View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by AFTI Scholar (Air Force Institute of Technology)

Air Force Institute of Technology

AFIT Scholar

Theses and Dissertations

Student Graduate Works

3-21-2013

Improving Bandwidth Utilization in a 1 Tbps Airborne MIMO Communications Downlink

Jonathan D. Hill

Follow this and additional works at: https://scholar.afit.edu/etd

 Part of the Electrical and Computer Engineering Commons

Recommended Citation

Hill, Jonathan D., "Improving Bandwidth Utilization in a 1 Tbps Airborne MIMO Communications Downlink" (2013). *Theses and Dissertations*. 876.
https://scholar.afit.edu/etd/876

This Thesis is brought to you for free and open access by the Student Graduate Works at AFIT Scholar. It has been accepted for inclusion in Theses and Dissertations by an authorized administrator of AFIT Scholar. For more information, please contact richard.mansfield@afit.edu.

IMPROVING BANDWIDTH UTILIZATION IN A 1 TBPS AIRBORNE MIMO COMMUNICATIONS DOWNLINK

THESIS

Jonathan D. Hill, Captain, USAF

AFIT-ENG-13-M-25

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

DISTRIBUTION STATEMENT A.
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the United States Air Force, the Department of Defense, or the United States Government.

This material is declared a work of the U.S. Government and is not subject to copyright protection in the United States.

AFIT-ENG-13-M-25

IMPROVING BANDWIDTH UTILIZATION IN A 1 TBPS AIRBORNE MIMO COMMUNICATIONS DOWNLINK

THESIS

Presented to the Faculty

Department of Electrical and Computer Engineering

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

in Partial Fulfillment of the Requirements for the

Degree of Master of Science in Electrical Engineering

Jonathan D. Hill, B.S.E.E.

Captain, USAF

March 2013

DISTRIBUTION STATEMENT A.
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

AFIT-ENG-13-M-25

IMPROVING BANDWIDTH UTILIZATION IN A 1 TBPS AIRBORNE MIMO COMMUNICATIONS DOWNLINK

Jonathan D. Hill, B.S.E.E.
Captain, USAF

Approved:

/signed/	8 Mar 2013
Richard K. Martin, PhD (Chairman)	Date
/signed/	15 Mar 2013
LtCol James A. Louthain, PhD (Member)	Date
/signed/	14 Mar 2013
Maj Mark D. Silvius, PhD (Member)	Date

AFIT-ENG-13-M-25

Abstract

Forward Error Correction (FEC) techniques are compared for different Multiple-Input Multiple-Output (MIMO) configurations of a high altitude, extremely wide bandwidth radio frequency downlink. Monte Carlo simulations are completed in MATLAB® with the aim of isolating the impacts of turbo codes and Low-Density Parity Check (LDPC) codes on system throughput and error performance. The system is modeled as a transmit-only static array at an altitude of 60,000 feet, with no interferers in the channel. Transmissions are received by a static receiver array. Simulations attempt to determine what modulation types should be considered for practical implementation, and what FEC codes enable these modulation schemes. The antenna configurations used in this study are [44:352], [62:248], and [80:160] transmitters to receivers. Effects from waveform generation, mixing, down-conversion, and amplification are not considered.

Criteria of interest were Bit-Error Rate (BER) and throughput, with the maximum allowable value of the former set at 1×10^{-5}, and the latter set at a 1 terabits per second (Tbps) transfer rate for a successful configuration. Results show that the best performing system configuration was unable to meet both criteria, but was capable of improving over Brueggen's 2012 research, which used Reed-Solomon codes and a MIMO configuration of [80:160], by 18.6%. The best-case configuration produced a throughput rate of 0.83 Tbps at a BER of less than 1×10^{-8}, by implementing a rate ⅔ LDPC code with Quadrature Amplitude Modulation (QAM) constellation of 16 symbols.

Acknowledgments

Anyone who worked with me at any time during the process of completing this program or producing this document, even in the form in now presently exists, knows I would not have made it without their constant support, and a serious amount of Divine intervention. With this in mind, I would like to thank my parents, friends, instructors, and especially Dr. Martin for helping me through. There is an old saying that goes something like this: "Whoever ignores instruction despises himself, but he who listens to reproof gains intelligence." To everyone who has challenged me to grow, whether inside or outside of academia, thank you.

Jonathan D. Hill

Table of Contents

Page

Page

List of Figures

Figure Page

List of Tables

Table Page

List of Symbols

Symbol	Definition
$\aleph$	state metric scaling factor
α	forward state metric
β	reverse state metric
δ	branch metric
Λ	likelihood ratio
π	branch probability ratio

Subscripts

k	current time sample
K	final time sample
r	index for reading

Superscripts

a	a priori
b	previous state function
c	channel
e	extrinsic
f	next state function
i	next state branch metric input (1 or 0)
j	previous state branch metric input (1 or 0)
s	current encoder state
S	maximum state value

List of Acronyms

Acronym	Definition
AWGN	Additive White Gaussian Noise
BPSK	Binary Phase-Shift Key
BER	Bit-Error Rate
BCH	Bose-Chadhuri-Hocquenghem
BCJR	Modified Bahl, Cocke, Jelinek, and Raviv
BLAST	Bell Labs Layered Space-Time
DoD	Department of Defense
DVB-S2	Second Generation Digital Video Broadcasting Standard
EIRP	Effective Isotropic Radiated Power
FEC	Forward Error Correction
ICD	Inverse Channel Detection
IIR	Infinite Impulse Response
IEEE	Institute of Electrical and Electronics Engineers
LR	Likelihood Ratio
LLR	Log-Likelihood Ratio
LDPC	Low-Density Parity Check
LOS	Line-of-Sight
MATLAB®	Matrix Laboratory
MIMO	Multiple-Input Multiple-Output
MISO	Multiple-Input Single-Output
MMSE	Minimum Mean Squared Error
MAP	Maximum A Posteriori
ML	Maximum Likelihood

Acronym	Definition
NASA	National Aeronautics and Space Administration
NSC	Nonsystematic Convolutional
PSK	Phase-Shift Key
QAM	Quadrature Amplitude Modulation
R-S	Reed-Solomon
RSC	Recursive Systematic Convolutional
SVD	Singular Value Decomposition
SISO	Single-Input Single-Output
SIMO	Single-Input Multiple-Output
SOVA	Soft-Output Viterbi Algorithm
SNR	Signal-to-Noise Ratio
TDL	Tapped Delay Line
USAF	United States Air Force

IMPROVING BANDWIDTH UTILIZATION IN A 1 TBPS
AIRBORNE MIMO COMMUNICATIONS DOWNLINK

I. Introduction and Problem Statement

DIGITAL communication systems are fundamental to the everyday function of government, military, commercial and civilian life. Nearly every person living in the United States makes use of a device built on digital communication principles or technology. The wireless technology in cellular phones has, in the past twenty years, advanced from single-antenna, analog wave forms, to present day multi-carrier Multiple-Input Multiple-Output (MIMO) arrays of antennas. The U. S. Air Force relies on many of these technological advances to conduct missions, collect and transmit intelligence information, and to tend to day-to-day operations. This chapter seeks to introduce a single facet of these activities as it applies to the mission of the Air Force and United States Department of Defense (DoD) — the wireless transmission of information — as well as define a current issue in this arena along with a path to a solution.

1.1 Motivation

In late 2009, the then head of Air Force intelligence, Lt. Gen. Deptula, is quoted as saying that the United States Air Force (USAF) would find itself "swimming in sensors and drowning in data" in the near future. Then acting Deputy Undersecretary of Defense for programs and resources, Kevin Meiners, relayed that accounting for the influx of high-definition video would likely cause challenges to existing communications architectures [1]. These comments only illustrate what many official have known for a long time: USAF and the rest of the DoD face a growing challenge of collecting, processing, and transferring

information from various, and geographically dispersed mission elements. One of the major benefits of technology is also an obstacle in this regard: the increase in both the quantity and quality of available information, ranging from sensor data to command and control, has created an incredible demand for communications relays and technologies to place it in the appropriate hands so that it may be utilized effectively.

MIMO communication systems have passed the peak of their novelty in academia and are finding their way into common commercial applications. It is necessary to have an understanding of the advantages and limitations of these systems if government actors intend to maintain pace with the state of digital communications, increasing wireless data demands, and the decrease in bandwidth allocated specifically for government use. It is reasonable to assume that the proportion of Single-Input Single-Output (SISO) communication systems will continue to diminish as new standards and technology seek to maximize the use of available spectrum.

Since the DoD made a fundamental shift to increased preference of commercially available equipment in the 1990s, and all indications are that MIMO systems will replace SISO systems in commercial markets, it is prudent to grow expertise within the government related to these systems. While the goals of private-sector firms differ from those of the government, many of the specific advantages of MIMO communication systems are directly transferable to government applications.

