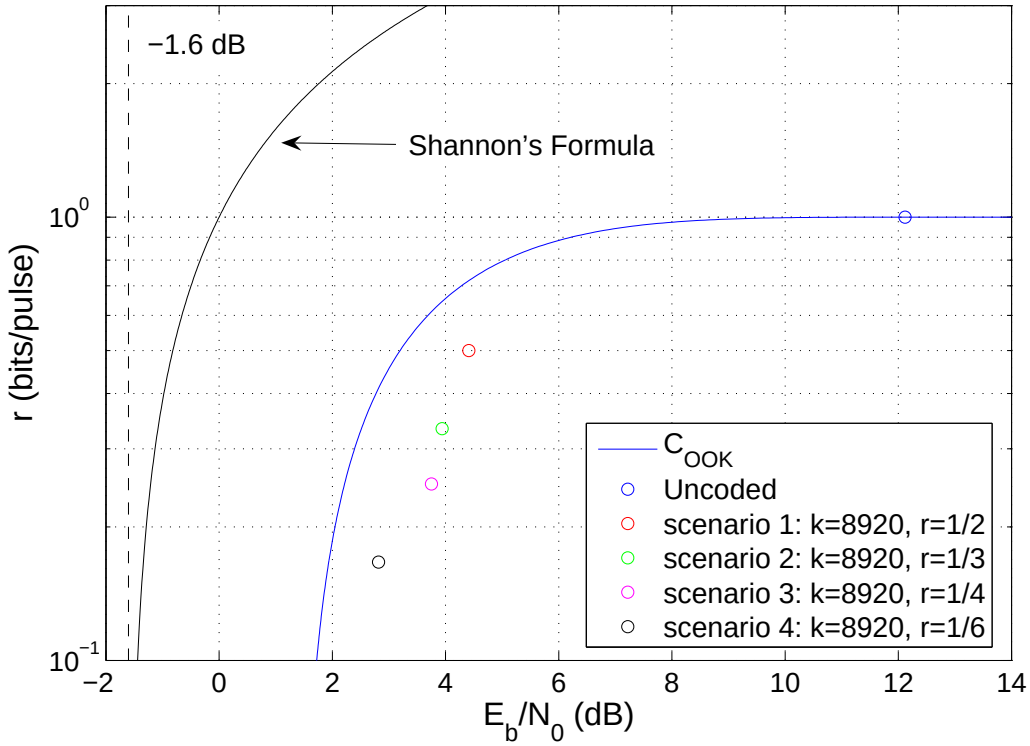


Figure 5.21: Comparison of experimental performance of the CCSDS turbo code with information-theoretic limits.

Table 5.18: Summary of experimental FEC coding gains for CCSDS turbo code.

Scenario	**Gain in E_b/N_0 (dB)**	**Expected gain in $d_a = c(405\text{ nm})d$**	**Measured gain in $d_a = c(530\text{ nm})d$**
1	7.7	0.88	0.88
2	8.1	1.13	0.94
3	8.3	1.30	1.19
4	9.5	1.65	1.21

Table 5.19: Summary of range extension for the CCSDS turbo codes.

Water Type	$c(\lambda)$	**Range Extension (m)**			
		Scenario 1	**Scenario 2**	**Scenario 3**	**Scenario 4**
Clear ocean	0.1514	5.78	6.18	7.87	7.99
Coastal ocean	0.399	2.19	2.34	2.99	3.03
Harbor water	2.195	0.40	0.43	0.54	0.55

system, $\lambda = 405$ nm. We note that the measured coefficient values were taken using a transmissometer operating at 530 nm. Thus, the measured attenuation coefficient values, $d_a = c(530 \text{ nm})d$, will differ from those predicted by (3.35), as discussed in Section 3.5.

Experimental results for BER vs. d_a are shown in Figures 5.22 through 5.25 for scenarios 1 - 4, respectively. Note that these values of d_a are based on measured values of $c(530 \text{ nm})$ from the transmissometer. The coding gains in terms of attenuation lengths are approximately 0.88 for scenario 1, 0.94 for scenario 2, 1.19 for scenario 3, and 1.21 for scenario 4. These results are summarized in column 3 of Table 5.18. Theory and simulation curves as a function of $c(530 \text{ nm})$ were calculated using the same method described in Section 5.1.3.

Following the same approach used previously for the UMTS turbo code, the experimental gains in d_a presented above are used to predict potential range extension that could be attained using these FEC codes. The range extension is examined for three different water conditions: clear ocean water ($c(\lambda) = 0.1514 \text{ m}^{-1}$), coastal water ($c(\lambda) = 0.399 \text{ m}^{-1}$), and turbid harbor water ($c(\lambda) = 2.195 \text{ m}^{-1}$). These attenuation coefficient values are based on measurements in [46] that were made in a spectral band centered at $\lambda \approx 530$ nm. Using the relationship given by (5.10) and assuming that Beer's Law holds at each value of $c(\lambda)$, we are able to predict the range extension achieved by each code scenario in each type of water. These results are summarized in Table 5.19. We note that since the measured attenuation coefficients from the transmissometer were used, the range extension capabilities presented are likely to be conservative.

5.3 Conclusion

In this chapter we evaluated the performance of state-of-the-art turbo codes on an underwater optical communication link operating at 405 nm. Two turbo code standards

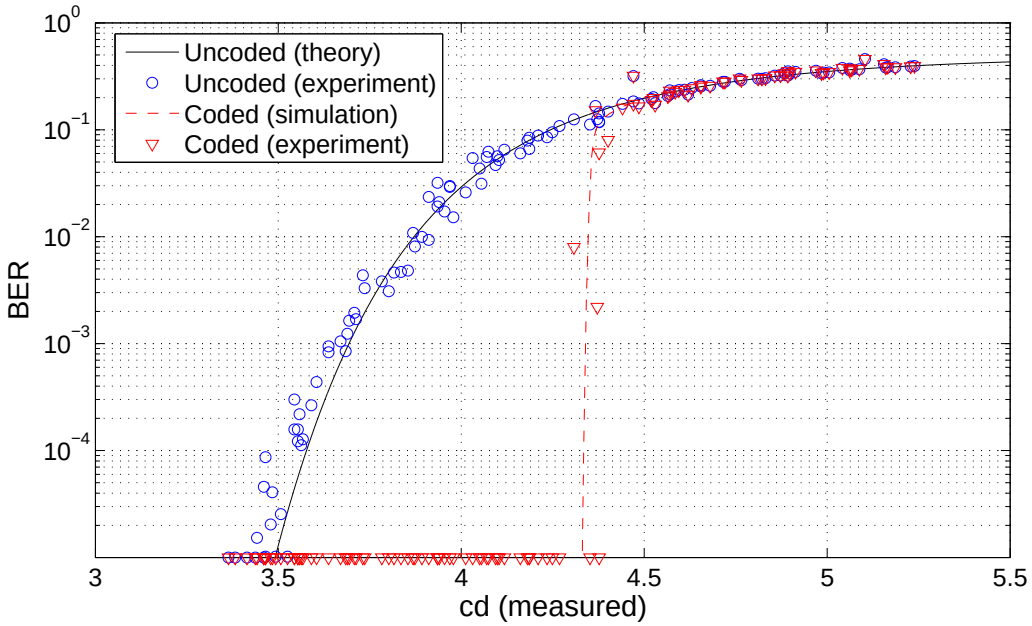


Figure 5.22: Experimental BER vs. d_a results for CCSDS scenario 1: $k = 8920$, $r = 1/2$, log-MAP decoding, 16 decoding iterations.

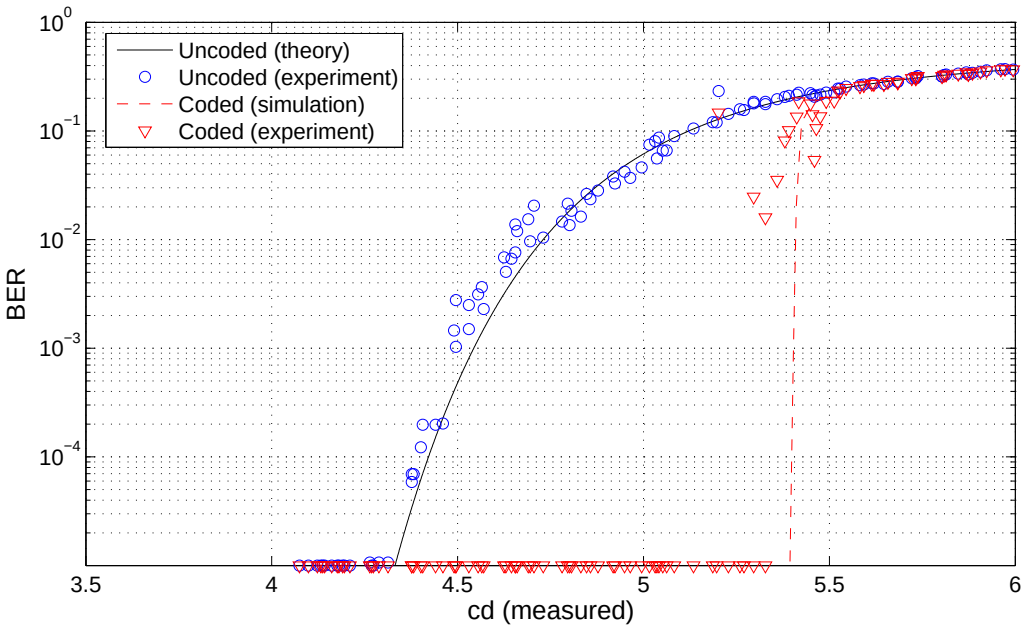


Figure 5.23: Experimental BER vs. d_a results for CCSDS scenario 2: $k = 8920$, $r = 1/3$, log-MAP decoding, 16 decoding iterations.

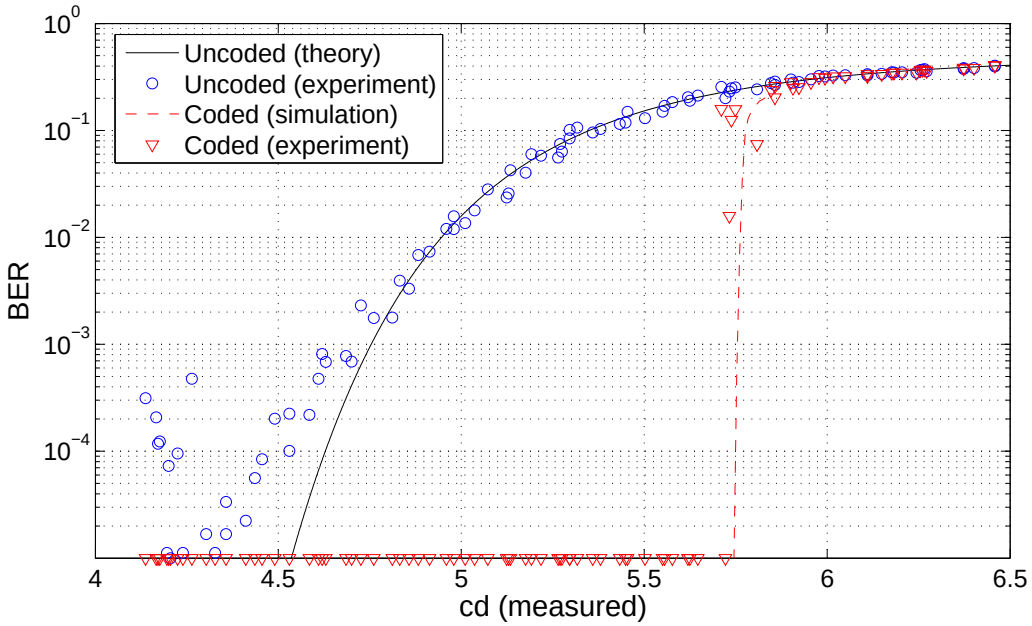


Figure 5.24: Experimental BER vs. d_a results for CCSDS scenario 3: $k = 8920$, $r = 1/4$, log-MAP decoding, 16 decoding iterations.

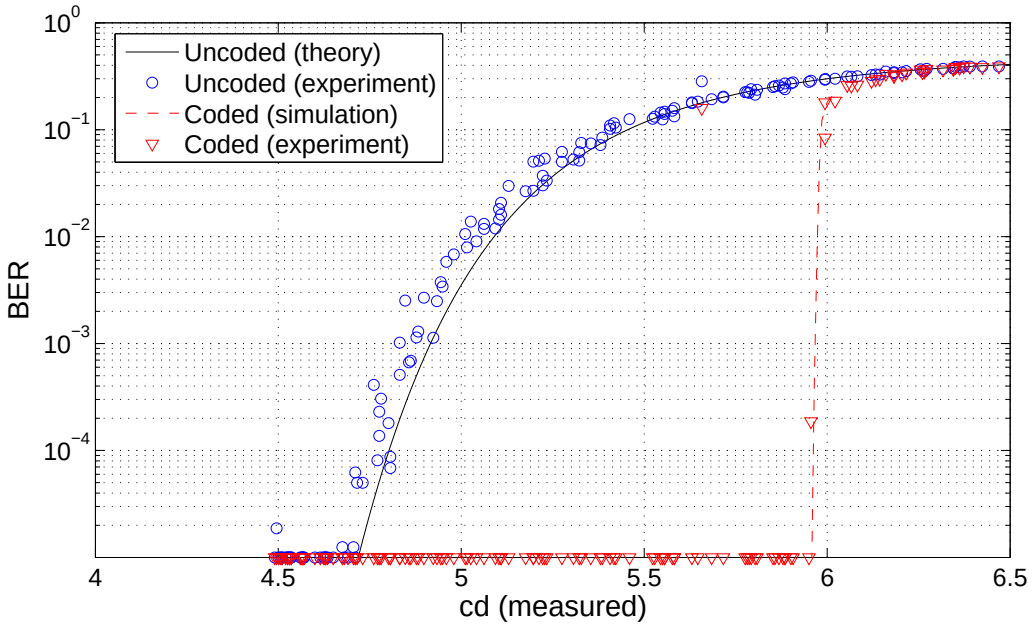


Figure 5.25: Experimental BER vs. d_a results for CCSDS scenario 4: $k = 8920$, $r = 1/6$, log-MAP decoding, 16 decoding iterations.

were investigated and implemented: UMTS and CCSDS. The first turbo code investigated was the UMTS turbo code, which provides a wide selection of frame sizes ranging from $k = 5114$ bits to $k = 40$ bits and operates at a rate $r = 1/3$. The second code investigated was the CCSDS turbo code, which operates at a code rate of $r = 1/2$, $r = 1/3$, $r = 1/4$, or $r = 1/6$, thus providing a trade-off between power efficiency and throughput. Based on simulation, two sets of UMTS code parameters and four sets of CCSDS code parameters were chosen for experimentation. The first UMTS scenario was chosen to have an uncoded frame size $k = 5114$ bits and is decoded using the log-MAP decoding algorithm with 16 decoding iterations. The second UMTS scenario was chosen to have an uncoded frame size $k = 530$ bits and is decoded using the max-log-MAP decoding algorithm with 10 decoding iterations. Both UMTS scenarios operate at a code rate $r = 1/3$. The four CCSDS scenarios each have an uncoded frame size $k = 8920$ bits, are decoded using the log-MAP decoding algorithm with 16 decoding iterations, and have code rates $r = 1/2$, $r = 1/3$, $r = 1/4$, and $r = 1/6$, respectively. UMTS scenario 1 and CCSDS scenarios 1 - 4 were each chosen to demonstrate the maximum gains in power efficiency that can be achieved by each code at each code rate. UMTS scenario 2, however, was chosen to as a potential option for future real-time implementation. It offers more modest coding gains but in return provides decreased decoder complexity and lower latency that may be appealing in a practical system.

Experimental results were collected in a laboratory setting using an underwater optical communication testbed that employs intensity modulation/direct detection using on-off keying with return-to-zero line coding to transmit at a baud rate of 500 kbps. Experimental results show very large coding gains that approach the information-theoretic limits and are consistent with theory and simulation. The experimental coding gains in terms of E_b/N_0 at a BER of 10^{-4} were approximately 9.0 dB for UMTS scenario 1, 6.8 dB for UMTS scenario 2, 7.7 dB for CCSDS scenario 1, 8.1 dB for CCSDS scenario 2, 8.3 dB for CCSDS scenario 3, and 9.5 dB for CCSDS scenario 4. Using attenuation coefficient values measured by a transmissometer during the experiments, the range extensions of the coded systems in units of attenuation lengths, d_a, were calculated. The experimental coding gains in terms of d_a at a BER of 10^{-4} were approximately 1.15 for UMTS scenario 1, 0.89 for UMTS scenario 2, 0.88 for CCSDS scenario 1, 0.94 for CCSDS scenario 2, 1.19 for CCSDS scenario 3, and 1.21 for CCSDS scenario 4. For transmission through clear ocean water, these results suggest a potential range extension for the coded system of up to 7.57 m for

UMTS scenario 1, 5.89 m for UMTS scenario 2, 5.78 m for CCSDS scenario 1, 6.18 m for CCSDS scenario 2, 7.87 m for CCSDS scenario 3, and 7.99 m for CCSDS scenario 4. The coding gains for these systems, in particular CCSDS scenario 4, are very large, which can mean a significant decrease in transmitter power necessary to transmit over a fixed distance and water turbidity. Alternatively, for a fixed transmitter power, they can enable extended link range for a given turbidity or the ability to communicate in more turbid waters for a fixed distance. Based on these substantial coding gains, we have demonstrated that turbo codes can be used to achieve significantly better performance than the Reed-Solomon codes investigated in Chapter 4.

Chapter 6

Low-Density Parity-Check Coding

In this chapter we investigate the application of low-density parity-check (LDPC) codes to the underwater optical communication channel. Specifically, the code designed for the second generation Digital Video Broadcasting satellite standard (DVB-S2) used for broadcasting and other broadband satellite applications is investigated. It will be shown that this powerful FEC coding scheme can provide experimental performance within 0.64 to 0.85 dB of the constellation-constrained capacity for OOK modulation at code rates ranging from $r = 1/2$ to $r = 1/4$. In Section 6.1 the DVB-S2 specification is presented. Then in Section 6.2 simulation results are presented for different code rates and block lengths. In Section 6.3 experimental results are presented and analyzed. The chapter ends with a conclusion and summary of results in Section 6.4.