1.2 Problem Statement

This effort seeks to present a model of an air-to-ground wireless communication system and to improve various system components. It has the aim of increasing bandwidth utilization in the form of increased throughput. This research follows the work of Adam Brueggen presented in [2] as it relates to implementing a MIMO transmission link. The original proposal sought to be capable of passing 1 terabit per second (Tbps) of information through this link with very low probability of error, and was concocted with the idea in mind

that high-resolution, persistent surveillance data would be collected by another payload on-board the stationary airship presented in Fig. 1.1.

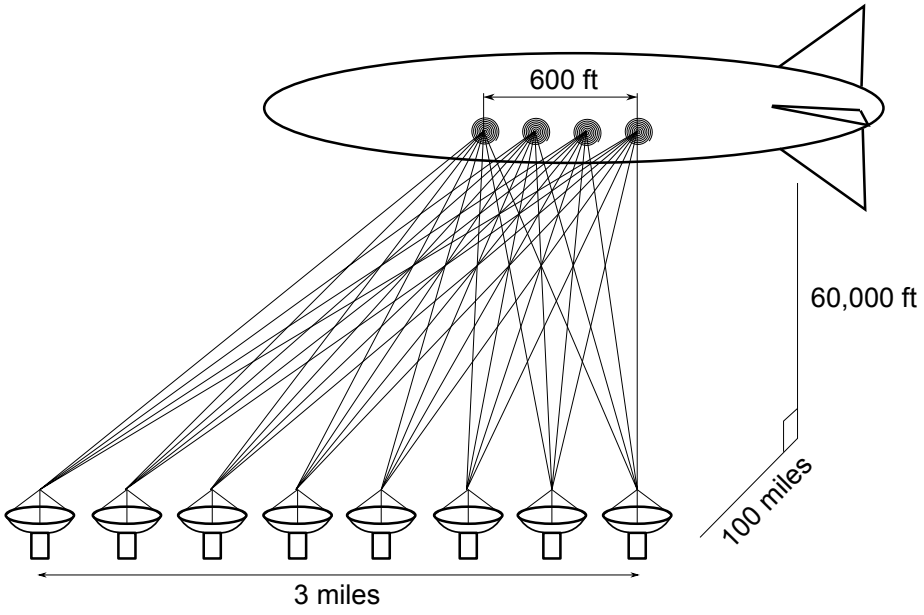


Figure 1.1: Physical representation of MIMO downlink

This investigation attempts to constrain errors beyond the capability of the previous system, but maintains the standard of a maximum Bit-Error Rate (BER) of 1×10^{-5}. The specific codes to be applied are convolutional, turbo, and Low-Density Parity Check (LDPC) codes. The former has been the cornerstone of satellite communications since the 1960s, and the two latter codes have been discovered or popularized in the last twenty years, and, at the time of this writing, are used every day by over 50% of the cellular market. With this in mind, the previously modeled system was able to nearly approach the desired throughput rate while meeting the maximum BER constraint, but was unable to meet 1 Tbps of throughput without sacrificing the BER constraint.

The improvements wrought by reducing error probability will be utilized to apply a digital signalling scheme capable of achieving higher throughput than in previous work.

Augmented performance via error control coding allows for nearly 20% improvement in throughput performance by allowing a larger modulation constellation, but still fails to achieve the goal of 1 Tbps of throughput.

The central ingredient to all of this is the application of these concepts to a MIMO communication system. Cellular networks already make widespread use of MIMO technology, and it is mature enough to be considered for government and military application. Attention will focus on applying the error correction codes described above to various modulation types, all applied to the MIMO context. An evaluation of the best codes, code rates, and modulation schemes will attempt to determine the best configuration of the system with the assumption that it is to be implemented with some or all of the concepts or technology described and discussed in this document.

1.3 Limitations

This document makes broad, and sweeping assumptions about the availability of resources such as bandwidth and physical space as defined in previous work. It also examines a problem from the standpoint of a limited set of variables related to modulation and error correction coding while ignoring other significant factors which affect the wireless communication link. Variables such as atmospheric and flight conditions, internal data processes, and the necessity of timely transmission of information are some of these things which are not addressed.

The results of this investigation are not for immediate application, but should serve as a guideline or reference for those seeking to build a similar system. All of the technology examined within this document exists, and could be applied with some additional inquiry. All link concerns have been assumed to have been addressed by a single Signal-to-Noise Ratio (SNR) value received into the demodulator.

In reality, the assumption which will be made later, that 5 GHz of spectrum centered at 12 GHz are available, is unrealistic. In the United States, this part of the electromagnetic

spectrum has been allocated for radio-location and a variety of satellite communication applications [3, pp. 59-63]. Governmental research has been focused on taking maximum advantage of contested bandwidth for the past several years due to these concerns. However, as this problem most nearly resembles a satellite communications issue, it will be assumed for that sake of argument that the necessary bandwidth could be procured given an urgent, justifiable need.

1.4 Methodology Overview

Experiments consist of Monte Carlo simulations within a mathematical model, designed to account for the many concerns encountered in wireless communications. These simulations will make use of a system model comprised of a variety of communication system blocks to be addressed, blocks which will be employed or built as necessary within MATLAB®.

The majority of collected data are throughput computations and BERs, but any technical considerations encountered in system implementation, which are applicable to a fielded system, will be collected as well. Results will be compared amongst different signalling constellations, Forward Error Correction (FEC) schemes, and antenna configurations. Finally, some consideration will be given to multipath modeling within the system, and results from two models will be juxtaposed.

1.5 Organization

Subsequent chapters outline the fundamental background and theoretical basis for understanding the experiments which were conducted. The experimental model, including all system blocks and technical considerations, is presented in Chapter 3, and is based on relevant theory and conclusions gleaned from previous work. Results of the experiments which were run using this model are detailed in Chapter 4, and attempts to address anomalies encountered in simulation. Finally, conclusions are drawn about the results as

they apply to this problem, and a brief examination of their relevance to similar or problems is provided in Chapter 5.

II. Theoretical Basis and Foundational Concepts

This chapter discusses fundamental digital communication principles relevant to understanding the overarching issues addressed in the problem statement. A limited discussion of basic communication system components is presented, followed by a discussion of interleaving. While interleaving is typically performed after channel coding, it is presented first as the extensive examination of turbo codes in the subsequent section requires a basic understanding of interleavers. The interleaving discussion segues into forward error correction methods including convolutional codes and LDPC codes. Modulation and fading channel models are addressed along with basic MIMO system characteristics to be drawn on in the experimental setup. Lastly, some attention is given to multipath modeling.

2.1 Basic Communication System Architecture

A basic wireless communication system contains a variety of components which transform the information the sender desires to pass to some end recipient or customer into some form suitable for wireless transmission. This transformation must be accomplished in ways which allow for reconstitution once they have been received on the other end. These components typically include things such as encryption, source coding and formatting, error correction coding, waveform generators, modulators, and the physical array of transmitters and receivers with their antennas. A subset of these wireless system components are shown in the block diagram given in Fig. 2.1. Traditionally, everything that is done to a piece of information before transmitting it must be undone in reverse order once the signal has been received at the other end.

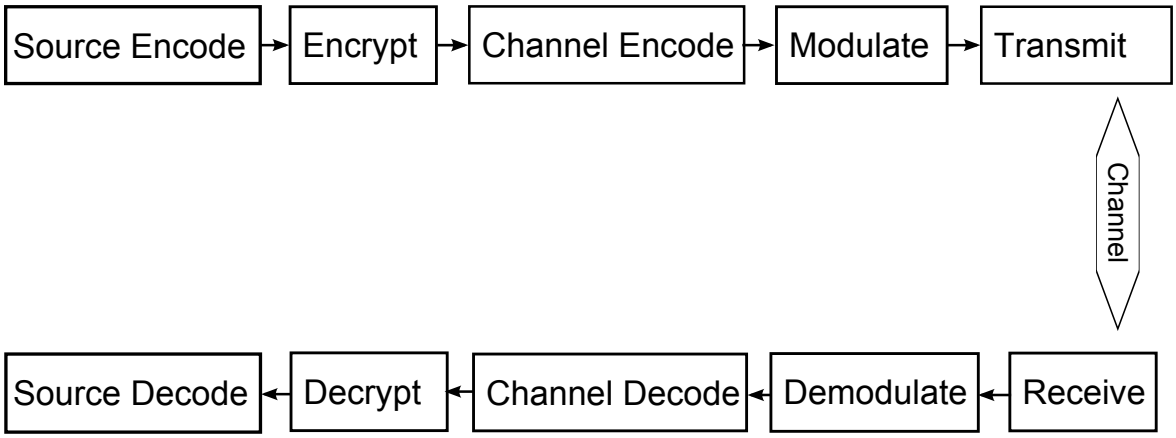


Figure 2.1: Generic wireless communication system

2.2 Interleaving

Channels which incorporate multipath elements usually cannot be assumed to be memoryless, and interleaving is typically implemented in turbo coding [4, p. 339]. For these reasons, a brief discussion of interleaving follows. Interleaving is one technique used when a channel cannot be assumed to be memoryless, or when it is desirable to mitigate the adverse effects of burst errors and fading in the channel. Two common types of interleavers are block interleavers and convolutional interleavers, both of which have been shown to be effective against burst noise. A typical interleaver will separate coded symbols in time by an order of several block lengths in the case of block interleavers, and several constraint lengths in the convolutional case, based on the anticipated duration of incident burst noise [5, pp. 461-468].

2.2.1 Block Interleaving.

The simplest form of a block interleaver can be represented as a matrix transpose as seen below. Data are effectively read into a block of interleaver memory by rows and then read out by columns. In the example below, data are read in to the matrix, transposed, and read out top to bottom, left to right. Larger memory blocks allow for longer error bursts to be accounted for and corrected. However, larger interleaver memory (by design) introduces

larger delay into the transmitted signal. For real-time systems, this is a significant design consideration.

$$\begin{bmatrix} A & B & C \\ D & E & F \\ G & H & I \end{bmatrix} \qquad \Longrightarrow \qquad \begin{bmatrix} A & D & G \\ B & E & H \\ C & F & I \end{bmatrix}$$

2.2.2 Convolutional Interleaving.

Convolutional interleavers make use of memory to incrementally reorder data much like a discrete convolution. Convolutional interleavers were proposed with the idea in mind that they are better suited for use with convolutional encoders than block interleavers [6, pp. 477]. In the simplest case, a parallel set of shift registers are successively loaded with longer length memory. Output from the interleaver is concatenated in parallel. Below, data are read out right to left from the input stream, into a shift register with steadily increasing delays so that they fill what appears similar to a convolution matrix. At the receiver, the exact opposite arrangement of registers read data in and reassemble the original sequence [5, pp. 466-467]. A simple representation of the interleaving process is shown below.

$$\begin{bmatrix} A & B & C & D & E & F & G & H & I \end{bmatrix} \quad \Longrightarrow \quad \begin{bmatrix} A & D & G & & \\ & B & E & H & \\ & & C & F & I \end{bmatrix}$$

$$\begin{bmatrix} A & D & G & & \\ & B & E & H & \\ & & C & F & I \end{bmatrix} \quad \Longrightarrow \quad \begin{bmatrix} A & D & B & G & E & C & H & F & I \end{bmatrix}$$

Received data are de-interleaved by reading data back into the de-interleaver in the same order, delayed in the opposite manner in which they were for encoding, and read back out. A continuation of the above example is given below, where the second line begins after the received data are passed through the register. After the delays, data are

read back out top to bottom, left to right.

$$\begin{bmatrix} A & D & B & G & E & C & H & F & I \end{bmatrix} \quad \Longrightarrow \quad \begin{bmatrix} A & D & G & \\ & B & E & H & \\ & & C & F & I \end{bmatrix}$$

$$\begin{bmatrix} A & D & G \\ B & E & H \\ C & F & I \end{bmatrix} \quad \Longrightarrow \quad \begin{bmatrix} A & B & C & D & E & F & G & H & I \end{bmatrix}$$

2.2.3 Non-Uniform Interleaving.

For a $\mu \times \mu$ memory matrix, a non-uniform interleaver for use with a turbo coder has been proposed in [7] as a good interleaver for turbo encoders while admitting that the optimal interleaver remains the subject of investigation. The non-uniform interleaver proposed follows construction given in (2.3), and seeks to reduce or eliminate patterns in the interleaved data.

$$m_r = ((\mu/2 \cdot (m+n)))_\mu \tag{2.1}$$

$$\xi = ((m+n))_8 \tag{2.2}$$

$$n_r = (([P(\xi) \cdot (n+1)] - 1))_\mu \tag{2.3}$$

Here data are read into the memory matrix, and then interleaved according to the rules for row and column indices m and n. Modulo N operations are represented by $((\cdot))_N$, therefore (2.1)-(2.3) represent modulo μ, eight, and μ respectively. The variable $P(\xi)$ is a set of numbers which are relatively prime with μ.

2.3 Forward Error Correction

It is a well known fact to communication systems engineers that noise in the channel can produce errors in transmitted data. FEC is frequently employed in communication systems to account for this. FEC is a transformation of the data to be transmitted which

increases the chance of correctly receiving the original message given errors are introduced in the channel. This is done by way of reconstructing the original message from the encoded message in a way which corrects errors in the received code, or estimation of the original transmission. FEC codes are usually represented by the name of the code and the code rate itself, where a rate k/n with constraint length K (if applicable) produces n coded elements for k data elements.

2.3.1 Block Codes.

Block codes are the simplest form of FEC codes implemented in digital systems. Typically, a block code is generated by adding some parity sequence to the data sequence somewhere in order to make the original message recoverable if errors are received in the transmission process, or by permuting the message signal by some generator polynomial, most commonly represented as a matrix multiplication operation. The received code is then decoded in a manner intended to remove errors which is specific to the code itself, but commonly takes the form of a parity check matrix. The LDPC codes to be discussed later fit into this category of codes. Discussion of block codes typically focuses on such codes as the Bose-Chadhuri-Hocquenghem (BCH), Golay, and Hamming codes. LDPC codes tend to be discussed in a category to themselves [5, pp. 315-370].

2.3.2 Reed-Solomon Codes.

Reed-Solomon (R-S) codes are thought to be the most commonly implemented codes in practice, and have found their way into a diverse range of applications ranging from satellite communications to the ISO 9660 compact disc format [4, p. 324]. R-S encoders make use of generator polynomials (typically base eight) derived from their constituent BCH codes [6, pp. 471-472]. They are non-linear in nature and were proposed as a possible FEC coding scheme to achieve a 1 Tbps communication relay as described in [2].

2.3.3 Convolutional Encoders.

Convolutional encoders are linear codes commonly implemented as shift registers with memory size $\nu = K - 1$. Convolutional codes do not have a fixed code length for any given input like block codes do, however they are often limited in length to a predetermined input length for practical implementation. This is often achieved by "flushing" the register with known data until a desired encoder state between is achieved, or implementing a reset in the hardware [5, p. 386].

2.3.3.1 Non-Systematic Convolutional Codes.

Nonsystematic Convolutional (NSC) codes are what are traditionally thought of when discussing convolutional codes. They are typically represented in terms of the individual code's constraint length, K, and octal generator polynomial, $[G_1, G_2, ..., G_n]_8$. Greater K values give rise to increased code free distance, and thereby, greater error correcting capability. Free distance of a convolutional code is the minimum Hamming distance between the encoder output of an intended code path and all errant paths which eventually converge back into the intended code path [5, p. 410].

Convolutional encoding fits the definition of a Markov process, and may be represented by a state machines. Transitions between states are commonly presented using state diagrams, tree diagrams, or trellis diagrams [8, p. 421]. A trellis essentially represents state transitions in a linear fashion, going forward in time, for as many state transitions are to be implemented. As such, the possible state transitions become repetitive after a number of transitions have been made. The state diagram of a simple NSC is shown in Fig. 2.2, and an example of the output generated by this encoder is given in Table 2.1. Examples of the trellis representation of convolutional codes which have reached the point of repetitive transition possibilities are shown later in Fig. 2.4.

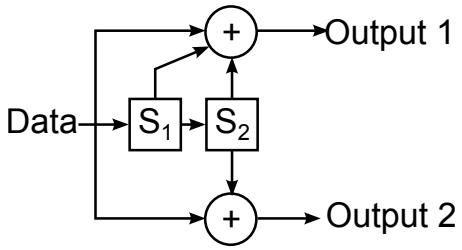


Figure 2.2: State diagram of a NSC encoder with $K = 3$ and generator polynomial $[7,5]_8$

Table 2.1: Rate ½ NSC code example with $K = 3$, generators $[7, 5]_8$

Time Index	Input	Encoder State	Output
0		0 0	
1	1	0 0	1 1
2	1	1 0	0 1
3	0	1 1	0 1
4	0	0 1	1 1
5	1	0 0	1 1
6	1	1 0	0 1
7	0	1 1	0 1
8	0	0 1	1 1
9	1	0 0	1 1
10	0	1 0	1 0

2.3.3.2 Recursive Systematic Convolutional Codes.

Recursive Systematic Convolutional (RSC) codes exhibit a smaller free distance than NSC codes with the same constraint length and generator polynomial, but because they implement a feedback system, they share some properties with Infinite Impulse Response (IIR) filters [9, p. 320]. This fact has the net effect of reducing the number of low-weight code-words (those with a small number of non-zero components) produced by the encoder, and this property is important when considering the concatenation of two codewords together. The reasoning for this is that a high incidence of low-weight code-words reduces the effective free distance between code-words when concatenated, when the greater free distance is desired for successful error correction [5, pp. 492-493]. Because of this, they are known to exhibit better BER performance at lower SNRs than NSC codes

(albeit poorer performance at higher SNR values) [7]. RSC codes, like NSC codes, can be decoded using the Viterbi algorithm, provided a convolutional coding scheme is the end-goal.