6.1 DVB-S2 LDPC Code Specification

The LDPC code considered in this thesis is from the second generation Digital Video Broadcasting satellite standard (DVB-S2) for broadcasting, interactive services, news gathering and other broadband satellite applications [58], [59], [60]. The FEC scheme proposed in the DVB-S2 specification utilizes a BCH outer code concatenated in series with an LDPC inner code in order to prevent error floors at very low BERs. It is important to note that in this work we investigate the LDPC code alone without the BCH outer code.

The DVB-S2 encoder systematically encodes an information block of size k into a codeword of length n. The systematic bits appear at the beginning of the codeword, followed

Table 6.1: DVB-S2 code parameters for the normal block length, $n = 64800$. Values for k and n are in bits.

Rate, r	**Dimension**, k	**Block length**, n
1/4	16200	64800
1/3	21600	64800
2/5	25920	64800
1/2	32400	64800
3/5	38880	64800
2/3	43200	64800
3/4	48600	64800
4/5	51840	64800
5/6	54000	64800
8/9	57600	64800
9/10	58320	64800

by the parity bits. Thus, an information block, $\mathbf{u} = [u_0, u_1, \cdots, u_{k-1}]$, results in a codeword of the form $\mathbf{c} = [u_0, u_1, \cdots, u_{k-1}, p_0, p_1, \cdots, p_{n-k-1}]$, where $\mathbf{p} = [p_0, p_1, \cdots, p_{n-k-1}]$ denote the $n-k$ parity bits. The transmission of the codeword starts with the first systematic bit, $c_0 = u_0$, and ends with the final parity bit $c_{n-1} = p_{n-k-1}$.

The DVB-S2 standard specifies two possible block lengths: $n = 64800$ and $n = 16200$. For normal applications, such as digital television broadcast, the longer block length, $n = 64800$, is used. However, for applications that are delay sensitive, such as some real-time interactive services, the shorter block length, $n = 16200$, may be used to reduce latency. A wide range of code rates are available, ranging from $r = 1/4$ to $r = 9/10$ for block length $n = 64800$, and from $r = 1/5$ to $r = 8/9$ for block length $n = 16200$. These rates and their corresponding values of k and n are summarized in Tables 6.1 and 6.2. It should be noted that for the short block length the DVB-S2 standard specifies "code identifier" rates used for pre-signaling purposes (i.e. FEC parameter definition in the layer 1 header) that correspond to the normal block length code rates. In some cases, the code identifier, which we will denote r_{ID}, is different from the actual rate of the code. To be consistent with existing literature, both identifier and actual code rates are presented in Table 6.2.

By definition, LDPC codes are specified by sparse parity-check matrices. There are two problems that arise, however, from defining the parity-check matrix using the "traditional" method originally proposed by Gallager, that is, by generating a pseudo-random

Table 6.2: DVB-S2 code parameters for the short block length, $n = 16200$. Values for k and n are in bits.

Code identifier rate, r_{ID}	**Actual Rate**, r	**Dimension**, k	**Block length**, n
1/4	1/5	3240	16200
1/3	1/3	5400	16200
2/5	2/5	6480	16200
1/2	4/9	7200	16200
3/5	3/5	9720	16200
2/3	2/3	10800	16200
3/4	11/15	11880	16200
4/5	7/9	12600	16200
5/6	37/45	13320	16200
8/9	8/9	14400	16200

matrix that satisfies the low-density parity-check matrix constraints as described in Section 2.4. The first problem is one of encoding complexity. In general, generator matrices are used to encode linear block codes. If the parity-check matrix of such a code is known, a generator matrix can be derived. However, the resulting generator may not be sparse, resulting in high encoding complexity for very large block lengths. The second problem is one of storage requirements. With block lengths on the order of tens of thousands of bits, storing an entire DVB-S2 parity-check matrix would be an extremely inefficient use of memory. To resolve these problems, the DVB-S2 parity-check matrices are structured in a way that simplifies encoding and facilitates description of the code.

First, to reduce encoding complexity, the parity-check matrix is restricted to the form

$$H_{(n-k)\times n} = \left[A_{(n-k)\times k} B_{(n-k)\times(n-k)}\right] \ , \tag{6.1}$$

where $A_{(n-k)\times k}$ is an $(n-k)$ by k pseudo-random submatrix following LDPC conventions and $B_{(n-k)\times(n-k)}$ is an $(n-k)$ by $(n-k)$ submatrix in staircase lower triangular form, as shown in Figure 6.1. Since DVB-S2 codewords are systematic, the task of the encoder is to determine $(n-k)$ parity bits for every k bit information block $\mathbf{u} = [u_0, u_1, \cdots, u_{k-1}]$. Exploiting the staircase lower triangular structure of the $B_{(n-k)\times(n-k)}$ submatrix, it is possible to calculate the parity bits directly from the parity-check matrix, without the need for a generator matrix. This is done by solving the parity-check equations, $\mathbf{c}H^T = \mathbf{0}$,

$$B_{(n-k)\times(n-k)} = \begin{bmatrix} 1 & 0 & 0 & 0 & . & . & 0 & 0 \\ 1 & 1 & 0 & 0 & . & . & 0 & 0 \\ 0 & 1 & 1 & 0 & . & . & 0 & 0 \\ 0 & 0 & 1 & 1 & . & . & 0 & 0 \\ . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . \\ 0 & 0 & 0 & . & . & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & . & . & 1 & 1 \end{bmatrix}$$

Figure 6.1: Staircase lower triangular submatrix of the DVB-S2 parity-check matrix. Adapted from [60].

sequentially starting with p_0, i.e.

$$a_{00}u_0 \oplus a_{01}u_1 \oplus \cdots \oplus a_{0,(k-1)}u_{k-1} \oplus p_0 = 0,$$

$$a_{10}u_0 \oplus a_{11}u_1 \oplus \cdots \oplus a_{1,(k-1)}u_{k-1} \oplus p_0 \oplus p_1 = 0,$$

$$\vdots$$

$$a_{(n-k-1),0}u_0 \oplus a_{(n-k-1),1}u_1 \oplus \cdots \oplus a_{(n-k-1),(k-1)}u_{k-1} \oplus p_{n-k-2} \oplus p_{n-k-1} = 0,$$

where a_{ij} denotes the element in the i-th row and j-th column of $A_{(n-k)\times k}$. Since the $A_{(n-k)\times k}$ submatrix is sparse, this approach results in encoding complexity that is linear with respect to block length.

Second, to reduce storage requirements, a periodicity of $M = 360$ is imposed on the $A_{(n-k)\times k}$ submatrix. Here we make use of the Tanner graph representation of the parity-check matrix developed in Section 2.4. Recall that the Tanner graph consists of bit nodes, c_j, and check nodes, $\mathbf{h_i}$, connected by edges. The graph is bipartite, meaning that edges only connect bit nodes with check nodes, but never two nodes of the same type. To impose periodicity, the k bit nodes of the Tanner graph corresponding to the columns of $A_{(n-k)\times k}$ are divided into groups of M consecutive bit nodes. Consider the first such group in which the first bit node, c_0, has degree d (i.e., it has d adjacent edges in the bipartite graph) and is connected to check nodes $\{\mathbf{h}_{x_1}, \mathbf{h}_{x_2}, \cdots, \mathbf{h}_{x_d}\}$, the indices of which are stored in a lookup table. The remaining $M-1$ bit nodes, $c_1, c_2, \cdots, c_{M-1}$, in the group will also be of degree

d, and the indices of the check nodes connected to the i-th bit node ($1 \leq i \leq M-1$) are calculated by

$$[x_1 + (i-1)q] \bmod (n-k),$$
$$\vdots$$
$$[x_d + (i-1)q] \bmod (n-k),$$

where $(n-k)$ is the total number of check nodes and $q = (n-k)/M$. This process is then repeated for each subsequent group of M bit nodes. With this structure, we need to store in memory only one set of check node indices (corresponding to the first bit node) for each group of M bit nodes. Thus, imposing this periodicity reduces the storage requirement of the DVB-S2 parity-check matrices by a factor of $M = 360$.

Decoding of the DVB-S2 LDPC code is performed using the method described in Section 2.4.2. The sum-product algorithm is used to determine the most likely received codeword by iteratively passing soft information between bit nodes and check nodes. To reduce complexity, the algorithm is implemented in the log domain using log-likelihood ratios (LLRs) as messages, as described in [61]. The decoding algorithm iterates until a valid codeword is found or a maximum number of decoding iterations are completed. Since the sum-product algorithm is a *stop-when-its-done* decoding algorithm, meaning it runs for the minimum number of iterations necessary to converge on a codeword, the maximum iterations value is typically chosen to be larger than that of other iterative decoding algorithms, such as the iterative decoding algorithms used for turbo codes. Typical maximum iterations values for LDPC codes range from approximately 30 to 100 iterations. The performance of the LDPC decoder is affected by its scheduling algorithm, which determines the order in which bit nodes and check nodes are updated. For our implementation, all check nodes are updated and subsequently all bit nodes are updated in each iteration. Thus, a new messages is passed across every edge of the Tanner graph in both the vertical step and the horizontal step in every iteration. This approach is often referred to as the flooding schedule [62].

6.2 Simulation Results

To provide a theoretical baseline for the underwater experiments, we simulated the performance of the DVB-S2 LDPC code for the AWGN channel. Simulations were run

in MATLAB using the Coded Modulation Library (CML) software developed by Iterative Solutions [55]. Both normal and short block lengths were simulated for each code rate. Simulation results are shown in Figures 6.2 and 6.3 for normal block length ($n = 64800$) and short block length ($n = 16200$), respectively. Based on the channel model developed in Section 3.3, the bit-error rate (BER) for OOK modulation in AWGN is given by

$$P_b = Q\left(\sqrt{\frac{E_b}{N_0}}\right), \tag{6.2}$$

where E_b/N_0 is the bit-energy-to-noise-density-ratio, and Q is the q-function (cf.,[51, p. 40]). Uncoded OOK achieves a BER of 10^{-4} (the regime of interest for this system) at an E_b/N_0 of 11.41 dB. The coding gain in E_b/N_0 that can potentially be obtained at this BER for a block length of $n = 64800$ therefore ranges from 8.34 dB for rate 1/4 to 4.62 dB for rate 9/10. These gains are summarized in Table 6.3. The coding gains that can potentially be obtained using the short block length, $n = 16200$, are slightly smaller, ranging from 8.20 dB for rate 1/5 ($r_{ID} = 1/4$) to 4.70 dB for rate 8/9. These gains are summarized in Table 6.4.

The constellation-constrained capacity is a fundamental limit that gives the optimum trade-off between channel utilization and power efficiency for a specific modulation scheme. By comparing the above simulation results to the constellation-constrained capacity, we can observe how well these codes perform relative to the best possible performance predicted by information theory. Recall from Section 3.3 that the constellation-constrained capacity for OOK modulation over the AWGN channel is given by

$$C_{OOK} = \frac{\rho}{2}\log_2(e) - \frac{e^{\frac{-\rho}{4}}}{\sqrt{2\pi}}\int e^{\frac{-t^2}{2}}\cosh\left(t\sqrt{\frac{\rho}{2}}\right)\log_2\left[\cosh\left(t\sqrt{\frac{\rho}{2}}\right)\right]dt \tag{6.3}$$

where C_{OOK} is the constellation-constrained capacity in units of data bits/pulse and ρ is the SNR. From (3.12), we apply the substitution

$$\frac{E_b}{N_0} = \frac{\rho}{2C_{OOK}} \tag{6.4}$$

to (6.3) to obtain an implicit relationship between the constellation-constrained capacity, C_{OOK} (in bits/pulse), and E_b/N_0. By evaluating both C_{OOK} and E_b/N_0 for different values of ρ, it is possible to generate the best possible E_b/N_0 for any given code rate, $C_{OOK} = r$. The DVB-S2 LDPC code is designed to operate at an E_b/N_0 very close to the information-theoretic limits for the AWGN channel. The distance of each code at a BER of 10^{-4} from

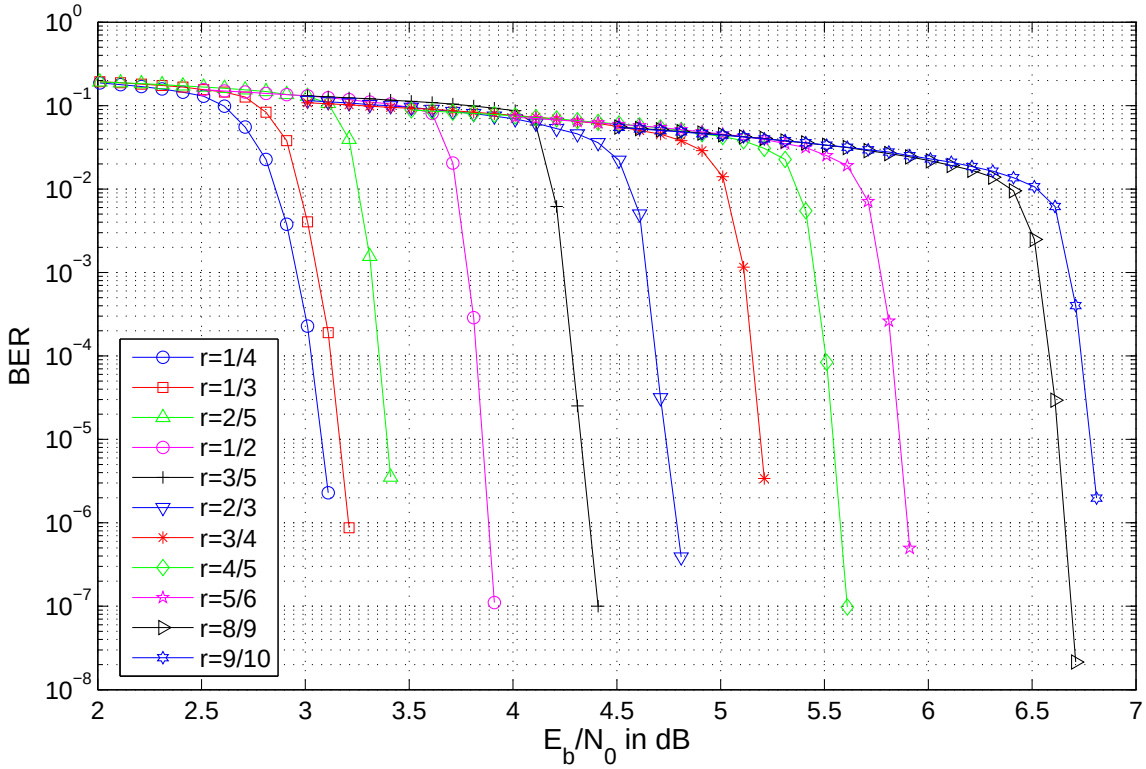


Figure 6.2: Simulated performance of the DVB-S2 LDPC code in AWGN for normal block length ($n = 64800$) and 100 maximum decoding iterations (BER vs. E_b/N_0).

Table 6.3: Summary of simulated performance of the DVB-S2 LDPC code in AWGN for normal block length ($n = 64800$) and 100 maximum decoding iterations. Coding gains and distance from constellation-constrained capacity, C_{OOK}, are in terms of E_b/N_0 in dB.

Rate, r	**Simulated Coding gain (dB)**	**Distance to C_{OOK} (dB)**
1/4	8.34	0.85
1/3	8.25	0.64
2/5	8.00	0.63
1/2	7.53	0.68
3/5	7.10	0.62
2/3	6.70	0.64
3/4	6.21	0.57
4/5	5.90	0.46
5/6	5.54	0.50
8/9	4.80	0.56
9/10	4.62	0.58

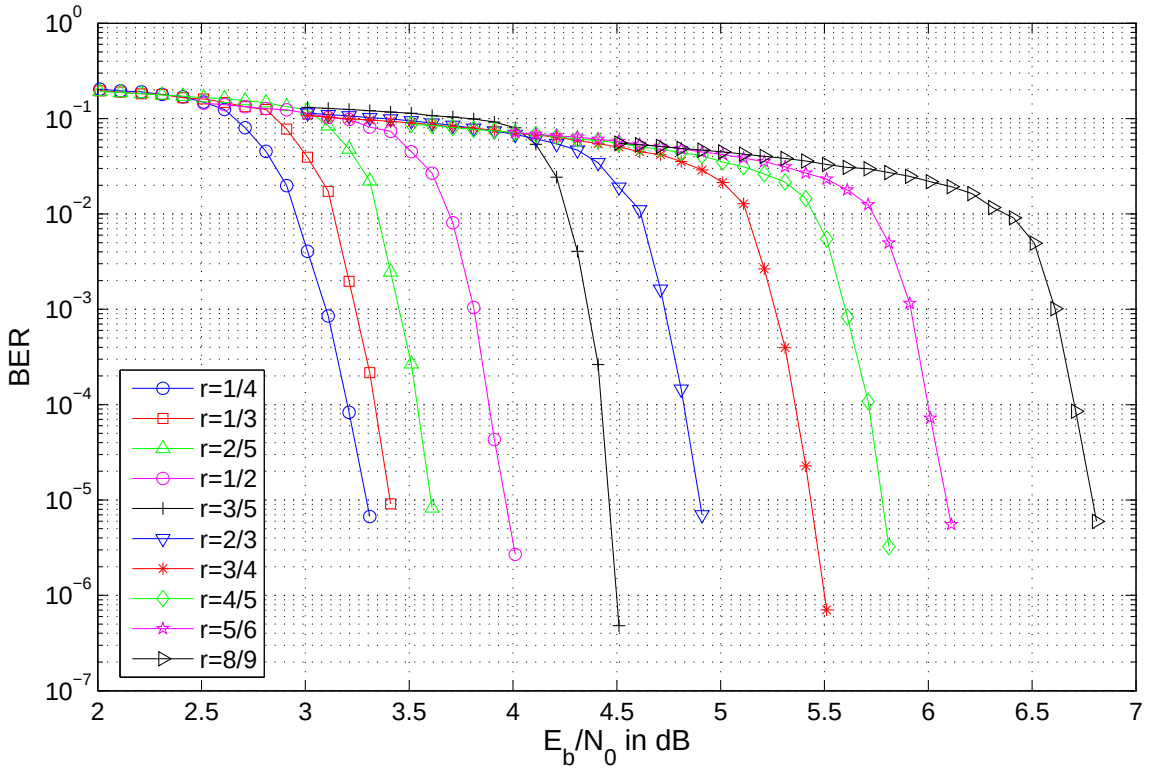


Figure 6.3: Simulated performance of the DVB-S2 LDPC code in AWGN for short block length ($n = 16200$) and 100 maximum decoding iterations (BER vs. E_b/N_0).