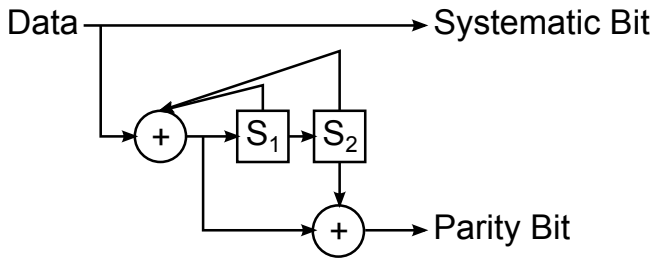


Figure 2.3: State diagram of a RSC encoder with constraint length of 3 and generator polynomial $[7,5]_8$

Table 2.2 shows the output generated from by the encoder in Fig. 2.3 for the same input as in the previous example. Figure 2.4 shows a comparison of the trellis of same code generator implemented as a NSC code and a RSC code. It should be noted that the encoder output corresponding to the inbound branch to a given state is unchanged, but the previous state varies in two cases.

Table 2.2: Rate ½ RSC code example with $K = 3$, generators $[7, 5]_8$

Time Index	**Input**	**Feedback**	**Encoder State**	**Output**
0		0	0 0	
1	1	1	0 0	1 1
2	1	0	1 0	1 0
3	0	1	0 1	0 0
4	0	1	1 0	0 1
5	1	1	1 1	1 0
6	1	1	1 1	1 0
7	0	0	1 1	0 1
8	0	1	0 1	0 0
9	1	0	1 0	1 0
10	0	1	0 1	0 0

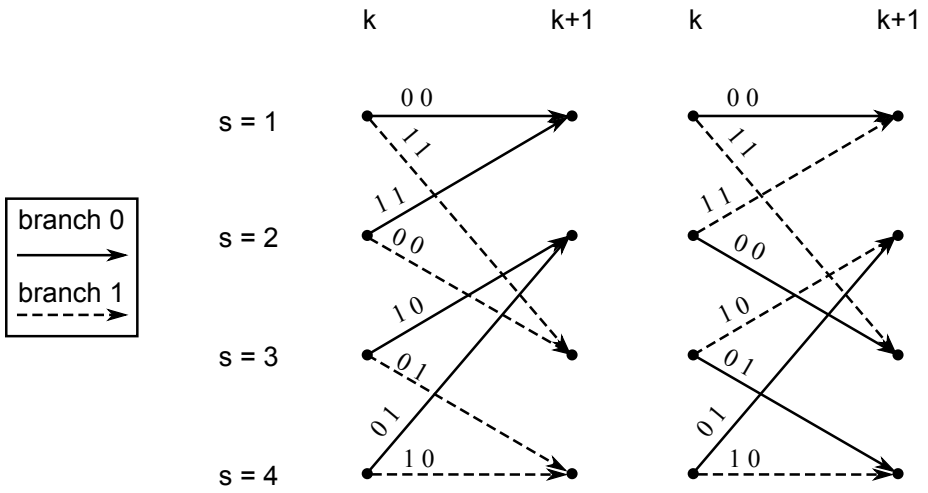


Figure 2.4: Rate ½ NSC (left) and RSC (right) encoder trellises with $K = 3$ and generator polynomial $[7,5]_8$ for both

2.3.4 Convolutional Decoding.

Convolutional decoders may be designed using a number of techniques including the maximum likelihood estimation, the Viterbi algorithm, or via sequential decoding. Maximum likelihood decoding and Viterbi decoding are briefly discussed in the following sections. Another important decoding algorithm is the Modified Bahl, Cocke, Jelinek, and Raviv (BCJR) algorithm which will be discussed later along with turbo decoding.

2.3.4.1 Maximum Likelihood Decoding.

The most immediately obvious way to decode a convolutional code is to compare the received message with all possible code-words of the same length as the received message block. This is precisely what Maximum Likelihood (ML) decoding attempts to do. However, it takes no stretch of the imagination to realize that the number of possible code-words for a mere binary sequence increases exponentially by factors of two. While it may be reasonable to employ this type of decoding to small blocks of code, it quickly becomes cumbersome when longer code-words are required. ML decoding attempts to find

the possible code-word, c which maximizes (2.4), where p is the probability of a code-word given r is the received message of length l and $\frac{1}{n}$ is the code rate for code rates less than or equal to $\frac{1}{2}$.

$$p(\mathbf{R}|\mathbf{C}) = \prod_{i=0}^{l-1} \prod_{j=1}^{n} p(r_{ij}|c_{ij}) \tag{2.4}$$

It can be shown that as the message size increases, the log-likelihood function necessary to obtain the ML estimate of the transmitted message vector also grows exponentially in complexity as a function of the constraint length of the code and the length of finite codewords produced by the encoder [10, p. 249].

2.3.4.2 Viterbi Algorithm.

The Viterbi algorithm, created by Andrew Viterbi in 1967, makes use of memory and the encoder trellis to decode received messages, and has been shown to be approximately equal to the ML for long codewords. The Viterbi algorithm is typically implemented to choose the path through the encoder trellis over a frame of time samples by computing branch metrics, or the Hamming distance between each possible trellis path, from start to end of the frame, and incrementally eliminating paths entering subsequent trellis nodes by virtue of their branch and cumulative path metrics. The fundamental idea is that when two paths enter the same node, one can always be eliminated based on its branch metric. After each time sample in the window, the branch metric between previous states and the current state are computed, and the branches with the weakest branch metric are eliminated. Once the branch metrics for the final time sample have been computed and weak branches eliminated, the cumulative path metric across the whole window is computed (by addition when Hamming distance is used). Finally, the path with the best path metric is assumed to be the transmitted message [10, pp. 252-253], [5, pp. 401-403]. An illustration of this is given in Appendix A.

The Viterbi algorithm has been shown to be the ML estimate of the transmitted message when implemented across the complete coded sequence [5, p. 401]. However,

when it is modified to compute statistics for a fixed window of time as described above, the result is a "good" estimate that is only approximately equal to the ML estimate of the message [10, pp. 249-253].

2.3.5 Turbo Coding.

Turbo codes are often viewed as members of the broader category of concatenated codes, and were known at the time of their discovery for their ability to attain very high coding gains over any other code implemented in practical applications. Concatenated codes have been shown to achieve the error performance of significantly longer codes than the relatively shorter constituent codes used in their generation. Turbo codes likewise have been shown to produce significant error correction performance approaching the theoretical Shannon limit of capacity in SISO systems, and find their strength in the iterative decoding process [6, p. 552], [5, p. 475].

2.3.6 Turbo Encoder Structure.

Well-studied turbo codes frequently make use of two of the same rate ½, RSC encoders. Data are passed into at least three places; the encoder's systematic output node, the input of the first encoder, and an interleaver. As seen in Fig. 2.3, each encoder block produces at least a systematic output and a parity output. The systematic output paths are ignored (or not implemented), but the parity streams of both RSC encoders are then concatenated in parallel with the systematic output as seen in Fig. 2.5 [11, p. 235]. The primary purpose of interleaving is not to reduce the impact of burst noise in the data, but is designed scramble the input stream in such a way that the two encoders produce parity output values which are not highly correlated. This improves the validity of the assumption, later made in the decoding step, that extrinsic information passed between decoders are statistically independent of each other [7]. While this assumption is simply untrue given that the same data are used to produce both or all parity streams, research has shown that presenting each decoder with very weakly correlated extrinsic information

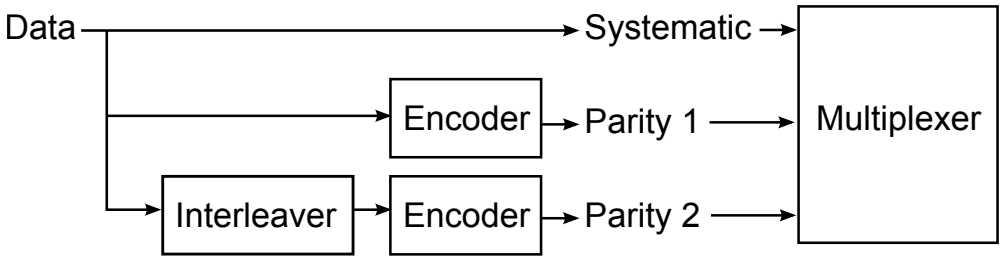


Figure 2.5: Basic turbo encoder structure

allows for effective implementation of the turbo concept. The weaker this correlation, the better the decoder will perform [5, p. 492].

2.3.7 Turbo Decoding.

The term "turbo code" was coined by C. Berrou and A. Glavieux in their original 1993 publication discussing the topic. Turbo codes were quickly adopted by such applications as the National Aeronautics and Space Administration (NASA) *Pathfinder* mission to Mars in 1997. The coding method itself has perhaps not been the most significant contribution to the to the field of digital communications, but instead, the broader application of what is referred to as the "turbo principle" as it applies to iteratively decoding, detecting, estimating, or equalizing some desired signal [12], [13, p. 158]. The following discusses the steps required for iterative turbo decoding which are later applied in Section 3.2.4.

Turbo decoding begins when signals pass into a demultiplexer from the demodulator. This demultiplexer separates the received message into its original constituent streams, consisting of a minimum of a data stream, and two parity streams. The first decoder receives the data stream and the first parity stream, and the original message is estimated using the modified BCJR algorithm presented in [7]. Alternatively, the Soft-Output Viterbi Algorithm (SOVA) could be employed to produce the requisite extrinsic information, but that process is not addressed in this document [10, p. 261]. With the exception of the final

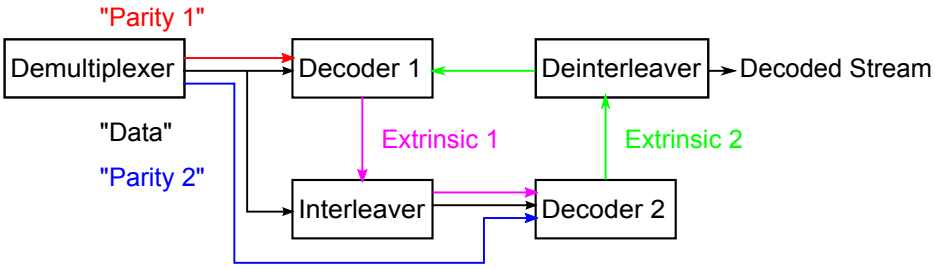


Figure 2.6: Iterative decoding loop for turbo decoding

decoder output from the pair of decoders (or triplet or more as applicable), hard-decision Viterbi decoders do not provide sufficient information to be employed as the decoding method for turbo codes.

Decoding continues when both the original data stream and the extrinsic output from the first decoder are (separately) interleaved and passed into the input of the second decoder, along with the second parity stream. At this point, the extrinsic output from the second decoder may be used to estimate the original message, but doing this would not be advantageous. Not only has an inordinate amount of effort been applied to a relatively simple task, but the performance of a single decoding cycle does not yield much better performance than simply transmitting the message without encoding it as will be seen in Section 3.2.5. For additional iterations, the output from the second decoder must be de-interleaved and passed to the first decoder as a priori input, along with the original data and first parity sequence. This basic decoding loop is shown in Fig. 2.6. Successive decoding cycles improve the BER of the transmitted message at the receiver up to a point of diminishing returns [10, pp. 259-262], [5, pp. 496-497].

2.3.7.1 Modified BCJR Algorithm.

The mechanics of performing turbo decoding boil down to applying the modified BCJR algorithm successfully at each decoding stage. This algorithm is represented in

summary by the Likelihood Ratio (LR) in (2.5). This equation is actually a factorization of Bayes' rule (2.6) so that the Maximum A Posteriori (MAP) decision rule (2.7) [14, pp. 79-82] for binary hypotheses can be applied later. The development for (2.5) is given in detail in [5, pp. 498-504]. Each of the terms in (2.5), the forward state metric for state s at time k, α_k^s, the reverse state metric, $\beta_{k+1}^{f(i,s)}$ based on the function $f(i, s)$ of the next state branch metric input i, and branch metric, $\delta_k^{i,s}$, is specific to the encoder trellis. It is assumed that the input into the algorithm contains soft-decisions from the demodulator based on antipodal signalling. Here antipodal should not be taken to imply a strict definition of antipodal signalling, but instead to mean that received binary ones and zeros are represented as some kind of random variable, or derivation therefrom, centered around the positive and negative equivalent of some numeric value, e.g. ± 1. Representing these values as binary ones and zeros will cause the algorithm to fail. This fact should be apparent from the first step of the decoding process: computing the branch metrics according to (2.8).

$$\Lambda(\hat{d}_k) = \frac{\sum_s \alpha_k^s \delta_k^{1,s} \beta_{k+1}^{f(1,s)}}{\sum_s \alpha_k^s \delta_k^{0,s} \beta_{k+1}^{f(0,s)}} \tag{2.5}$$

$$P(H_i|x) = \frac{p(\mathbf{x}|H_i)P(H_i)}{p(\mathbf{x})} \tag{2.6}$$

$$P(H_1|\mathbf{x}) \underset{H_0}{\overset{H_1}{\gtrless}} P(H_0|\mathbf{x}) \tag{2.7}$$

$$\delta_k^{i,s} = A_k \pi_k^i \exp\left[\frac{1}{\sigma^2}\left(x_k u_k^i + y_k v_k^{i,s}\right)\right] \tag{2.8}$$

$$A_k = \frac{1}{2^\nu 2\pi\sigma^2} \exp\left(\frac{-x_k^2 - y_k^2 - 2}{2\sigma^2}\right) \tag{2.9}$$

The branch metric in (2.8) assumes an Additive White Gaussian Noise (AWGN) channel. In (2.8), π_k^i is the a priori probability of following branch i, x_k contains received

data stream channel values, y_k contains received parity stream channel values, and σ^2 is the variance of the channel noise. The constant A_k is a collection of terms resulting from the derivation of the branch metric from the standard normal distribution. As none of these terms are unique to i or s, this constant has been ignored for the rest of this document. A visual representation of the branch metric is given in Fig. 2.7. Once branch metrics have been computed for the entire trellis, the other two of the metrics for (2.5) may be computed, forward state α_k^s metric first, going forward in time, followed by reverse state metric β_k^s, going backward in time from the final state. These equations are both recursive, and in order to prevent computational overflow, each α_k^s should be normalized by $\aleph_k$ given in (2.11). The value $\aleph_k$ should be saved so that β_k^s may be scaled by the same factor at each time sample. Hence the final forward and reverse state metrics are given in (2.13).

$$\alpha_k^s = \sum_{j=0}^{1} \alpha_{k-1}^{b(j,s)} \delta_{k-1}^{j,b(j,s)} \tag{2.10}$$

$$\aleph_k = \sum_{s} \sum_{j=0}^{1} \alpha_{k-1}^{b(j,s)} \delta_{k-1}^{j,b(j,s)} \tag{2.11}$$

$$\beta_k^s = \sum_{j=0}^{1} \beta_{k+1}^{f(j,s)} \delta_k^{j,s} \tag{2.12}$$

$$\bar{\alpha}_k^s = \frac{\alpha_k^s}{\aleph_k} \qquad \bar{\beta}_k^s = \frac{\beta_k^s}{\aleph_{k-1}} \tag{2.13}$$

It is expedient at this point to reorganize (2.5) for purposes of the iterative decoding loop. By factoring out components from (2.14) which are not dependent upon the encoder state (but still vary with k), the expression for $\Lambda(\hat{d}_k)$ can split in to three manageable parts as shown in (2.16). Taking the natural logarithm of this factorization, as in (2.17) will provide a relationship which will enable iterative decoding later.

$$\Lambda(\hat{d}_k) = \frac{\sum_s \bar{\alpha}_k^s \delta_k^{1,s} \bar{\beta}_{k+1}^{f(1,s)}}{\sum_s \bar{\alpha}_k^s \delta_k^{0,s} \bar{\beta}_{k+1}^{f(0,s)}} \tag{2.14}$$

$$= \pi_k \exp\left(\frac{2x_k}{\sigma^2}\right) \frac{\sum_s \bar{\alpha}_k^s \exp\left(\frac{y_k v_k^{1,s}}{\sigma^2}\right) \bar{\beta}_{k+1}^{f(1,s)}}{\sum_s \bar{\alpha}_k^s \exp\left(\frac{y_k v_k^{0,s}}{\sigma^2}\right) \bar{\beta}_{k+1}^{f(0,s)}} \tag{2.15}$$

$$= \Lambda_k^a \Lambda_k^c \Lambda_k^e \tag{2.16}$$

$$L(\hat{d}_k) = \ln(\Lambda(\hat{d}_k)) = L_k^a + L_k^c + L_k^e \tag{2.17}$$

In (2.16), the a priori probability fed into the decoder is $\Lambda_k^a = \pi_k = \pi_k^1/\pi_k^0$ and L_k^a is its natural logarithm. The channel estimate, Λ_k^c, maps to the first exponential term and by extension L_k^c. Both Λ_k^e and L_k^e correspond to the ratio of sums, or the decoder's extrinsic output. The values of Λ_k^c, and L_k^c do not change based on the a priori probability. For a single decoder, the entire expression in either (2.16) or (2.17) may be evaluated to produce either hard or soft decisions regarding the transmitted sequence at this point. As stated before, this process produces the MAP estimate of the transmitted sequence.