Table 6.4: Summary of simulated performance of the DVB-S2 LDPC code in AWGN for short block length ($n = 16200$) and 100 maximum decoding iterations. Coding gains and distance from constellation-constrained capacity, C_{OOK}, are in terms of E_b/N_0 in dB.

Code identifier rate, r_{ID}	**Actual Rate**, r	**Simulated Coding gain (dB)**	**Distance to C_{OOK} (dB)**
1/4	1/5	8.20	1.16
1/3	1/3	8.04	0.85
2/5	2/5	7.83	0.80
1/2	4/9	7.50	0.95
3/5	3/5	6.94	0.78
2/3	2/3	6.57	0.77
3/4	11/15	6.02	0.88
4/5	7/9	5.69	0.86
5/6	37/45	5.40	0.75
8/9	8/9	4.70	0.66

C_{OOK} is summarized in Tables 6.3 and 6.4. Depending on block length and code rate, the simulated performance was within an E_b/N_0 of 0.46 dB to 1.16 dB of the constellation-constrained capacity.

The best performing codes for a given rate in terms of power efficiency are the normal block length ($n = 64800$) codes. However, this performance comes at a cost of increased latency due to block length. On the other hand, the short block length codes ($n = 16200$) offer slightly lower gains in power efficiency (on the order of 0.1 to 0.2 dB less), but in return provide lower latency due to decreased decoding time per block, which scales linearly with block length, n. This trade-off makes the short block length codes a potentially attractive option for practical systems particularly if real-time applications are used.

6.3 Experimental Results

We now present experimental measurements of the BER performance of the DVB-S2 LDPC code when used on the underwater optical communication channel. Based on simulated performance in the previous section, four sets of parameters, designated scenario 1 - 4, were selected for experimentation. The parameters used in each scenario are summarized in Table 6.5. Scenarios 1, 2, and 3 have block length $n = 64800$ and code rates $r = 1/2$, $r = 1/3$, and $r = 1/4$, respectively. These codes were chosen for their large coding gains, which are close to the constellation-constrained capacity. Scenario 4 is the rate $r = 4/9$ ($r_{ID} = 1/2$) block length $n = 16200$ code. This code is included as an example of a short block length code that offers slightly smaller coding gain in exchange for lower latency. To further reduce latency, the maximum number of decoding iterations for scenario 4 has been reduced to 50 iterations. Based on simulation, this comes at a cost of less than 0.1 dB power efficiency and in return the decoding time of an uncorrectable block is reduced by a factor of 1/2 as compared to the case in which 100 maximum decoding iterations are used. These aspects make this scenario attractive for future implementation in a real-time system.

Experimental results were obtained using the experimental testbed and procedure described in Section 3.2. The encoder and decoder were implemented in MATLAB using functions from the CML [55] in addition to software written by the author. Experimental packets are generated prior to experimentation. As with previous the experiments, limita-

Table 6.5: DVB-S2 experimental scenarios.

Scenario	**Rate**	**Block length,** n	**Dimension,** k	**Max. Decoding Iterations**
1	1/2	64800	32400	100
2	1/3	64800	21600	100
3	1/4	64800	16200	100
4	1/2	16200	7200	50

tions in buffer size constrain the packet size to at most $200,000$ bits, including a 1024 bit packet header. To maximize the number of information bits sent per packet, the largest multiple of the block length that is below this limit is used. For each scenario, this results in $194,400$ coded bits (approx. 24 KB) per packet. In terms of information bits per packet, this results in $97,200$ information bits (approx. 12 KB) for $r = 1/2$, $64,800$ information bits (approx. 8 KB) for $r = 1/3$, and $48,600$ information bits (approx. 6 KB) for $r = 1/4$. In order to have an accurate basis for calculating empirical coding gains, it is necessary to measure the performance of the uncoded system for each experiment. A single transmitted packet is used to estimate the BER of the system both with and without coding, where the BER of the uncoded system is determined by performing a hard decision on the received data and calculating the BER prior to decoding. This approach eliminates the need to send two packets for the same water condition.

As in previous chapters, the accuracy achieved using the packet sizes given above is quantified in terms of confidence intervals. Recall that the 95% confidence interval, denoted (y_+, y_-), is defined as the interval about the BER estimate, $\hat{p}$, such that

$$\text{Prob}\left[y_+ \leq p \leq y_-\right] = 0.95, \tag{6.5}$$

where y_+ is the upper bound on the confidence interval, y_- is the lower bound on the confidence interval, and p is the actual BER. Furthermore, the bounds on the 95% confidence interval can be closely approximated using Equation (4.6) (cf., Section 4.2). Substituting into this equation the DVB-S2 packet sizes listed above, it is possible to calculate the 95% confidence intervals for a BER estimate of $\hat{p} = 10^{-4}$, the BER of interest for this system. These confidence intervals are summarized in Table 6.6.

At the receiver, the decoding algorithm is initialized by calculating the LLRs of

Table 6.6: Summary of 95% confidence intervals for DVB-S2 packets at BER $\hat{p} = 10^{-4}$.

Scenario	**Coded System**	**Uncoded System**
1	$(1.86\hat{p},\ 0.54\hat{p})$	$(1.55\hat{p},\ 0.64\hat{p})$
2	$(2.12\hat{p},\ 0.47\hat{p})$	$(1.55\hat{p},\ 0.64\hat{p})$
3	$(2.37\hat{p},\ 0.42\hat{p})$	$(1.55\hat{p},\ 0.64\hat{p})$
4	$(1.93\hat{p},\ 0.52\hat{p})$	$(1.55\hat{p},\ 0.64\hat{p})$

the received sequence prior to decoding. Recall from Section 2.3.2 that the LLR is equal to

$$R(c_i) = \ln\left[\frac{P(y_i|c_i = 1)}{P(y_i|c_i = 0)}\right] \tag{6.6}$$

where $P(y_i|c_i = b)$ is the conditional probability of receiving y_i given that $c_i = b$ was transmitted. For BPSK modulation with equiprobable symbols $\{+1, -1\}$, the LLR is simply

$$R(c_i) = 2\rho y_i \tag{6.7}$$

where ρ is the SNR (cf.,[15, pp. 780-781]). Based on this equation, the LLRs of the received data for the OOK modulated system can be calculated similarly as

$$R(c_i) = \rho \bar{y}_i \tag{6.8}$$

where $\bar{y}_i$ is the OOK received sequence normalized to zero mean and noise variance one, as defined in Section 3.3[1]. In this equation the factor of 2 goes away when changing from BPSK to OOK modulation due to the fact that $\rho_{OOK} = 2\rho_{BPSK}$.

6.3.1 BER Performance

The experimentally measured BER for scenarios 1 - 4 are plotted versus E_b/N_0 in Figures 6.4 - 6.7, respectively, along with the performance of uncoded OOK modulation. In each plot, the uncoded data are represented by blue circles and the coded data are represented by red triangles. Points plotted on the x axis have a BER$\leq 10^{-5}$. Also shown for comparison are:

[1]Prior to forming the LLR, the received data is normalized to match the statistics expected by the CML decoding algorithm. The received sequence (normalized to mean zero) is modeled according to the CML algorithm as $\tilde{y}_i = a\left(\bar{x}_i + \sqrt{\frac{2}{\rho}}\bar{n}_i\right)$, where $\bar{x}_i \in \{+1, -1\}$, $\bar{n}_i \sim N(0, 1)$, and a is a constant scale factor. Note that this definition differs from $\bar{y}_i$. To compensate, the scale factor is estimated for each received packet by taking the variance of $\tilde{y}_i$ and solving for a. The LLR is then formed using the equation $R(c_i) = \frac{\rho}{\hat{a}}\tilde{y}_i$, where $\hat{a}$ is the estimate of a.

- the theoretical BER for the uncoded system given by (6.2),
- the simulated BER for the coded system from Section 6.2,
- the constellation-constrained capacity for OOK at the appropriate code rate given by (3.18), or more precisely, this is the E_b/N_0 at which $C_{OOK} = r$.

For each of these curves, the ideal AWGN channel is assumed. From the figures, we see that the experimental results for the coded system agree very closely with simulation, differing by less than 0.25 dB in all four plots. The experimental results for the uncoded system follow close to theory but deviate slightly at high SNRs, performing about 1 dB worse than theory at a BER of 10^{-4}. As a result, the measured coding gain is about 1 dB better than the gains predicted by simulation. The resulting coding gains in terms of E_b/N_0 are approximately 8.4 dB for scenario 1, 8.9 dB for scenario 2, 9.2 dB for scenario 3, and 7.7 dB for scenario 4. These results are summarized in column 2 of Table 6.7. The performance of these codes in terms of code rate vs. power efficiency in E_b/N_0 is summarized visually in Figure 6.8 which shows both coded and uncoded rates vs. E_b/N_0 and their relation to the constellation-constrained capacity.

The fact that the experimental data agrees so closely with theory and simulation supports the AWGN channel model for the underwater optical communications link. In particular, since the decoding of an LDPC code requires knowledge of channel statistics to form the received symbol LLRs, the fact that the experimental coded data falls right on top of simulation is a strong indicator that the model is reasonably accurate. However, the systematic deviation of the uncoded data from the theory curve at high SNR suggests a deviation from the AWGN channel model. This could be caused by a number of factors. It could be due to non-Gaussian sources of noise or the presence of weak fading due to turbulence. It could also be caused by the experimental setup itself, possibly due to quantization error at high signal levels or imprecise synchronization. Based on these experimental results, the cause is yet unclear but is beyond the scope of this thesis. Investigating the behavior of the uncoded system at high SNR is an interesting subject for future work.

6.3.2 Range Extension

In the previous section, we saw that LDPC coding can significantly improve the power efficiency, E_b/N_0, of an underwater optical communication link. We now examine

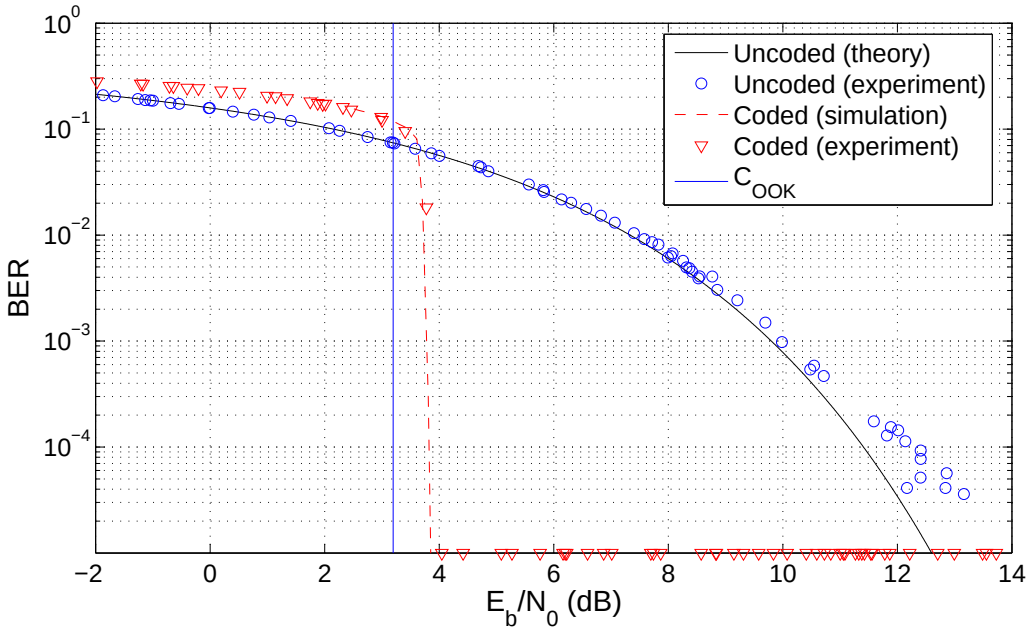


Figure 6.4: Experimental BER vs. E_b/N_0 results for DVB-S2 scenario 1: $n = 64800$, $r = 1/2$, 100 max. decoding iterations.

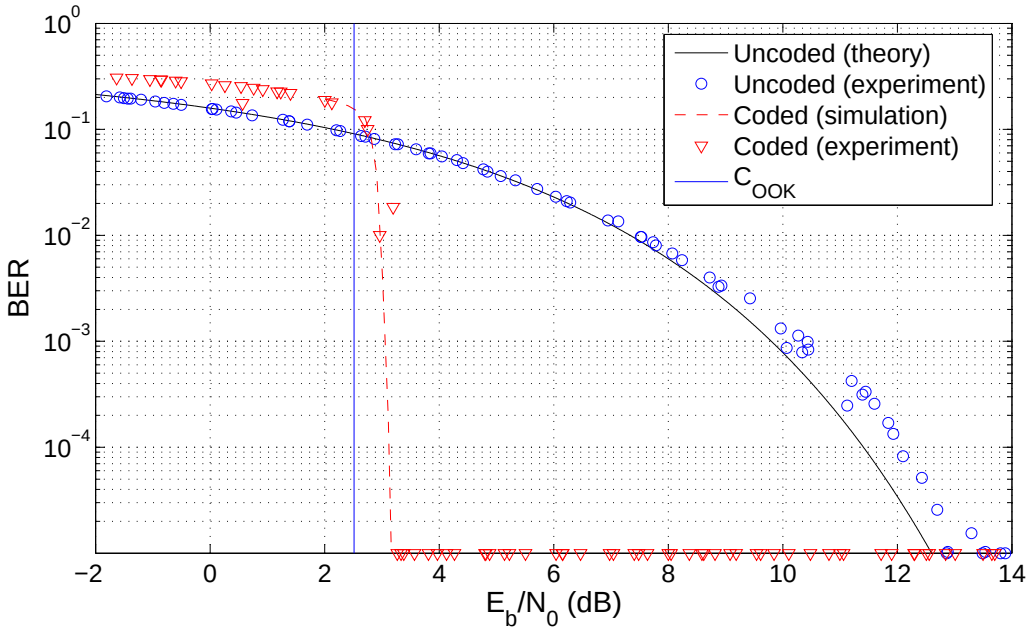


Figure 6.5: Experimental BER vs. E_b/N_0 results for DVB-S2 scenario 2: $n = 64800$, $r = 1/3$, 100 max. decoding iterations.

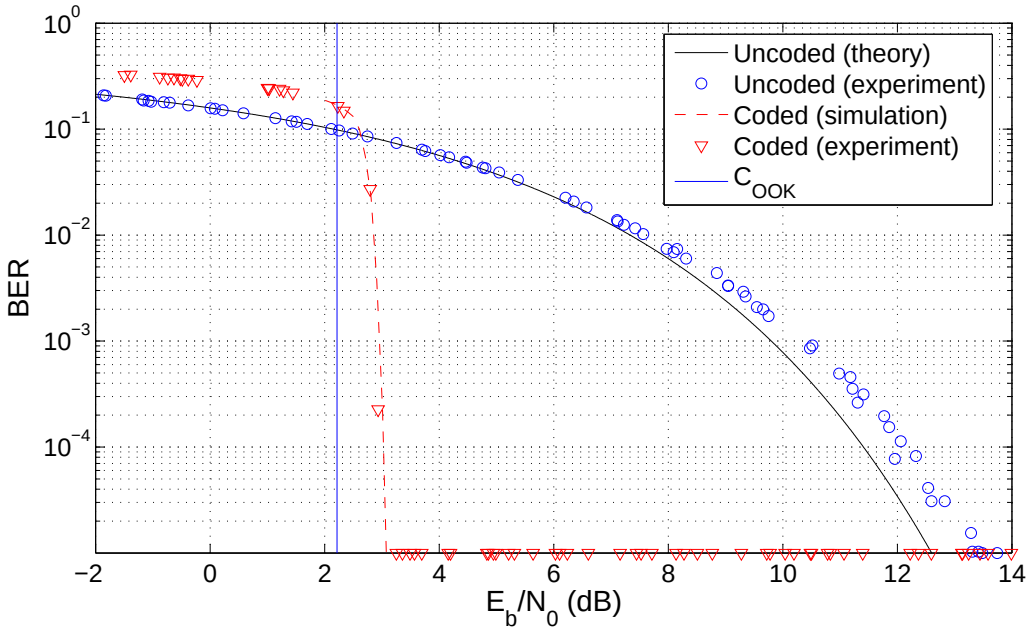


Figure 6.6: Experimental BER vs. E_b/N_0 results for DVB-S2 scenario 3: $n = 64800$, $r = 1/4$, 100 max. decoding iterations.

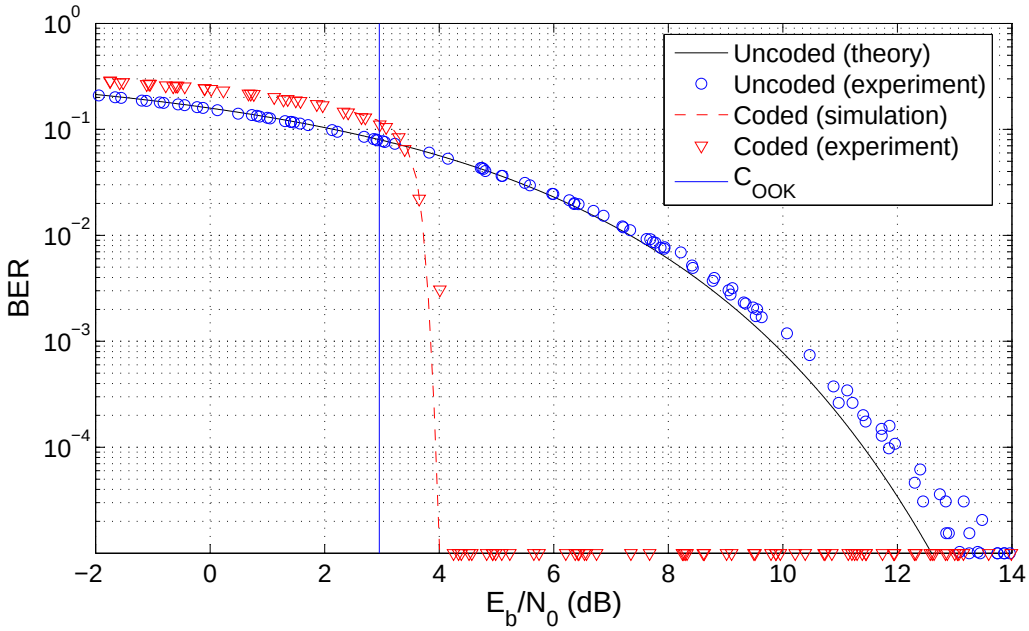


Figure 6.7: Experimental BER vs. E_b/N_0 results for DVB-S2 scenario 4: $n = 16200$, $r = 1/2$, 50 max. decoding iterations.