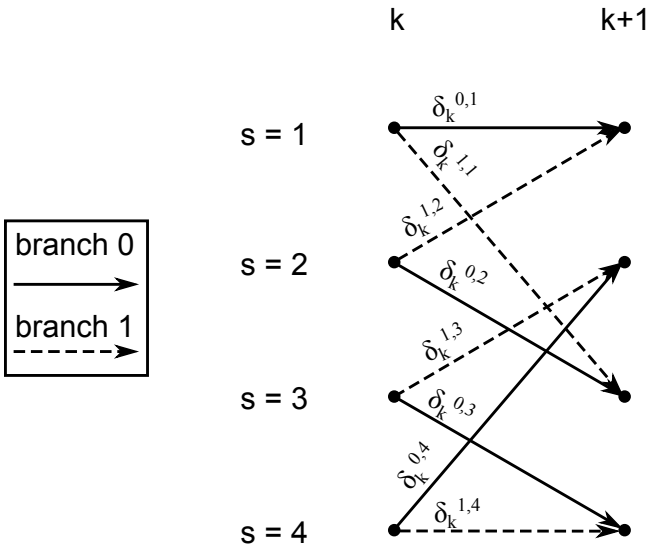


Figure 2.7: Branch metrics overlaid onto the trellis of a $[7, 5]_8$ RSC code

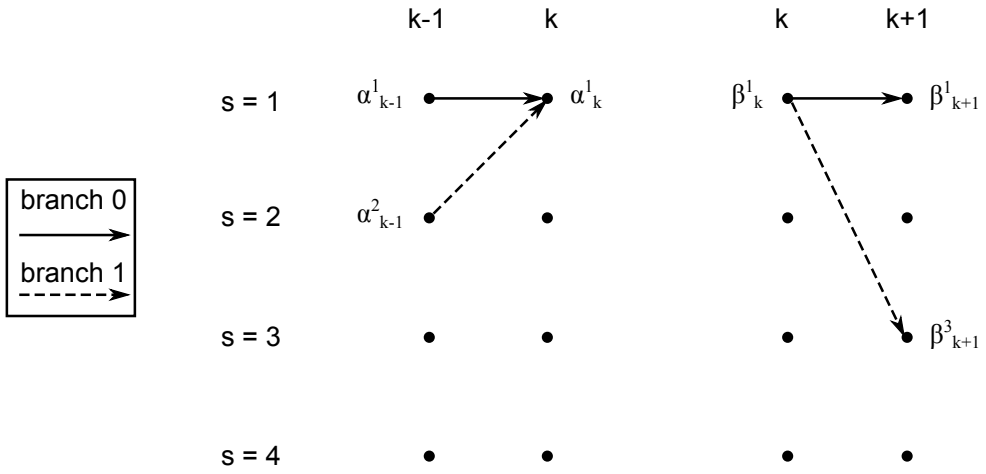


Figure 2.8: Example state metric feeding for a $[7,5]_8$ RSC code trellis. Following (2.10), (2.12) and the trellis paths in Fig. 2.7, $\alpha_k^1 = \alpha_{k-1}^1\delta_{k-1}^{0,1} + \alpha_{k-1}^2\delta_{k-1}^{1,2}$ and $\beta_k^1 = \beta_{k+1}^1\delta_k^{0,1} + \beta_{k+1}^3\delta_k^{1,1}$.

2.3.7.2 Iterative Decoding.

When multiple decoders are implemented in series as in Fig. 2.6, the extrinsic value from the decoder is passed to the next decoding stage as the a priori probability or Log-Likelihood Ratio (LLR) as appropriate. This cycle continues until the estimates from both decoders converge, or more frequently in practice, desired number of decoding iterations has been completed. Hard decisions may be made on the decoder output following the final decoding iteration using (2.18). The estimated binary sequence, or soft output is then passed to the next stage of the overall communication system.

$$L(\hat{d}_k) \underset{\hat{d}_k=0}{\overset{\hat{d}_k=1}{\gtrless}} 0 \tag{2.18}$$

2.3.8 Low-Density Parity Check Codes.

LDPC codes attempt to create a sparse parity matrix in effort to maximize minimum codeword distance. They are known to be simpler to implement than turbo codes, and have found themselves employed in many applications ranging from the wireless 802.11n standard to satellite television applications [4, pp. 340-344], [15]. They are linear block codes which have a set number of ones allowed per row and column of the parity matrix.

This number is assumed to be much less than the number of rows or columns of the matrix [6, p. 569].

2.3.9 Decoding LDPC Codes.

The decoding process for LDPC codes is also iterative, however the complexity of the decoding algorithm is far less than that of the turbo code. The bit-flipping algorithm operates on demodulator hard decisions. It finds its basis in computing the syndrome **s** of a received sequence, x, which is computed as in (2.19)

$$\mathbf{s} = \mathbf{x}\mathbf{H}^T, \tag{2.19}$$

wherein **H** is the parity-check matrix. Bits which do not satisfy the parity equations defined by the vector sum of the rows of the parity check matrix are flipped and the syndrome is recalculated. This is done until a desired number of iterations has transpired, or there are no longer any unsatisfied parity equations [6, p. 571].

Alternatively, messages may be decoded using the sum product algorithm as applied to the code's factor graph. The sum-product algorithm is a MAP decoding technique which attempts to maximize the sum over all code words of the product of all parity check functions with the a posteriori probability of each received message bit, given a specific coded bit.

2.4 Modulation

Binary data cannot be transmitted through a wireless channel without first being modulated by some carrier wave. Two common methods of digitally modulating signals are the use of a Phase-Shift Key (PSK) or Quadrature Amplitude Modulation (QAM) scheme, both of which are discussed in the following sections. Analog modulation techniques are not discussed here as digital modulation is the preferred approach to the passage of digital information in a bandpass system in general [16, p. 608], and are virtually unwelcome in

analogous satellite applications due to cost, physical size, and power considerations [17, p. 558].

2.4.1 Phase-Shift Keying.

PSK is a spectrally efficient method of modulating signals which is preferred for satellite applications where many channels exist in the same bandwidth, or frequency spectrum is limited [17, p. 560]. PSK symbols are generated using (2.20) [6, pp. 101-107]

$$x(t) = A\cos\left(2\pi f_c t + \frac{2\pi}{M}(m-1)\right), \quad m = 1, 2, \cdots, M, \tag{2.20}$$

where f_c is the carrier frequency, m is the specific symbol over some range of time t, and M is the constellation size.

It is important to note that within signal constellation arrangements based on Gray code, varying adjacent signals in any direction on the constellation map by only a single bit, is known to reduce bit-error when noise or fading cause an error in symbol estimation at the receiver. The bit-error probability, P_b, for PSK, assuming Gray coding, is given by [6, p. 194]

$$P_b = \frac{2}{\log_2(M)} Q\left(\sqrt{(2\log_2 M)\sin^2\left(\frac{\pi}{M}\right)\frac{E_b}{N_0}}\right), \tag{2.21}$$

or in the case of Binary Phase-Shift Key (BPSK), by [5, p. 533]

$$P_b = Q\left(\sqrt{2\frac{E_b}{N_0}}\right), \tag{2.22}$$

wherein $Q(\cdot)$ is the complementary error function defined by [5, p. 545]

$$Q(x) = \frac{1}{\sqrt{2\pi}}\int_x^\infty \exp\left(-\frac{u^2}{2}\right)du, \tag{2.23}$$

E_b is the average energy per bit, and N_0 is the noise power in the band.

2.4.2 Quadrature Amplitude Modulation.

Another popular modulation scheme, known as QAM, makes use of a cosine (in-phase), and sine (quadrature) pair to generate symbols. The amplitude of the in-phase

and quadrature components varies based on which symbol is being transmitted, and the number of acceptable amplitudes determines the constellation size, M. QAM symbols are generated according to (2.24), and can be represented in complex form as in (2.25) [6, pp. 106-107].

$$x(t) = A_I \cos(2\pi f_c t) - A_Q \sin(2\pi f_c t) \tag{2.24}$$

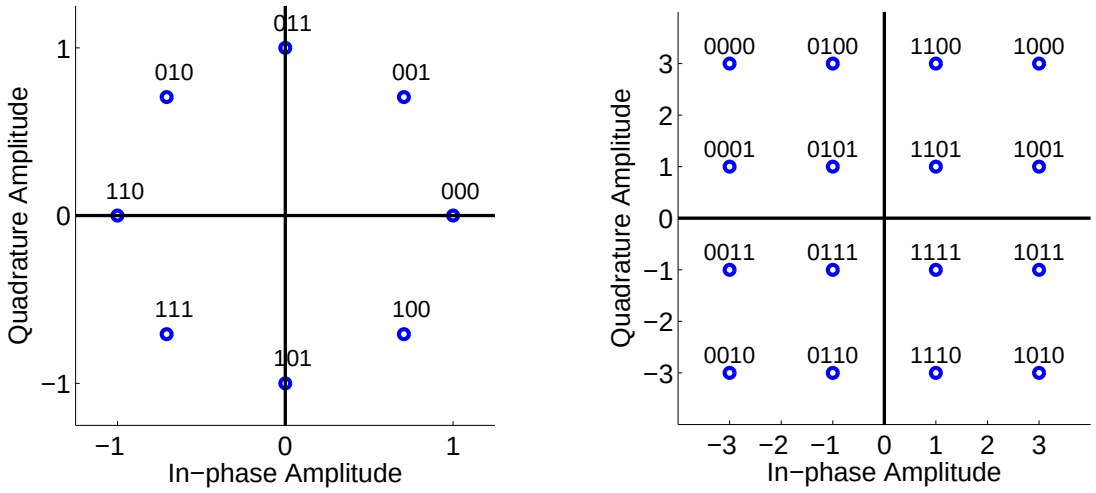
$$x_k = A_I + jA_Q \tag{2.25}$$

One of the chief advantages of higher order QAM is the ability to transmit much more information with a single symbol than is possible with BPSK. Although the error performance of the system decreases with larger QAM constellation sizes (a fact which can be seen in Fig. 2.10), it is superior to that of M-PSK constellations of equivalent M. Furthermore, this trade-off is often deemed acceptable up to a certain point because the gains in the information rate of the system increases linearly with $\log_2(M)$. Rectangular QAM constellations such as the one in Fig. 2.9 do not represent the ideal spacing of constellation points with regard to BER, but are typically favored for the simplicity by which they are implemented. The probability of bit-error for the rectangular QAM constellation is by (2.26) [6, p. 198]

$$P_b = \frac{4}{\log_2(M)}\left(1 - \frac{1}{\sqrt{M}}\right) Q\left(\sqrt{\frac{3\log_2(M)}{M-1}\frac{E_b}{N_0}}\right) \times \left(1 - \left(1 - \frac{1}{\sqrt{M}}\right) Q\left(\sqrt{\frac{3\log_2(M)}{M-1}\frac{E_b}{N_0}}\right)\right). \tag{2.26}$$

2.5 Channels

The environment through which a transmitted signal must pass is referred to as the channel. Wireless channels are often modeled as linear and time-varying in nature. The



(a) (b)

Figure 2.9: Constellations for Gray coded 8-PSK in (a) and 16-QAM in (b)

rate at which channel conditions change relative to the carrier and sampling frequencies determines whether the channel is referred to as slow or fast-fading [8, p. 452].

2.5.1 Rayleigh Fading Channel.

The Rayleigh distribution is frequently used to model the multipath components of an complex Gaussian signal passing through a AWGN channel, and is given by [10, p. 78], [18, p. 210]

$$p_Z(z) = \frac{z}{\sigma^2} \exp\left[-\frac{z^2}{2\sigma^2}\right], \quad z \geq 0, \tag{2.27}$$

where the in-phase and quadrature components of the signal variance are both assumed to be equal to σ^2 and z is the magnitude of the signal envelope. The Rayleigh fading channel applies to flat-fading, complex AWGN channels, for as long as the constituent components are, in-fact, zero-mean in nature, and to individual multipath components of a frequency selective channel.

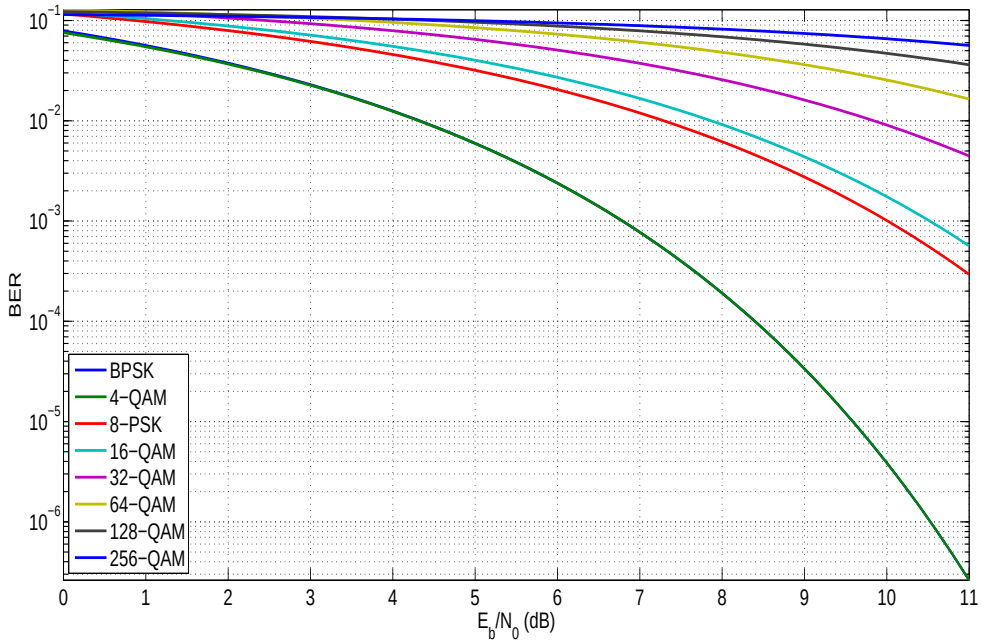


Figure 2.10: Theoretical error performance comparison of uncoded modulation schemes for channel E_b/N_0 values

2.5.2 Ricean Fading Channel.

The fading experienced by the Line-of-Sight (LOS) component of a transmission pathway through dispersive media can be modeled by the Ricean distribution. This distribution is modeled as

$$p_Z(z) = \frac{z}{\sigma^2} \exp\left[-\frac{(z^2+\gamma^2)}{2\sigma^2}\right] I_0\left(\frac{z\gamma}{\sigma^2}\right), \quad z \geq 0, \tag{2.28}$$

where $2\sigma^2$ is the average power in the multipath transmissions, γ^2 is the average power in the LOS transmission, and I_0 is a modified zeroth order Bessel function of the first kind. Once the LOS signal has attenuated significantly, the pathway degenerates into the aforementioned Rayleigh distribution [10, p. 78], [18, p. 212], [5, p. 215].

$$I_0(x) = \frac{1}{2\pi} \int_0^{2\pi} \exp\left[x\cos(\theta)\right] d\theta \tag{2.29}$$

2.6 MIMO Communication Systems

MIMO Communication systems take advantage of spatial diversity to achieve higher data rates than possible with a single transmitter and receiver pair. At a very simplistic level, the capacity increases for a wireless system with the number of transmitter and receiver pairs until signal discrimination becomes encumbered by interference in the channel. Different ratios of receivers and transmitters affect this capacity as addressed below.

2.6.1 Channel Capacity.

Capacity is a measure used to determine how much information can be passed through a channel, and is characterized by the mutual information between the mutual input between the signal entering the channel, and the signal exiting the channel. In-depth discussion of information theory will be avoided, but it is difficult to examine MIMO channel capacity without addressing it. Capacity for the generic SISO and MIMO channels are discussed in the following sections.

2.6.1.1 SISO Capacity.

The capacity of a SISO link with bandwidth W in an AWGN channel is typically represented by some form of (2.30) below. This equation presents capacity in units of bits-per-second and makes use of some known total signal power, S, and known average noise power over the entire band, N_0.

$$C = W \log_2 \left(1 + \frac{S}{N_0 W} \right) \tag{2.30}$$

This number is the theoretical bound on capacity presented by C. Shannon in 1948. It ran contrary to contemporary wisdom regarding realizable capacity at the time, and was thought unreachable until the advent of turbo codes in 1993. The Berrou et al. publication on turbo codes reignited the discussion of capacity bounds and attainable data rates [10, pp. 100-101].