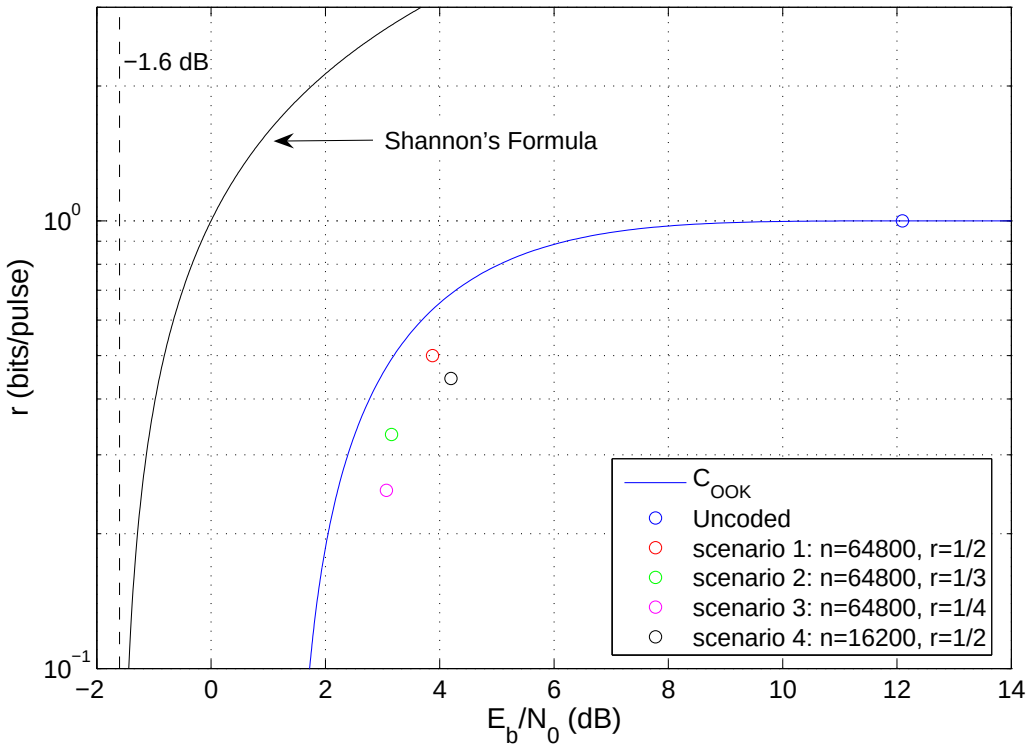


Figure 6.8: Comparison of experimental rates vs. E_b/N_0 for DVB-S2 LDPC code.

Table 6.7: Summary of experimental FEC coding gains for DVB-S2 LDPC code.

Scenario	**Gain in E_b/N_0 (dB)**	**Expected gain in $d_a = c(405\text{ nm})d$**	**Measured gain in $d_a = c(530\text{ nm})d$**
1	8.4	1.32	0.79
2	8.9	1.57	1.04
3	9.2	1.75	1.06
4	7.7	1.24	0.86

how these improvements in power efficiency can extend the range of reliable underwater communication. Recall from Section 3.5 that the relationship between optical intensity, turbidity, and distance is given by Beer's Law:

$$I = I_0 e^{-c(\lambda)d}, \tag{6.9}$$

where I is the optical intensity at the receiver, I_0 is the transmitted optical intensity, $c(\lambda)$ is the attenuation coefficient as a function of wavelength, and d is the optical path length in meters. Thus, (6.9) suggests that the SNR at the receiver is a decreasing function of $d_a = c(\lambda)d$, which we have defined as the link distance expressed in units of attenuation lengths. The attenuation length, $c(\lambda)^{-1}$, is the distance at which the optical intensity has been decreased by a factor of e^{-1}. Since $c(\lambda)$ is a function of wavelength, we note that d_a will also vary with wavelength for a given water condition. Expressing the range of the system in units of attenuation lengths, d_a, has the advantage that the coding gain can easily be extrapolated to various distances and water turbidities.

It was previously shown in Section 3.5 that an expected coding gain in d_a can be calculated based on coding gain in SNR. This relationship is given by

$$\Delta d_a = \frac{\Delta\rho(\text{dB})}{20\log_{10}(e)}, \tag{6.10}$$

where $\Delta\rho(\text{dB})$ is the coding gain in SNR in dB. Using (6.10) and the experimental coding gains in SNR measured in Section 6.3.1, the expected coding gains in d_a were calculated for each scenario. These expected gains are shown in column 2 of Table 6.7. Note that these expected gains are in terms of $d_a = c(405\text{ nm})d$, in which the attenuation lengths are evaluated at 405 nm, the operating wavelength of the experimental system. Empirical measurements of the attenuation coefficient collected during experimentation were collected using a transmissometer operating at 530 nm. As discussed in Section 3.5, a link operating at $\lambda = 405$ nm will suffer greater attenuation than light operating at $\lambda = 530$ nm for the water conditions emulated in the tank. Thus,

$$c(405\text{ nm})d > c(530\text{ nm})d, \tag{6.11}$$

which means that the expected gains calculated using (6.10) will be less than the gains measured gains. However, reliably converting gains in $d_a = c(530\text{ nm})d$ into gains in $d_a = c(405\text{ nm})d$ will require further experimental investigation and is beyond the scope of

Table 6.8: Summary of range extension for the DVB-S2 LDPC codes.

Water Type	$c(\lambda)$	**Range Extension (m)**			
		Scenario 1	**Scenario 2**	**Scenario 3**	**Scenario 4**
Clear ocean	0.1514	5.22	6.87	7.00	5.68
Coastal ocean	0.399	1.98	2.61	2.66	2.16
Harbor water	2.195	0.36	0.47	0.48	0.39

this thesis. Thus, experimental results are presented in terms of the $d_a = c(530\text{ nm})d$ using measured attenuation coefficient values from the transmissometer. We note, however, that actual coding gains in d_a are likely larger than those presented.

Experimental results for BER vs. d_a for scenarios 1 - 4 are shown in Figures 6.9 - 6.12 respectively. Note that these values of d_a are based on measured values of $c(530\text{ nm})$ from the transmissometer. The coding gains in terms of attenuation lengths are approximately 0.79 for scenario 1, 1.04 for scenario 2, 1.06 for scenario 3, and 0.86 for scenario 4. These results are summarized in column 3 of Table 6.7. Theory and simulation curves as a function of $c(530\text{ nm})$ were calculated as follows. For each set of experimental data, the measured $c(530\text{ nm})$ values from the transmissometer were first plotted as a function of estimated SNR. These plots, shown in Appendix A, demonstrate a linear relationship between $\rho(\text{dB})$ and $c(530\text{ nm})$. To establish this relationship explicitly, a linear, least-square regression was used to determine a line of best fit for each experiment. We observe that the slopes of these lines are approximately the same, however the y-intercept varies by experiment. Using these lines, it is possible to convert SNR to $c(530\text{ nm})$ for each experiment.

Using the experimental gains in d_a, we can extrapolate the potential range extension that could be attained using these FEC codes. We examine the range extension achieved by each code in three different water conditions: clear ocean water ($c(\lambda) = 0.1514\text{ m}^{-1}$), coastal water ($c(\lambda) = 0.399\text{ m}^{-1}$), and turbid harbor water ($c(\lambda) = 2.195\text{ m}^{-1}$). These attenuation coefficient values are based on measurements in [46] that were made in a spectral band centered at $\lambda \approx 530$ nm. Using the relationship

$$d = \frac{d_a}{c(530\text{ nm})} \tag{6.12}$$

and assuming that Beer's Law holds at each value of $c(\lambda)$, we are able to predict the range extension achieved by each code scenario in each type of water. These results are summarized in Table 6.8. We note that since the measured attenuation coefficients from the

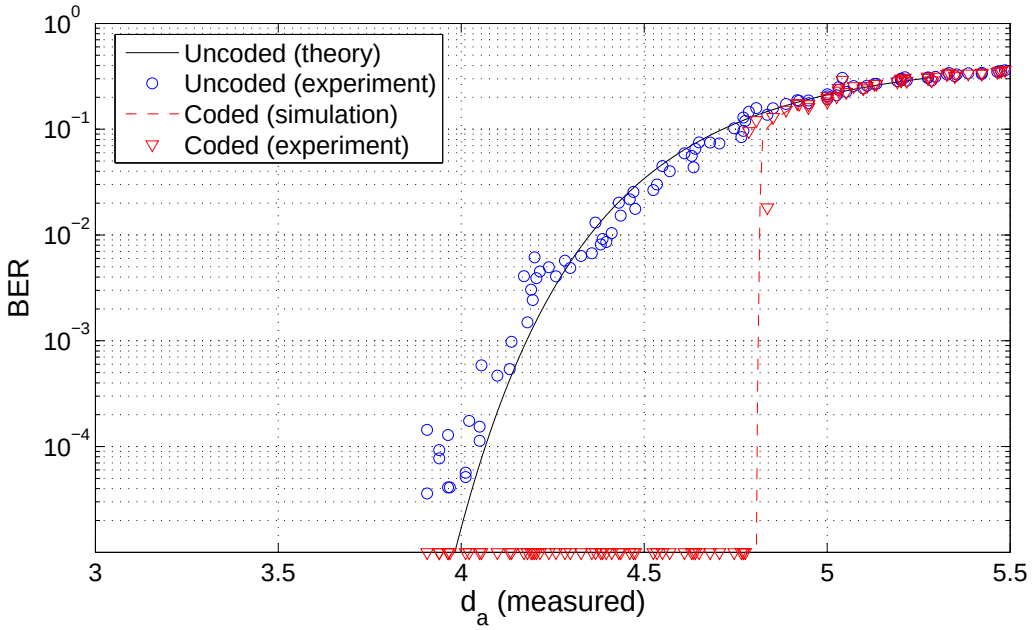


Figure 6.9: Experimental BER vs. d_a results for DVB-S2 scenario 1: $n = 64800$, $r = 1/2$, 100 max. decoding iterations.

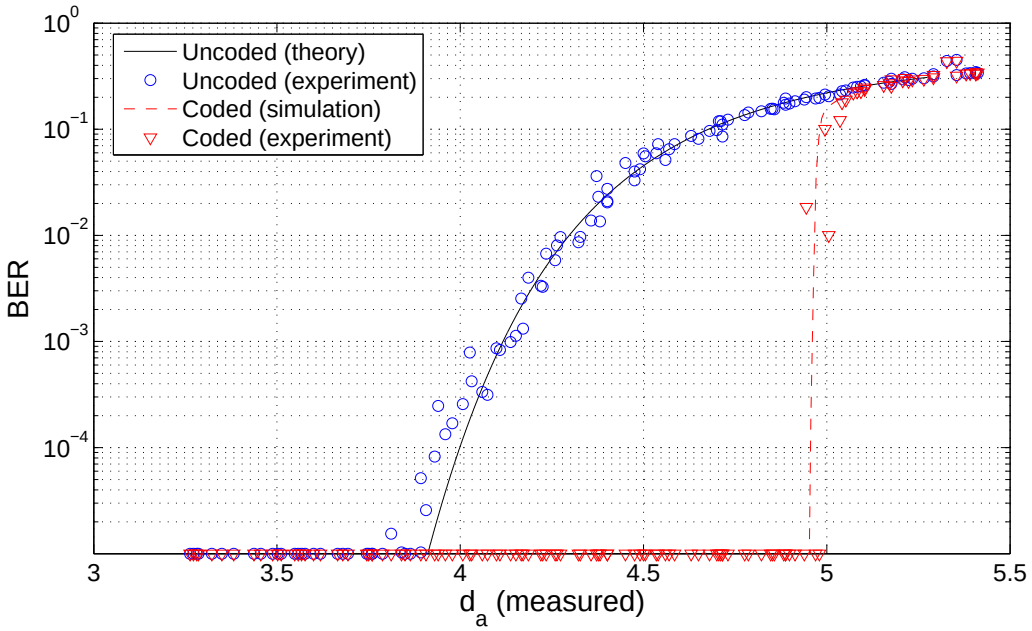


Figure 6.10: Experimental BER vs. d_a results for DVB-S2 scenario 2: $n = 64800$, $r = 1/3$, 100 max. decoding iterations.

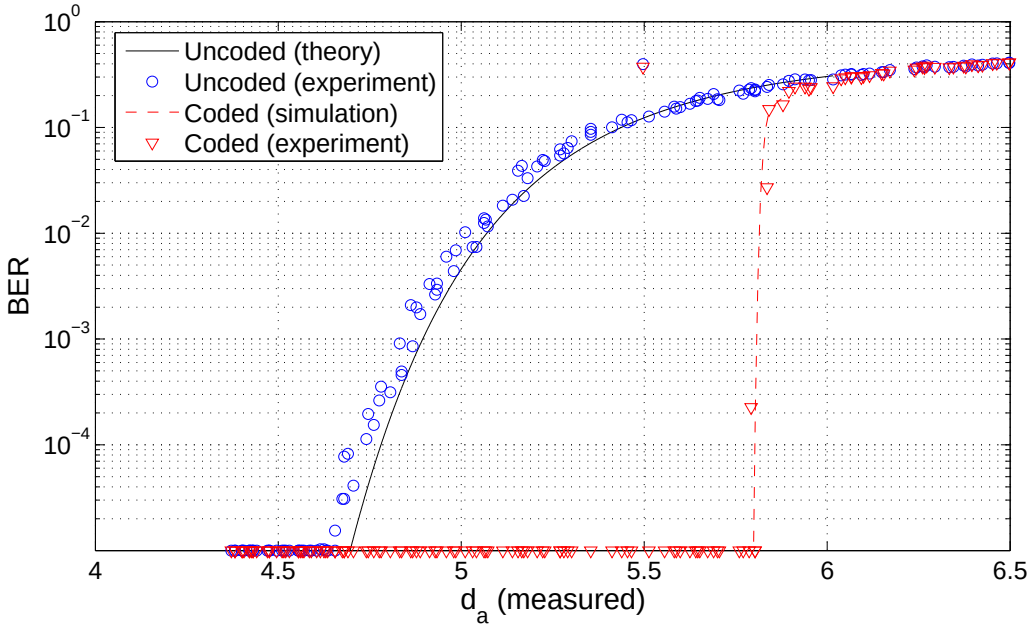


Figure 6.11: Experimental BER vs. d_a results for DVB-S2 scenario 3: $n = 64800$, $r = 1/4$, 100 max. decoding iterations.

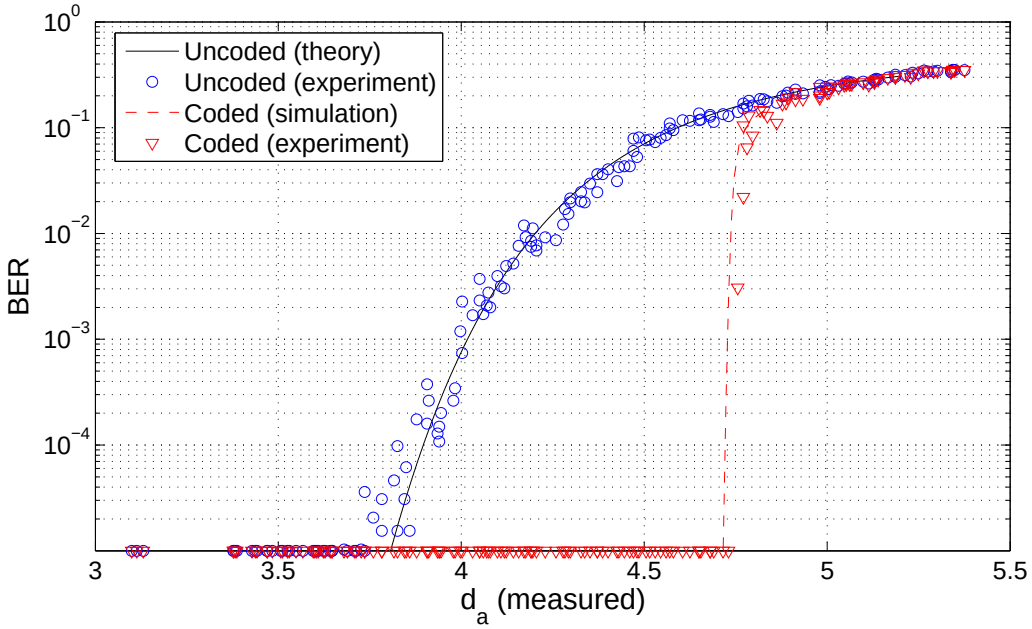


Figure 6.12: Experimental BER vs. d_a results for DVB-S2 scenario 4: $n = 16200$, $r = 1/2$, 50 max. decoding iterations.

transmissometer were used, the measured coding gains in d_a are likely to be lower than the actual value if $c(405\text{ nm})$ had been used. Thus, the range extension capabilities presented are conservative.

6.4 Conclusion

In this chapter we demonstrated the use of the DVB-S2 LDPC forward-error correction code on an underwater optical communication link operating at 405 nm. Experimental results were collected in a laboratory setting using an underwater optical communication testbed that employs intensity modulation/direct detection using on-off keying with return-to-zero line coding to transmit at a baud rate of 500 kbps. Based on simulation, four sets of DVB-S2 code parameters were chosen for experimentation, denoted by scenario 1 - 4. The first three scenarios were chosen to have a block length $n = 64800$, 100 maximum decoding iterations, and code rates of $r = 1/2$, $r = 1/3$, and $r = 1/4$ respectively. The fourth scenario was chosen to have block length $n = 16200$, 50 maximum decoding iterations, and a code rate of $r = 1/2$. This scenario offers more modest coding gains but in return provides a lower latency option suitable for future real-time implementation in a practical system.

Experimental results show very large coding gains that approach information-theoretic limits and are consistent with theory and simulation. The experimental coding gains in terms of E_b/N_0 at a BER of 10^{-4} were approximately 8.4 dB for scenario 1, 8.9 dB for scenario 2, 9.2 dB for scenario 3, and 7.7 dB for scenario 4. This performance is within 0.7 dB to 1.25 dB of the constellation-constrained capacity for OOK. Using attenuation coefficient values measured by a transmissometer during the experiments, we also calculated the experimental coding gains in terms of range extension, d_a. The experimental coding gains in units of attenuation lengths at a BER of 10^{-4} were approximately 0.79 for scenario 1, 1.04 for scenario 2, 1.06 for scenario 3, and 0.86 for scenario 4. For transmission through clear ocean water, these results suggest a potential range extension for the coded system of up to 5.22 m for scenario 1, 6.87 m for scenario 2, 7.00 m for scenario 3, and 5.68 m for scenario 4. These large coding gains can translate to a significant decrease in transmitter power necessary to transmit over a fixed distance and water turbidity. Alternatively, for a fixed transmitter power, they can enable extended link range for a given turbidity or the ability to communicate in more turbid waters for a fixed distance.

Chapter 7

Conclusions and Future Work

7.1 Conclusions

Underwater free-space optical communication is an emerging technology that shows the potential to offer high-data-rate, short range wireless communication in the underwater environment. In this thesis, we focused on the application of forward-error correction to an underwater FSO link to improve the performance of the system. Specifically, we demonstrated experimentally the implementation of Reed-Solomon coding, turbo coding, and low-density parity-check coding on an underwater FSO link using an underwater testbed in a laboratory setting. The experimental performance of each code in terms of code rate vs. performance in E_b/N_0 at a BER of 10^{-4} is summarized visually in Figure 7.1. The constellation-constrained capacity for OOK is also plotted for comparison. The first code investigated was the $\mathcal{C}_{RS}(255, 129)$ Reed-Solomon code. This code was implemented for its robustness and simplicity. We observed that this rate 1/2 code provided a coding gain of approximately 2.5 dB E_b/N_0 over the uncoded system at a BER of 10^{-4}.

Next, more modern, state-of-the-art turbo codes were implemented to provide even larger coding gains. Two standardized turbo codes were implemented, namely the UMTS turbo code and the CCSDS turbo code. It was shown that a coding gain of approximately 9.0 dB E_b/N_0 at a BER of 10^{-4} could be achieved using a rate 1/3 UMTS turbo code. A second rate 1/3 UMTS turbo code was demonstrated that provides a smaller coding gain of approximately 6.8 dB E_b/N_0 but in return provides lower latency and decreased decoder complexity. This code is presented as a potentially attractive option for future real-time

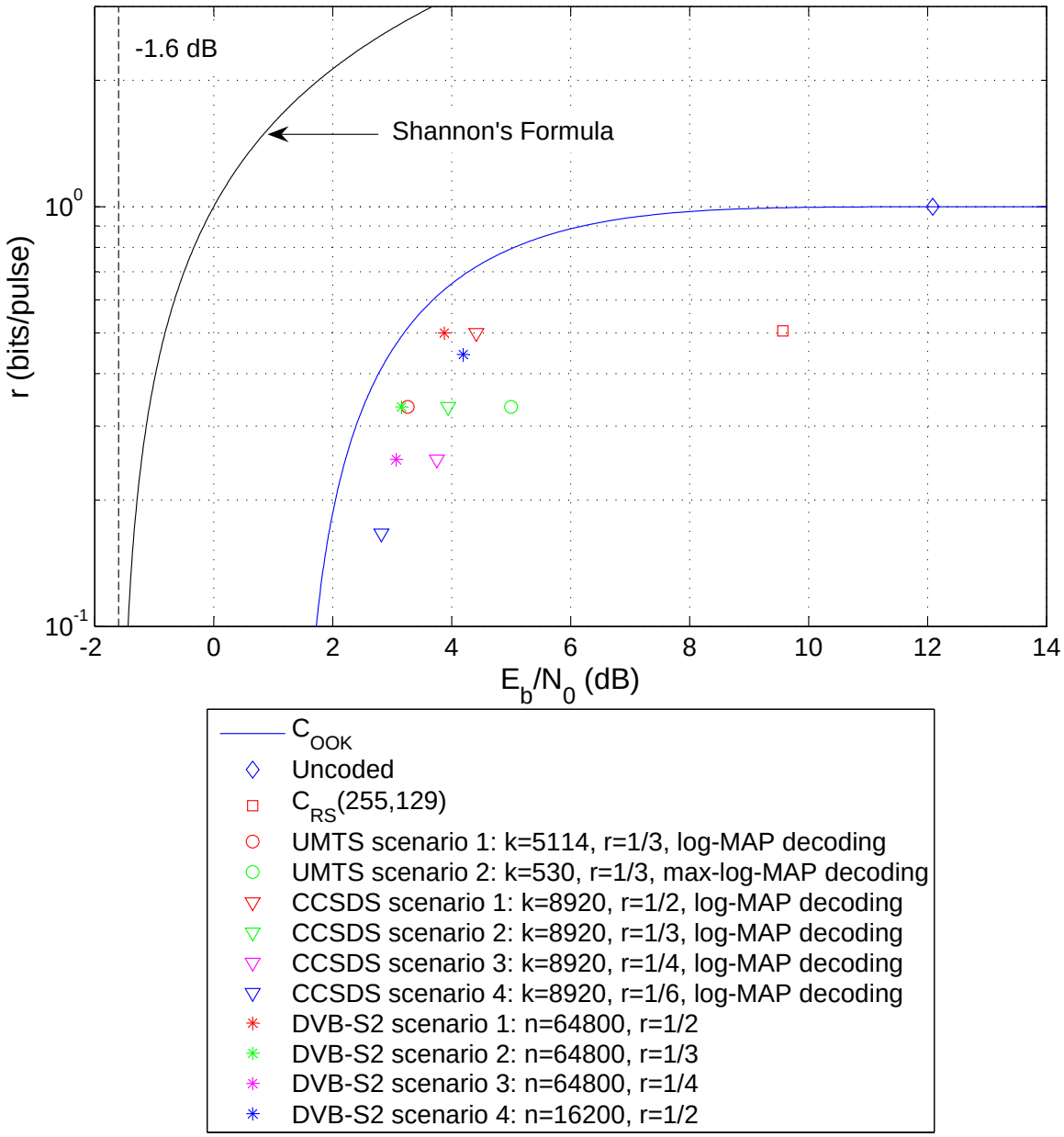


Figure 7.1: Summary of rates vs. experimentally measured E_b/N_0 at BER 10^{-4}.

implementation in a practical turbo-coded system. Next, CCSDS turbo codes with rates 1/2, 1/3, 1/4, and 1/6 were implemented. We observed that these codes provided coding gains in E_b/N_0 of approximately 7.7 dB, 8.1 dB, 8.3 dB, and 9.5 dB, respectively, over the uncoded system at a BER of 10^{-4}.

Finally, state-of-the-art LDPC codes were implemented which also provide large coding gains close to the information-theoretic limits. The LDPC code from the DVB-S2 standard was used. It was shown that the best performing DVB-S2 LDPC codes could provide coding gains in E_b/N_0 of approximately 8.4 dB, 8.9 dB, and 9.2 dB over the uncoded system at a BER of 10^{-4} for code rates of 1/2, 1/3, and 1/4, respectively. Additionally, we demonstrated a rate 1/2 LDPC code that provides a smaller coding gain of approximately 7.7 dB E_b/N_0, but in return provides a lower latency option that may be attractive for future real-time implementation in a practical LDPC coded system.

In addition to coding gains in E_b/N_0, we attempted to infer what these improvements in power efficiency could imply with regards to range extension. To do so, experimental measurements of the attenuation coefficient, $c(\lambda)$, were obtained using a transmissometer. Using this experimental data, the experimental coding gains were then observed in units of attenuation lengths. However, we faced two challenges in extrapolating the potential range extensions of the codes using this data. First, the transmissometer we used operates at $\lambda = 530$ nm, which is different from the underwater FSO link of the experimental testbed, which operates at $\lambda = 405$ nm. Second, the measurements were taken in a 3.66 m tank, and it is unclear whether these results can be directly extrapolated to much longer link ranges. With these two points in mind, we used measured attenuation coefficients from the transmissometer along with Beer's Law to predict the potential range extension of the coded systems in three water conditions. These results are summarized in Table 7.1.

The secondary focus of this thesis was to begin to establish a theoretical model for the underwater free-space optical channel. To establish a channel model, we first analyzed the statistical properties of the channel noise taken experimentally at very low SNR, when the received signal was almost entirely noise. From this simple analysis, we observed the channel noise to be approximately white and Gaussian. Based on this, an AWGN channel model was developed and used as a theoretical basis for comparison with experimental results. The fact that the experimental results presented in Chapters 4 through 6 agree

Table 7.1: Summary of range extension.

FEC Code	**Range Extension (m)**		
	Clear ocean ($c(\lambda) = 0.1514$ m^{-1})	**Coastal ocean** ($c(\lambda) = 0.399$ m^{-1})	**Harbor water** ($c(\lambda) = 2.195$ m^{-1})
$\mathcal{C}_{RS}(255, 129)$	2.61	0.99	0.18
UMTS scenario 1	7.57	2.87	0.52
UMTS scenario 2	5.89	2.24	0.41
CCSDS scenario 1	5.78	2.19	0.40
CCSDS scenario 2	6.18	2.34	0.43
CCSDS scenario 3	7.87	2.99	0.54
CCSDS scenario 4	7.99	3.03	0.55
DVB-S2 scenario 1	5.19	1.97	0.36
DVB-S2 scenario 2	6.89	2.61	0.48
DVB-S2 scenario 3	7.00	2.66	0.48
DVB-S2 scenario 4	5.69	2.16	0.39

so closely with the theory based on this channel model suggests that the AWGN channel model closely approximates the underwater FSO link, at least in this laboratory setting. To further support this theoretical channel model, we note that the LLRs used to initialize the turbo and LDPC decoders require knowledge of the channel statistics. If a theoretical channel model had been chosen that was statistically significantly different than the actual channel, it is unlikely that the turbo and LDPC coded results would align with theory. However, our experimental results agree closely with theory. We do observe, however, that there appears to be a systematic deviation of the uncoded data from the theory curve at high SNR that suggests a deviation from the AWGN channel model. This could be caused by a number of factors. It could be due to non-Gaussian sources of noise or the presence of weak fading due to turbulence. It could also be caused by the experimental setup itself, possibly due to quantization error at high signal levels or imprecise synchronization. Based on these experimental results, the cause remains unclear.

7.2 Future Work

Work is currently under way by other members of the group to implement the $\mathcal{C}_{RS}(255, 129)$ Reed-Solomon code in a real-time system. In Chapters 5 and 6, we presented two coding scenarios, one UMTS turbo code and one DVB-S2 LDPC code, in which the

parameters were chosen to facilitate real-time implementation. This was done by choosing smaller block sizes to decrease latency and, in the case of the UMTS turbo code, a simplified decoding algorithm to reduce decoder complexity. One future objective will be to implement real-time systems that use turbo or LDPC coding in a form factor that would allow experimentation outside the laboratory.

In the course of this research, we established an AWGN channel model and observed that this model is a good approximation to the underwater FSO link. However, it was observed that the experimental results deviate from theory at high SNR. An immediate future objective is to investigate this phenomenon and determine its cause. If the phenomenon is in fact due to the statistics of the underwater channel, and not an error introduced by the underwater testbed itself, this investigation could lead to a more robust theoretical model for the underwater FSO communication channel.

Finally, we observe that there are optics-based questions that have arisen during the course of this thesis which require further investigation. One topic of interest will be to investigate experimentally the relationship between the attenuation coefficient measured by the transmissometer at $\lambda = 530$ nm and the actual attenuation of the system at $\lambda = 405$ nm. Demonstrating a relationship between these values will be useful in determining accurate coding gains of the system in terms of range extension. Another, perhaps more significant, open question is whether the coding gains presented in this thesis can be extrapolated to longer ranges. A future objective will be to implement these codes over longer-range links to see if the gains in terms of range extension scale as predicted by Beer's Law.

Bibliography

[1] D. B. Kilfoyle and A. B. Baggeroer, "The state of the art in underwater acoustic telemetry," *IEEE Journal of Oceanic Engineering*, vol. 25, no. 1, pp. 4-27, Jan. 2000.

[2] M. Stojanovic, "Recent advances in high-speed underwater acoustic communications," *IEEE Journal of Oceanic Engineering*, vol. 21, no. 2, pp. 125-136, Apr. 1996.

[3] F. Hanson and S. Radic, "High bandwidth underwater optical communication," *Applied Optics*, vol. 47, no. 2, pp. 277-283, Jan. 2008.

[4] M. A. Chancey, "Short range underwater optical communication links," M.S. thesis, North Carolina State University, Raleigh, NC, 2005.

[5] B. Woodward and H. Sari, "Underwater speech communications with a modulated laser," *Applied Physics B: Lasers and Optics*, vol. 91, no. 1, pp. 189-194, Apr. 2008.

[6] I. Vasilescu, K. Kotay, D. Rus, M. Dunbabin and P. Corke, "Data collection, storage, and retrieval with an underwater sensor network," in *Proceedings of the Third International Conference on Embedded Networked Sensor Systems*, 2005, pp. 154-165.

[7] I. Talzi, A. Hasler, S. Gruber, and C. Tschudin, "Permasense: Investigating permafrost with a WSN in the swiss alps," in *Proceedings of the Fourth Workshop on Embedded Networked Sensors (Emnets IV)*, 2007, pp. 8-12.

[8] A. Nasipuri, K. Subramanian, V. Ogunro, J. Daniels, and H. Hilger, "Development of a wireless sensor network for monitoring a bioreactor landfill," in *Proceedings of GeoCongress 2006*, 2006, pp. 1-6.

[9] N. Farr, A. D. Chave, L. Freitag, J. Preisig, S. N. White, D. Yoerger and F. Sonnichsen, "Optical modem technology for seafloor observatories," in *OCEANS 2006*, 2006.

[10] J. Boutros, G. Caire, E. Viterbo, H. Saway, and S. Vialle, "Turbo code at 0.03 dB from capacity limit," in *Proceedings of the IEEE International Symposium on Information Theory*, 2002, p. 56.

[11] S.-Y. Chung, J. G. D. Forney, T. Richardson, and R. Urbanke, "On the design of low-density parity-check codes within 0.0045 dB of the Shannon limit," *IEEE Communications Letters*, vol. 5, pp. 5860, Feb. 2001.

[12] W. Cox, "A 1 Mbps underwater communication system using a 405 nm laser diode and photomultiplier tube," M.S. thesis, North Carolina State University, Raleigh, NC, 2007.

[13] J. Simpson, "A 1 Mbps underwater communications system using LEDs and photodiodes with signal processing capability," M.S. thesis, North Carolina State University, Raleigh, NC, 2007.

[14] C. E. Shannon, "A mathematical theory of communication," *Bell System Technical Journal*, vol. 27, pp. 379-423, 623-656, July and October 1948.

[15] S. Lin and D. J. Costello, *Error Control Coding: Fundamentals and applications*, 2nd ed. Upper Saddle River, NJ: Prentice-Hall, 2004.

[16] P. Elias, "Coding for noisy channels," in *IRE Convention Record*, vol. 3, pt. 4, 1955. Reprinted in *Key Papers in the Development of Coding Theory*, E. R. Berlekamp, ed. New York: IEEE Press, 1974.

[17] A. J. Viterbi, "Error bounds for convolutional codes and asymptotically optimum decoding algorithm," *IEEE Transactions on Information Theory*, vol. IT-13, no. 2, pp. 260-269, Apr. 1967.

[18] G. D. Forney, Jr., "The Viterbi algorithm," *Proceedings of the IEEE*, vol. 61, no. 3, pp. 268-278, Mar. 1973.

[19] G. D. Forney, *Concatenated codes*. Cambridge, MA: MIT Press, 1966.

[20] C. Berrou, A. Glavieux and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo-codes," in *Proceedings of the IEEE International Conference on Communications*, 1993, pp. 1064-1070.

[21] I. S. Reed and G. Solomon, "Polynomial codes over certain finite fields," *Journal of the Society for Industrial and Applied Mathematics*, vol. 8, no. 2, pp. 300-304, June 1960.

[22] J. C. Moreira and P. G. Farrell, *Essentials of Error-Control Coding*, West Sussex, UK: John Wiley & Sons Ltd., 2006.

[23] M. C. Valenti and J. Sun, "Turbo codes," in *Handbook of RF and Wireless Technologies*, F. Dowla, Ed. Oxford, UK: Elsevier, 2004.

[24] J. Hagenauer and P. Hoeher, "A Viterbi algorithm with soft-decision outputs and its applications," in *Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM)*, 1989, pp. 16801686.

[25] L. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Transactions on Information Theory*, vol. 20, pp. 284287, Mar. 1974.

[26] A. J.Viterbi, "An intuitive justification and simplified implementation of the MAP decoder for convolutional codes," *IEEE Journal on Selected Areas of Communication*, vol. 16, no. 2, pp. 260-264, Feb. 1998.

[27] M. C. Valenti and J. Sun, "The UMTS turbo code and an efficient decoder implementation suitable for software defined radios," *International Journal on Wireless Information Networks*, vol. 8, no. 4, pp. 203-216, Oct. 2001.

[28] R. G. Gallager, "Low-density parity-check codes," *IRE Transactions on Information Theory*, vol. IT-8, no. 1, pp. 21-28, Jan. 1962.

[29] R. G. Gallager, "Low density parity check codes," Sc.D. dissertation, Massachusetts Institute of Technology, Cambridge, MA, 1960.

[30] D. J. C. MacKay, R. M. Neal, "Near Shannon limit performance of low density parity check codes," *Electronics Letters*, vol. 32, no. 18, pp. 1645-1646, Aug. 1996.

[31] D. J. MacKay, *Information Theory, Inference, and Learning Algorithms*, 3rd ed. Cambridge, UK: Cambridge University Press, 2004. [Online] Available: http://www.inference.phy.cam.ac.uk/mackay/itila/book.html. [Accessed: Nov. 17, 2009].

[32] F. Kschischang, B. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Transactions on Information Theory*, vol. 47, pp. 498519, Feb. 2001.

[33] L. M. Tanner, "A recursive approach to low complexity codes," *IEEE Transactions on Information Theory*, vol. 27, no. 5, pp. 533-547, Sept. 1981.

[34] S. G. Lambert and W. Casey, *Laser Communication in Space*. Norwood, MA: Artech House Inc., 1995, pp. 347-348.

[35] M. A. Blizard, "Ocean optics: Introduction and overview," in *Ocean Optics VIII*, 1986, pp. 2-17.

[36] R. E. Walker, *Marine Light Field Statistics*. New York: John Wiley & Sons Inc., 1994, pp. 142-163.

[37] Y. J. Gawdi, "Underwater free space optics," M.S. thesis, North Carolina State University, Raleigh, NC, 2006.

[38] N. G. Jerlov, *Marine Optics*, 2nd ed. New York: Elsevier Scientific Publishing Co., 1976, p. 58.

[39] K. S Shifrin, *Physical Optics of Ocean Water*. New York: American Institute of Physics, 1988.

[40] R. C. Smith and K. S. Baker, "Optical properties of the clearest natural waters (200 800 nm)," *Applied Optics*, vol. 20, no. 2, pp. 177-184, Jan. 1981.

[41] H. Willebrand and B. S. Ghuman, *Free-Space Optics: Enabling optical connectivity in today's networks*. Indianapolis, IN: Sams Publishing, 2002, pp. 48-53.

[42] B. Cochenour, L. Mullen, A. Laux and T. Curran, "Effects of multiple scattering on the implementation of an underwater wireless optical communications link," in *OCEANS 2006*, 2006.

[43] B. Cochenour, L. Mullen, and A. Laux, "Characterization of the beam-spread function for underwater wireless optical communications links," *IEEE Journal of Oceanic Engineering*, vol. 33, no. 4, pp. 513-521, Oct. 2008.

[44] W. C. Cox, J. A. Simpson, C. P. Domizioli, J. F. Muth and B. L. Hughes, "An underwater optical communication system implementing reed-solomon channel coding," in *Oceans 2008*, 2008.

[45] S. Duntley, "Underwater Lighting by Submerged Lasers and Incandescent Sources," Scripps Institution of Oceanography Visibility Laboratory, San Diego, CA, SIO Ref. 71-1, June 1971, pp. 143-180.

[46] T. J. Petzold, "Volume scattering functions for selected ocean waters," Scripps Institute of Oceanography Visibility Laboratory, San Diego, CA, SIO Ref. 72-78, Oct. 1972.

[47] A. Laux, R. Billmers, L. Mullen, B. Concannon, J. Davis, J. Prentice and V. Contarino, "The *a, b, c*s of oceanographic lidar predictions: a significant step toward closing the loop between theory and experiment," *Journal of Modern Optics*, vol. 49, no. 3, pp. 439-451, Mar. 2002.

[48] WETLabs, C-Star manual. [Online] Available: http://www.wetlabs.com. [Accessed: Nov. 17, 2009].

[49] X. Zhu and J. M. Kahn, "Free-space optical communication through atmospheric turbulence channels," *IEEE Transactions on Commununications*, vol. 50, pp. 1293-1300, Aug. 2002.

[50] R. M. Gagliardi and C. M. Thomas, "PCM data reliability monitoring through estimation of signal-to-noise ratio," *IEEE Transactions on Communications*, vol. COM-16, pp. 479-486, June 1968.

[51] J. G. Proakis, *Digital Communications*, 4th ed. Boston: McGraw-Hill, 2001.

[52] M. C. Jeruchim, "Techniques for estimating the bit error rate in the simulation of digital communication systems," *IEEE Journal on Selected Areas in Communications*, vol. 2, no. 1, pp. 153-170, Jan. 1984.

[53] European Telecommunications Standards Institute, "Universal mobile telecommunications system (UMTS); Multiplexing and channel coding (FDD)," 3GPP TS 25.212 version 8.3.0 release 8, pp. 15-22, Sept. 24, 2008.

[54] T. Iliev, "A study of turbo codes for UMTS third generation cellular standard," *Proceedings of the CompSysTech 2007*, 2007, pp. VI.6-1 VI.6-6.

[55] M. Valenti, "Iterative Solutions Coded Modulation Library (ISCML)," version 1.10, May 23, 2008. [Online] Available: http://www.iterativesolutions.com. [Accessed: Nov. 17, 2009].

[56] Consultative Committee for Space Data Systems, "Recommendations for space data system standards: TM synchronization and channel coding," CCSDS 131.0-B-1 Blue book, pp. 31-38, Sept. 2003.

[57] S. Dolinar, D. Divsalar and F. Pollara, "Turbo codes and space communications," in *Proceeding of the Fifth International Symposium on Space Mission Operations and Ground Data System: SpaceOps '98*, 1998, pp. 1-8. [Online] Available: http://track.sfo.jaxa.jp/spaceops98/paper98/track5/5e011.pdf [Accessed: Nov. 17, 2009].

[58] European Telecommunications Standards Institute, "Digital video broadcasting (DVB); Second generation framing structure, channel coding and modulation systems for broadcasting, interactive services, news gathering and other broadband satellite applications (DVB-S2)," ETSI EN 302.307 version 1.2.1, pp. 20-25, 37-49, Apr. 2009.

[59] M. Eroz, F.-W. Sun, L.-N. Lee, "DVB-S2 low density parity check codes with near Shannon limit performance," *International Journal on Satellite Communication Networks*, vol. 22, no. 3, May-June 2004.

[60] A. Morello and V. Mignone, "DVB-S2: The second generation standard for satellite broad-band services," *Proceedings of the IEEE*, vol. 94, no. 1, Jan. 2006.

[61] J. Chen, A. Dholakia, E. Eleftheriou, M. P. C. Fossorier, and X.-Y. Hu, "Reduced-complexity decoding of LDPC codes," *IEEE Trans. on Communications*, vol.53, no.8, pp. 1288-1299, Aug. 2005.

[62] F. R. Kschischang and B. J. Frey, "Iterative decoding of compound codes by probability propagation in graphical models," *Journal on Selected Areas in Communications*, vol. 16, no.2, pp.219-230, Feb. 1998.

Appendix

Appendix A

Plots of Experimental Attenuation Coefficient vs. SNR

In order to establish an empirical relationship between measured attenuation coefficient, $c(530\text{ nm})$, and estimated SNR, plots are generated of $c(530\text{ nm})$ vs. SNR for each experiment. A linear regression is then calculated using MATLAB for each experiment. To avoid skewed data, outliers are excluded and the robust regression option is used. The resulting regression lines are summarized numerically in the table below. The slope of the line is denoted by m, and the y-intercept is denoted by b.

Table A.1: Summary of linear regression data.

Experiment	m	m (upper)	m (lower)	b	b (upper)	b (lower)
$\mathcal{C}_{RS}(255, 129)$	-0.0201	-0.0203	-0.0199	1.55	1.55	1.56
UMTS scenario 1	-0.0205	-0.0207	-0.0203	1.42	1.42	1.42
UMTS scenario 2	-0.0202	-0.0204	-0.0199	1.22	1.21	1.22
CCSDS scenario 1	-0.0197	-0.0200	-0.0194	1.26	1.26	1.26
CCSDS scenario 2	-0.0207	-0.0209	-0.0205	1.51	1.51	1.51
CCSDS scenario 3	-0.0213	-0.0215	-0.0211	1.57	1.57	1.58
CCSDS scenario 4	-0.0193	-0.0195	-0.0191	1.59	1.59	1.59
DVB-S2 scenario 1	-0.0192	-0.0196	-0.0188	1.39	1.38	1.39
DVB-S2 scenario 2	-0.0200	-0.0203	-0.0197	1.38	1.38	1.39
DVB-S2 scenario 3	-0.0194	-0.0196	-0.0192	1.59	1.58	1.59
DVB-S2 scenario 4	-0.0205	-0.0208	-0.0203	1.36	1.36	1.36

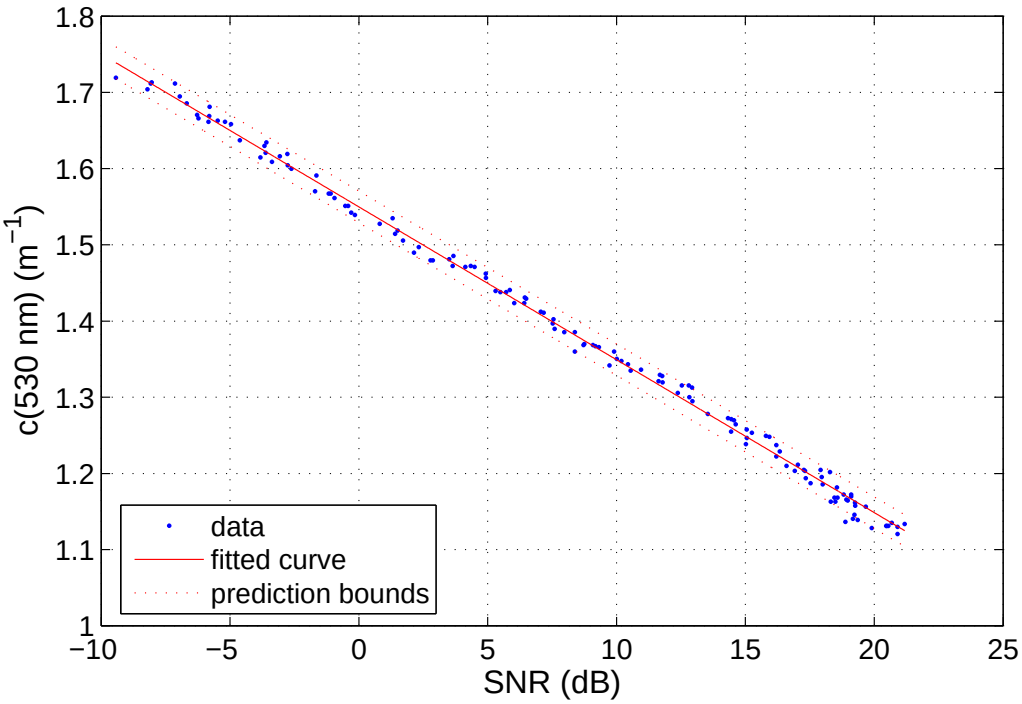


Figure A.1: Plot of $c(530\text{ nm})$ vs. SNR and linear regression for $\mathcal{C}_{RS}(255, 129)$.

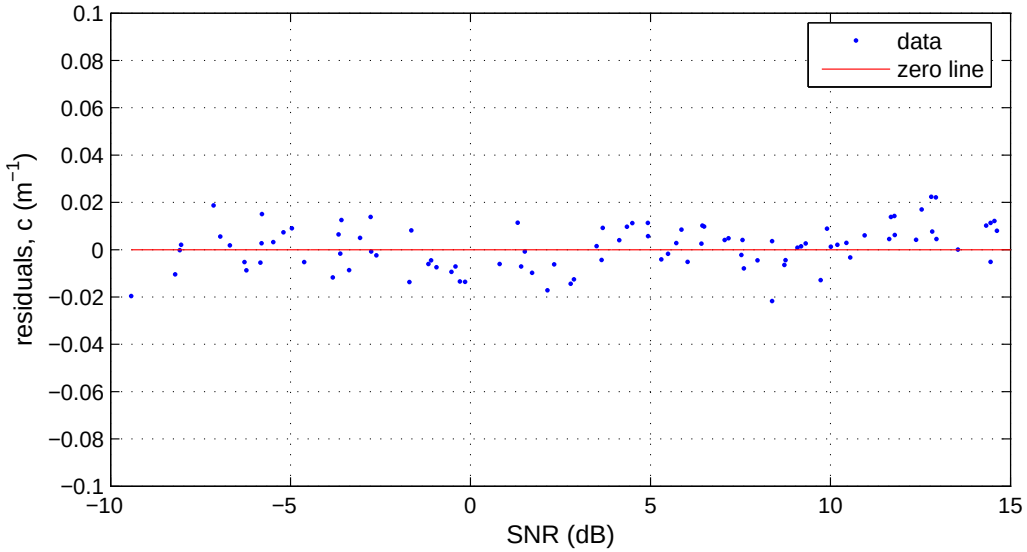


Figure A.2: Plot of residuals for $\mathcal{C}_{RS}(255, 129)$.

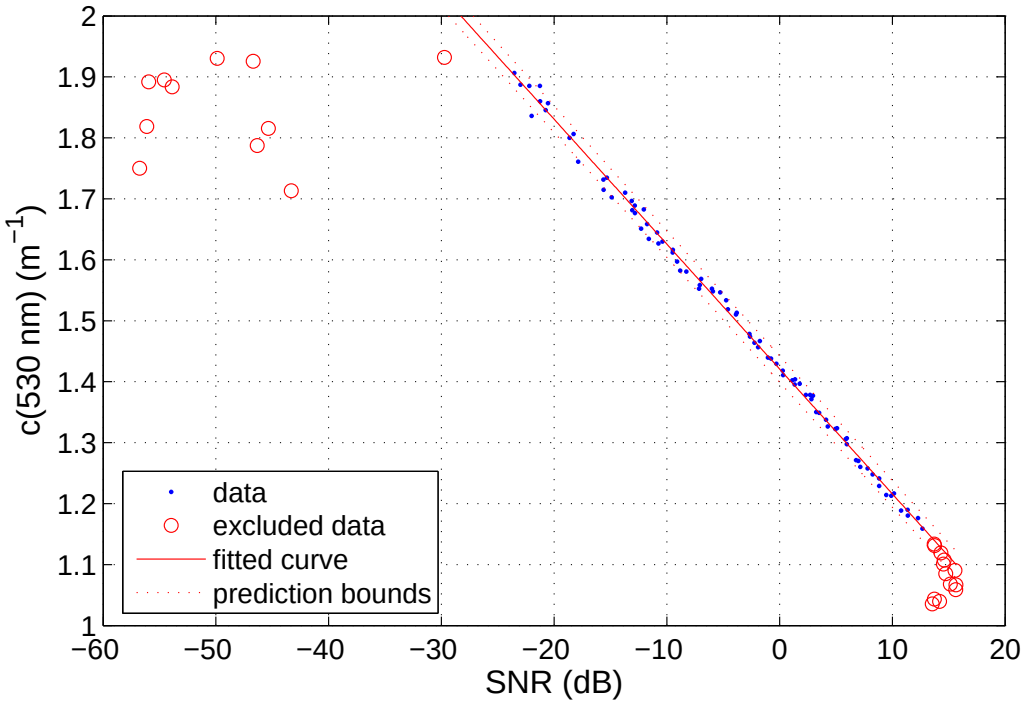


Figure A.3: Plot of $c(530\text{ nm})$ vs. SNR and linear regression for UMTS scenario 1.

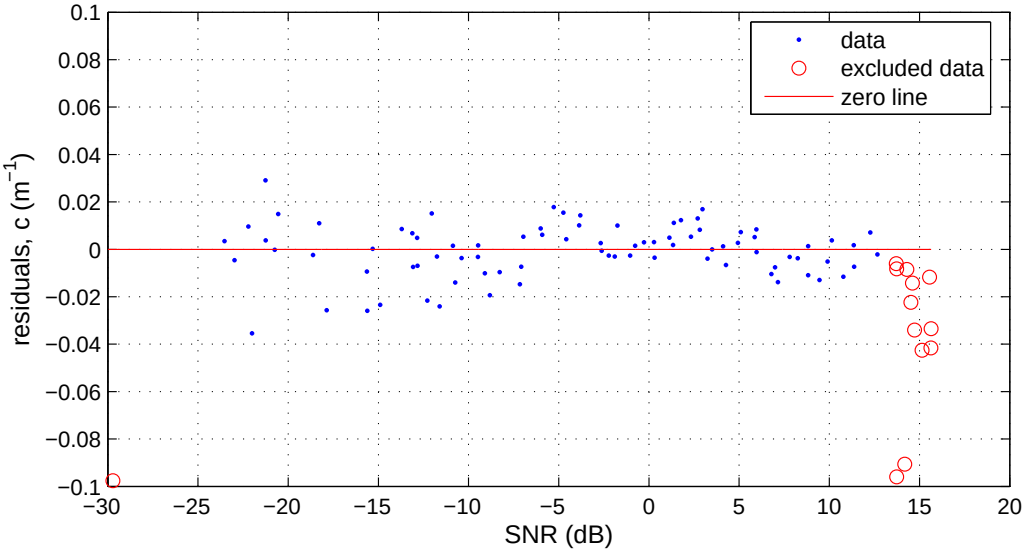


Figure A.4: Plot of residuals for UMTS scenario 1.

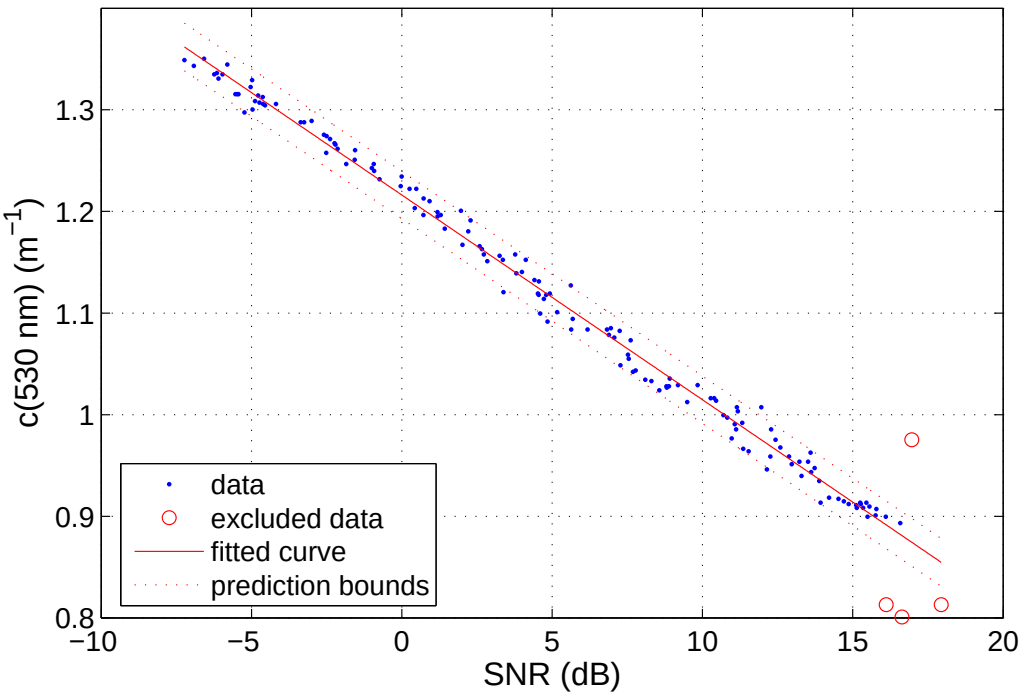


Figure A.5: Plot of $c(530\text{ nm})$ vs. SNR and linear regression for UMTS scenario 2.

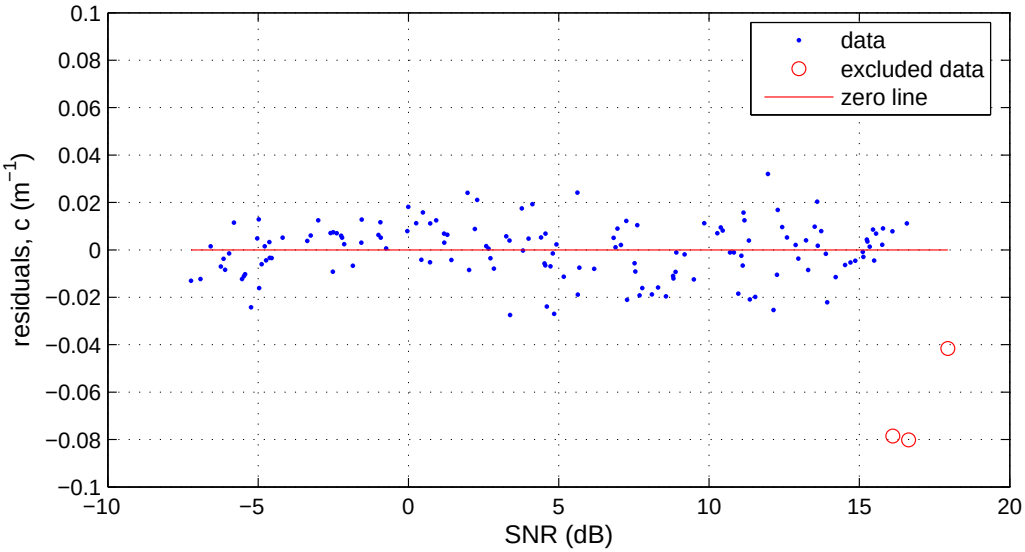


Figure A.6: Plot of residuals for UMTS scenario 2.

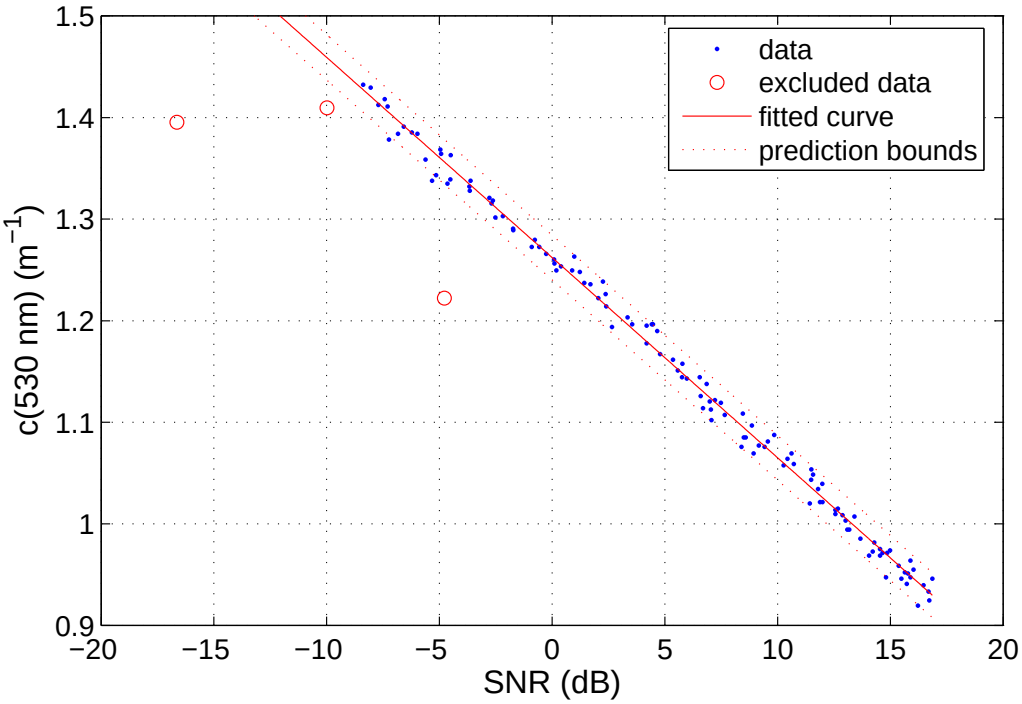


Figure A.7: Plot of $c(530\text{ nm})$ vs. SNR and linear regression for CCSDS scenario 1.

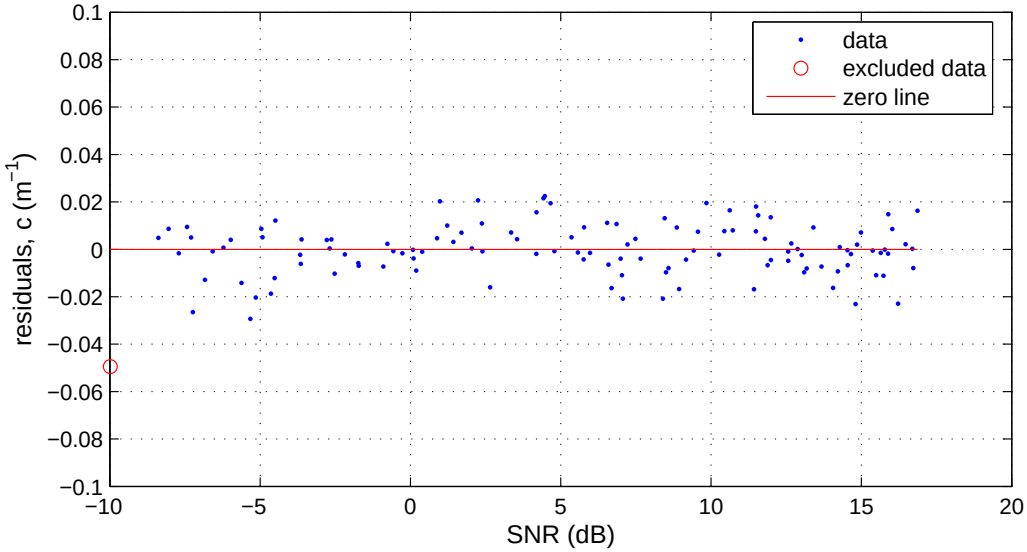


Figure A.8: Plot of residuals for CCSDS scenario 1.

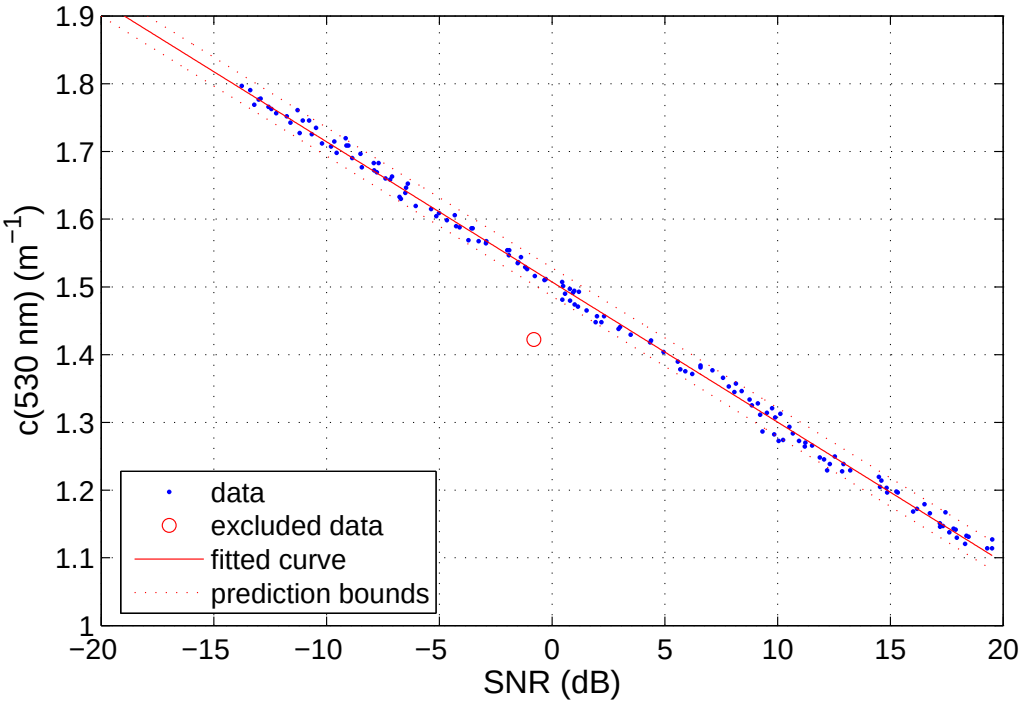


Figure A.9: Plot of $c(530\text{ nm})$ vs. SNR and linear regression for CCSDS scenario 2.

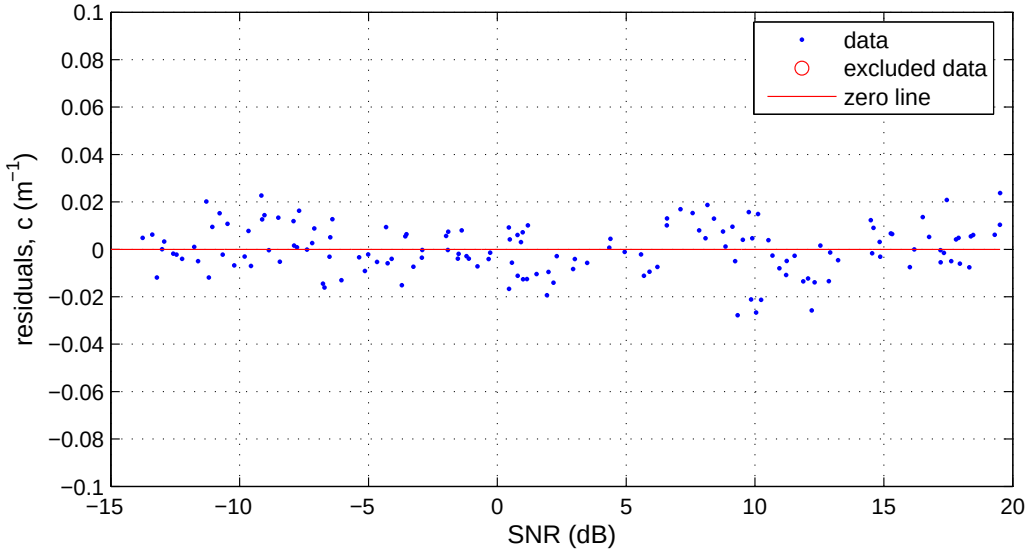


Figure A.10: Plot of residuals for CCSDS scenario 2.

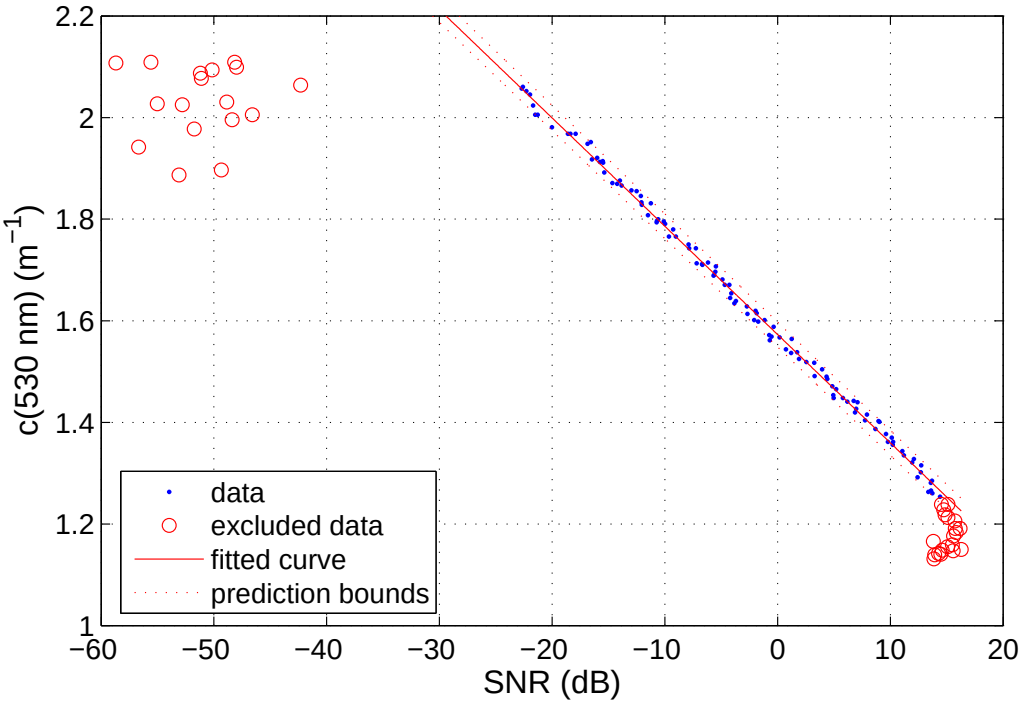


Figure A.11: Plot of $c(530\text{ nm})$ vs. SNR and linear regression for CCSDS scenario 3.

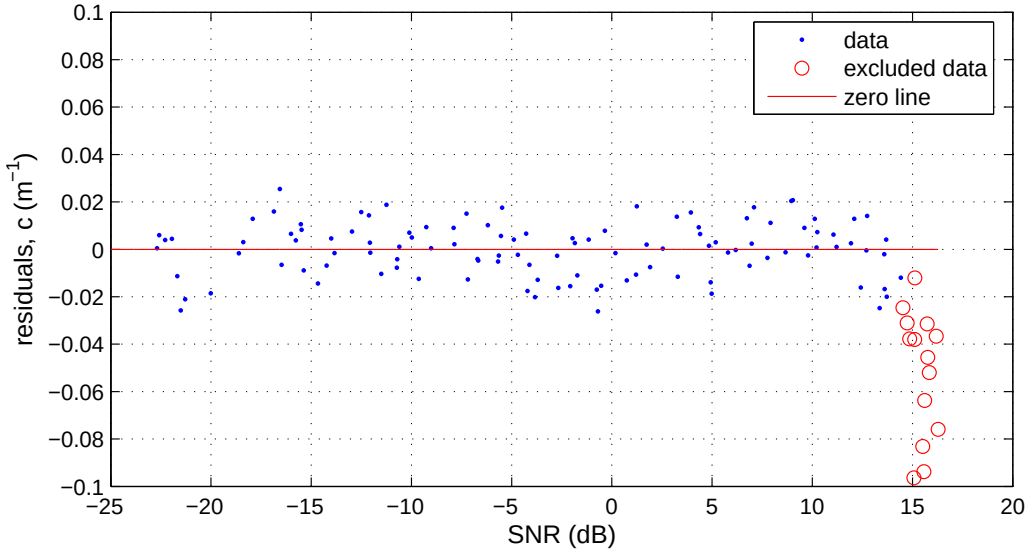


Figure A.12: Plot of residuals for CCSDS scenario 3.

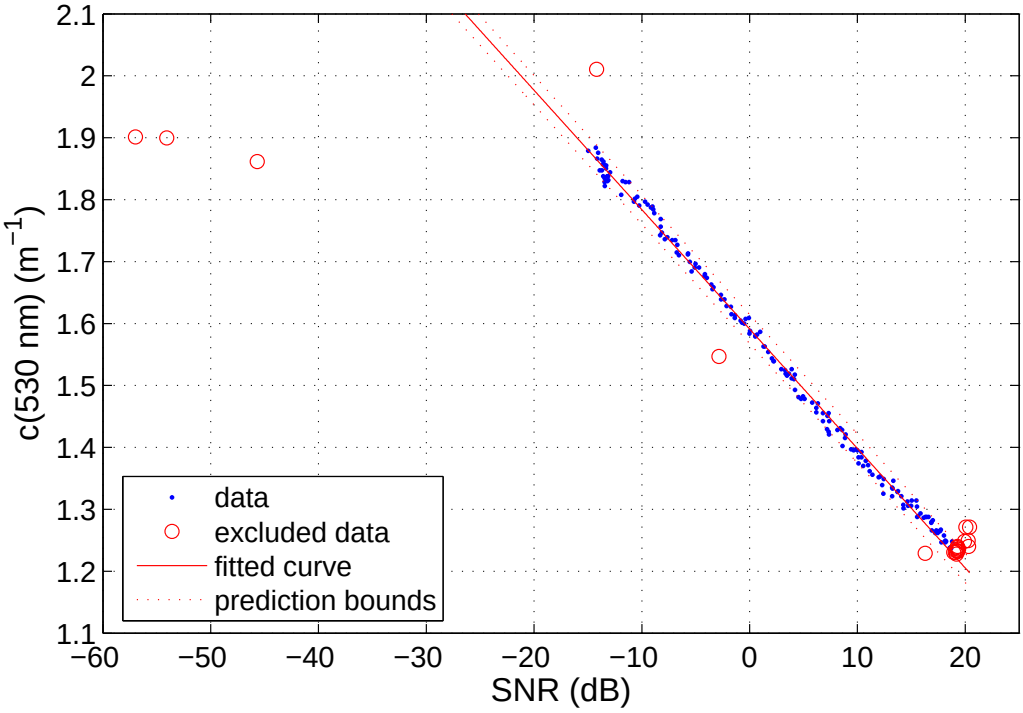


Figure A.13: Plot of $c(530\text{ nm})$ vs. SNR and linear regression for CCSDS scenario 4.

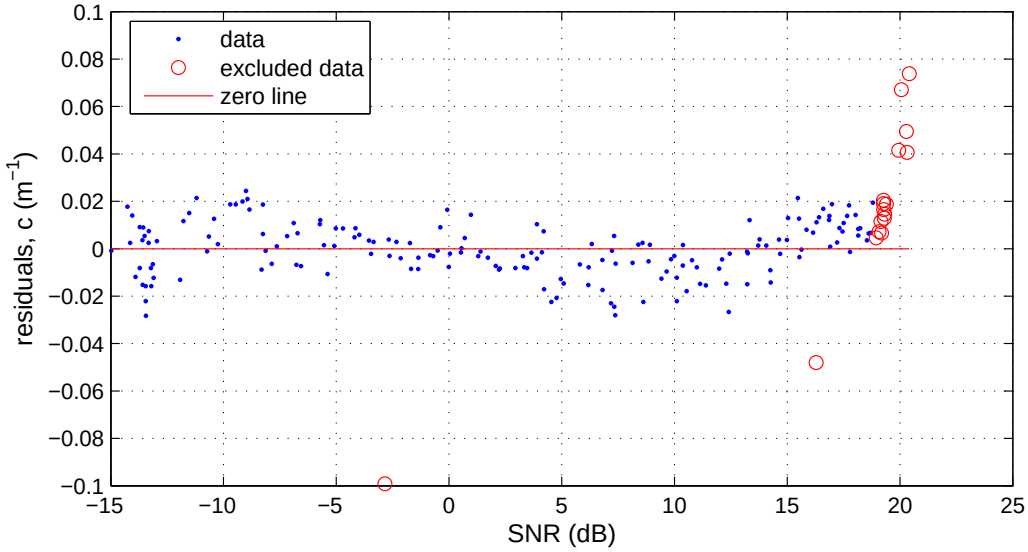


Figure A.14: Plot of residuals for CCSDS scenario 4.

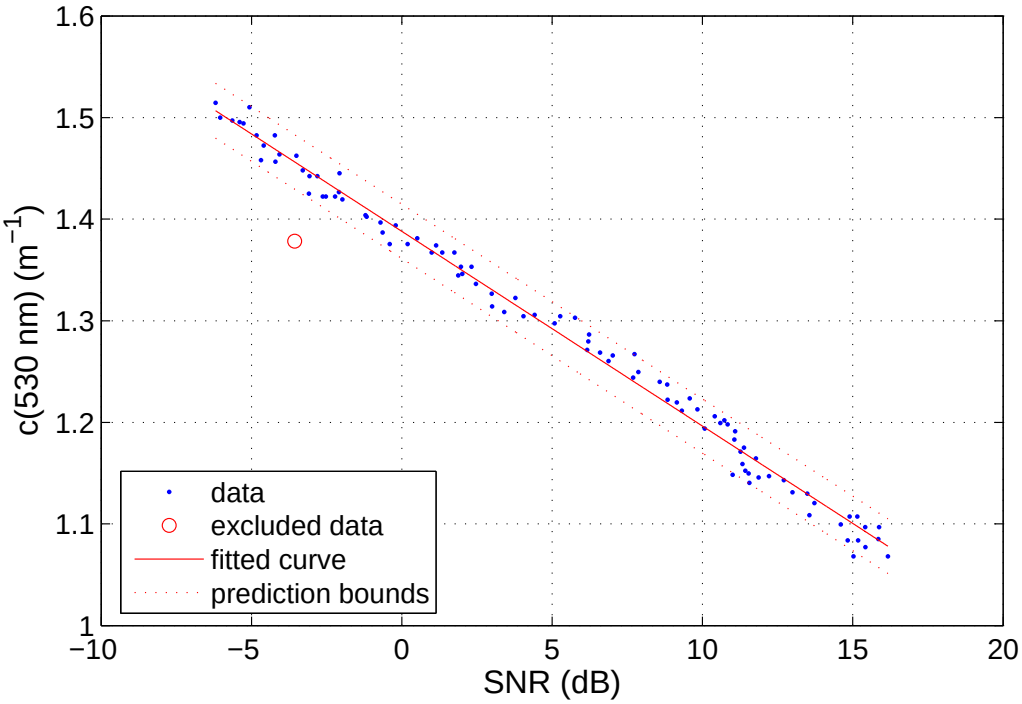


Figure A.15: Plot of $c(530\text{ nm})$ vs. SNR and linear regression for DVB-S2 scenario 1.

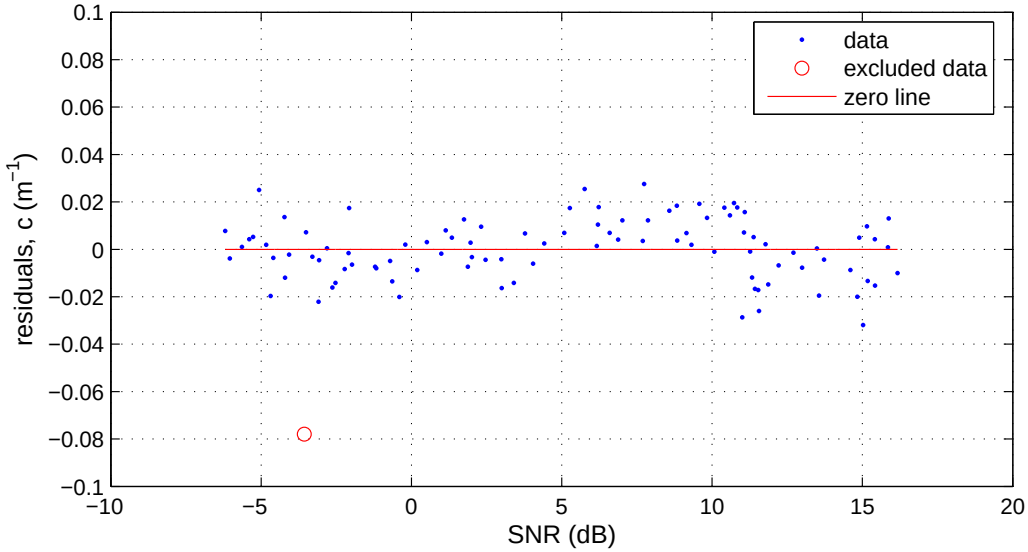


Figure A.16: Plot of residuals for DVB-S2 scenario 1.

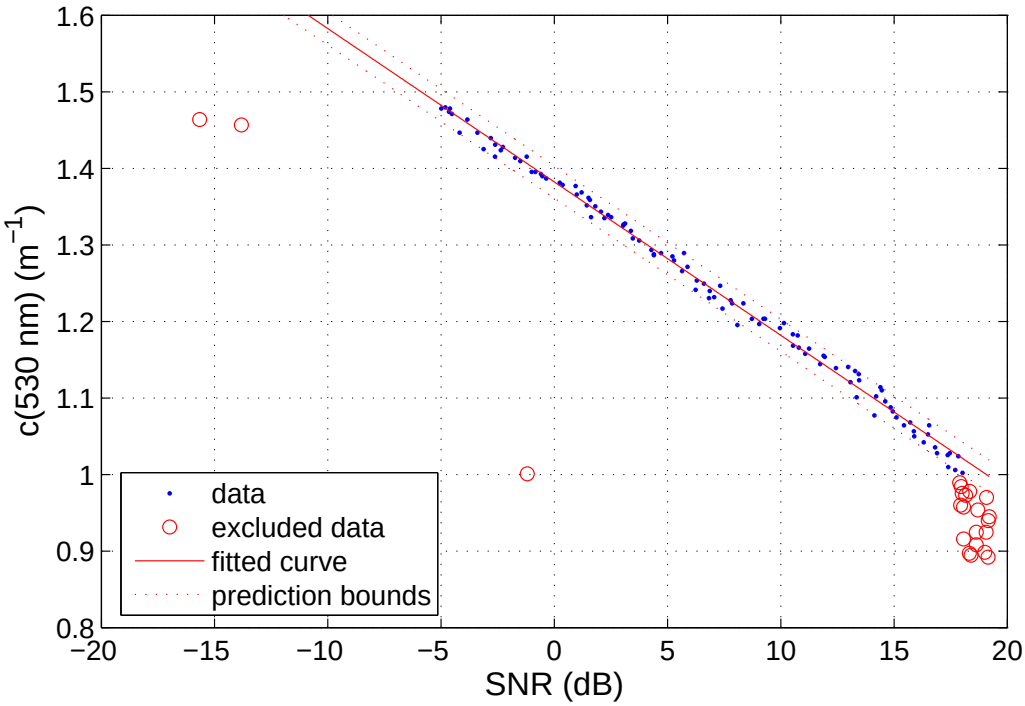


Figure A.17: Plot of $c(530\text{ nm})$ vs. SNR and linear regression for DVB-S2 scenario 2.

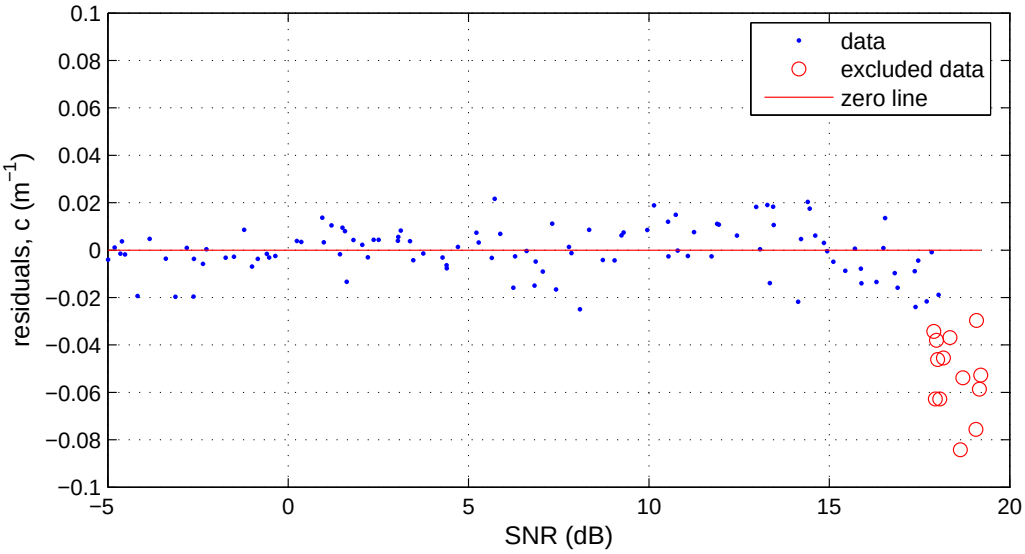


Figure A.18: Plot of residuals for DVB-S2 scenario 2.

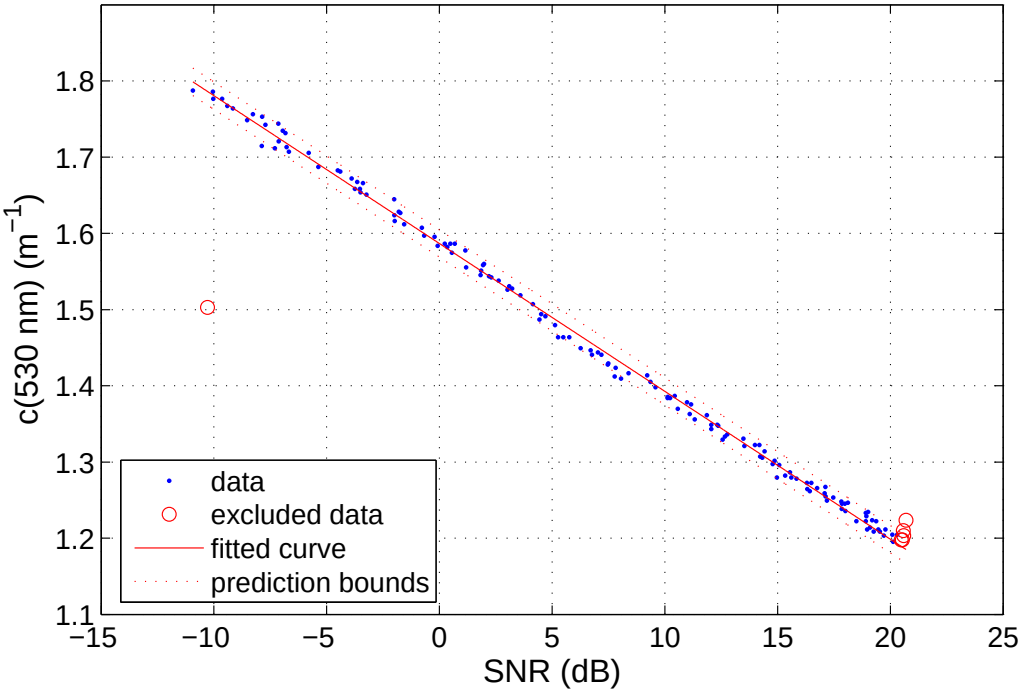


Figure A.19: Plot of $c(530\text{ nm})$ vs. SNR and linear regression for DVB-S2 scenario 3.

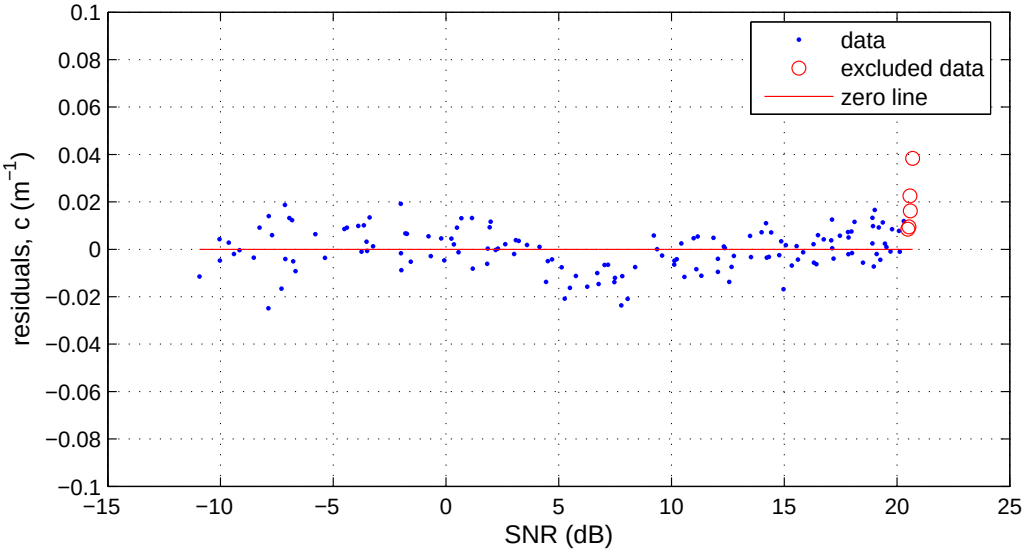


Figure A.20: Plot of residuals for DVB-S2 scenario 3.

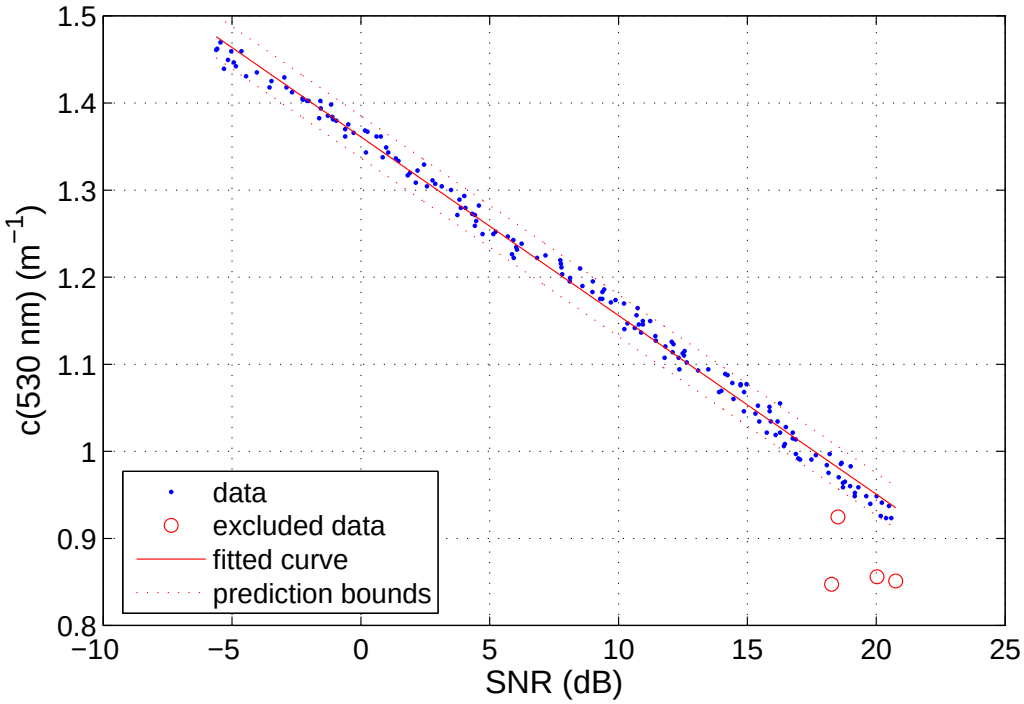


Figure A.21: Plot of $c(530\text{ nm})$ vs. SNR and linear regression for DVB-S2 scenario 4.

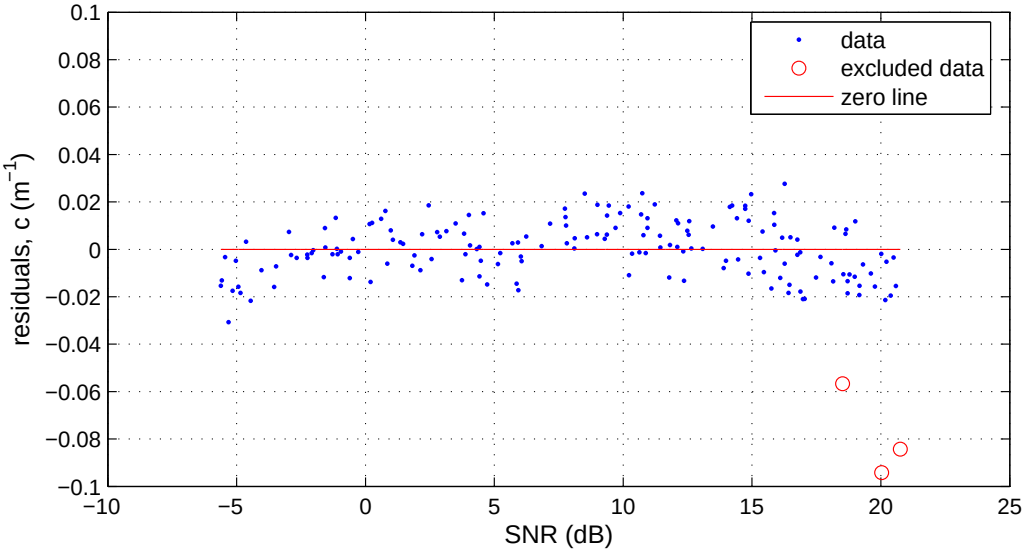


Figure A.22: Plot of residuals for DVB-S2 scenario 4.