2.6.1.2 *MIMO Capacity.*

MIMO capacity is more complicated to calculate than SISO capacity is. Equation (2.30) above is actually an application of the Shannon limit given by (2.31), in which $(X; Y)$ denotes mutual information.

$$C = \max_{p(x)}(X; Y) \tag{2.31}$$

However, in a MIMO system, the channel coefficients themselves cannot be ignored. Below, $\mathbf{H}$ is a $N_r \times N_t$ matrix of channel coefficients (N_r is the number of receivers, and N_t is the number of transmitters).

$$C = \log_2 \left| \mathbf{I}_{N_r} + \frac{E_s}{N_t N_0} \mathbf{H}\mathbf{H}^H \right| \tag{2.32}$$

In the signal to noise ratio, E_s represents the total energy from all transmitters, $\mathbf{I}$ represents the identity matrix, of subscripted dimension, $(\cdot)^H$ denotes the conjugate transpose, or Hermitian operator, and $|\cdot|$ indicates a matrix determinant. Capacity expressed in this form carries units of throughput per unit of bandwidth, most commonly bps per Hz [6, p. 983].

2.6.2 *Maximum Likelihood Signal Detection.*

ML detectors are known to be optimal in AWGN channels. This type of detection selects the symbol with the minimum Euclidean distance from expected transmissions, and thereby minimizes error probability [6, p. 970]. However, the computational complexity of implementing ML detectors frequently renders them impractical for real systems for the same reasons discussed in Section 2.3.4.1.

2.6.3 *Signal Detection by Means of Singular Value Decomposition.*

When channel conditions at both the receiver and the transmitter are known, signal detection may be accomplished by Singular Value Decomposition (SVD). The signal is processed prior to transmission, and again upon receipt to take advantage of these known channel conditions. Besides added complexity, handling the signal in this way ignores the possible benefits of channel diversity in the MIMO channel, such as improved error

performance, but improves the capacity of the system. A linear representation of the channel is given as

$$\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{n}. \tag{2.33}$$

In (2.33), $\mathbf{y}$ is the received signal vector of dimension $N_r \times 1$ at a single time sample, $\mathbf{x}$ and $\mathbf{n}$, both of dimension $N_t \times 1$, are the transmitted signal vector and received noise vector respectively for that time sample, and $\mathbf{H}$ is the $N_r \times N_t$ channel matrix. Matrix $\mathbf{H}$ of arbitrary rank r is factored, in (2.34) below, into a diagonal matrix of singular values $[r \times r]$ represented by $\mathbf{\Sigma}$, orthonormal matrices $\mathbf{U}$ $[N_r \times r]$, and $\mathbf{V}$ $[N_t \times r]$.

$$\mathbf{H} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^H \tag{2.34}$$

A single matrix multiplication operation by $\mathbf{V}$ prior to transmission provides for decoupling and estimation of the original transmission with another matrix multiplication upon reception. The final relationship for signal estimation is given below as [6, p. 975]

$$\hat{\mathbf{x}} = \mathbf{U}^H\mathbf{y} = \mathbf{\Sigma}\mathbf{x} + \mathbf{U}^H\mathbf{n}. \tag{2.35}$$

2.6.4 Inverse Channel Detection.

Frequently, channel information at the transmitter is unavailable or undesired. When this is the case, the receiver must account for all signal estimation. Given that training data are transmitted for the sake of estimating $\mathbf{H}$, and it is often the case that $\mathbf{H}$ is not invertible, computing its pseudo-inverse, $\mathbf{H}^+$, produces the optimal least-squares solution for $\hat{x}$ in (2.33) [19, p. 335]. This approach to determining transmitted symbols is known as Inverse Channel Detection (ICD). ICD is frequently implemented in MIMO systems, and relies on a good estimate of channel conditions at at the receiver only. This method of separating out individual transmission components is less computationally complex than most other detection methods, and lends itself to reducing processing requirements at the transmitter [20, p. 251]. When choosing channel weights in this fashion (assuming $N_r > N_t$), the channel matrix pseudo-inverse is computed by (2.36) [6, p. 970]

$$\mathbf{H}^{+} = (\mathbf{H}^{H}\mathbf{H})^{-1}\mathbf{H}^{H}. \tag{2.36}$$

2.6.5 Minimum Mean-Squared Error Detection.

Minimum Mean Squared Error (MMSE) detection attempts to set the scaling matrix for the received signal by minimizing the error function between the transmitted symbol, s, and the received symbol by making use of the autocorrelation of the received signal vector, $\mathbf{R}_{yy}$. Optimum weight vectors of length N_r are given by [6, p. 970]

$$\mathbf{w}_n = \mathbf{R}_{yy}^{-1}\mathcal{E}[s_n^{*}\mathbf{y}], \quad n = 1, 2, ..., N_t. \tag{2.37}$$

Here $\mathcal{E}[\cdot]$ denotes expected value and $(\cdot)^{*}$ denotes the complex conjugate operation. These weight vectors are assembled into a weight matrix

$$\mathbf{W} = [\mathbf{w}_1,\ \mathbf{w}_2,\ \mathbf{w}_{Nt}] \tag{2.38}$$

which may be applied to the received signal via matrix multiplication

$$\hat{\mathbf{x}} = \mathbf{W}^{H}\mathbf{y}. \tag{2.39}$$

2.7 Multipath Modeling

A common issue which must be dealt with when designing a communication system is multipath interference. Multipath is the constructive and destructive combination of electromagnetic waves which alters their amplitude and phase at the point of convergence, and occurs in nearly every wireless channel. As the transmitted waveform travels through the channel, reflective surfaces, such as the ground or buildings, will alter the direction the wave is travelling and the phase of the signal itself, so that multiple copies of the original transmission arrive at the receiver at different times, and with different amplitude and phase than the LOS transmission. A notable exception to this is a ground-to-satellite transmission, in which it may be unlikely that any scattering due to physical objects will occur, depending on frequency and orbital mechanics. Two models for accounting for multipath in a communication system are presented in the following sections.

2.7.1 Two-Ray Ground Reflection Model.

The two-ray ground reflection model treats only a LOS transmission and a reflected facsimile of that original transmission, as the name implies. The total energy at the receiver is the simple combination of the two rays. For instances in which the horizontal distance the ray travels is significantly larger than the vertical distance (and the resulting ground reflection angle is small), the difference in the path length Δ can be defined based on transmitter height h_t receiver height h_r and horizontal distance travelled d as in

$$\Delta \approx \frac{2h_r h_t}{d}. \tag{2.40}$$

When this condition does not apply, the path difference must be calculated using

$$\Delta = \sqrt{(h_t + h_r)^2 + d^2} - \sqrt{(h_t - h_r)^2 + d^2}. \tag{2.41}$$

The difference between the time required for the LOS ray and the reflected ray to travel to the receiver is represented by (2.42) below, where c is the speed of light in a vacuum. Dividing this delay by the receiver's sample period yields the number of sample periods which transpire between the arrival of the LOS transmission, and that of the multipath copy [18, pp. 120-125] shown here

$$\tau = \frac{\Delta}{c}. \tag{2.42}$$

2.7.2 Tapped Delay Line Channel Model.

The Tapped Delay Line (TDL) channel model attempts to account for all multipath rays which could conceivably be received in a given sample period. Instead of a single LOS ray being combined with a single reflected ray, the receiver obtains multipath for as many time samples as a single transmission can be expected to reflect into the receiver with any distinguishable energy. The TDL model makes use of (2.43), in which multipath rays are summed until some maximum delay K [14, pp. 170-172].

$$y[k] = \sum_{m=0}^{K} h[m]x[k-m] + n[k] \tag{2.43}$$

2.8 Conclusion

This chapter has addressed the spectrum of coding, modulation, and channel modeling background necessary for understanding the experimental set-up presented in the next chapter. These concepts will be built upon and modified to implement a simulated MIMO communication system.

III. Experimental Configuration

THE MIMO communication system configuration discussed in this chapter touches on the specifics of error-control codes, modulation, and channel models, and fading which will be used to evaluate its feasibility. Details of the system implementation are presented, technical challenges are illuminated, and data to be collected are discussed. This begins with assumptions and constraints to be applied to the model, along with specific variables to be utilized, and continues to examine each block in greater detail. Finally, the expected results and approach to collecting them are considered.

3.1 System Model

The overarching system consists of an airship and a ground receiver array. The airborne component of the system is represented as a stationary airship similar to a Zeppelin, which contains a collector, control systems, and a transmitter assembly. The collector is a persistent surveillance payload collecting high-resolution video, and thus requiring a significant data transfer rate to the ground to avoid overloading on-board storage. The control systems consist of physical equipment required for power generation and airship automation, and computer hardware and software necessary to operate that equipment. No further attention will be given to the payload or control systems. The transmitter assembly consists of all hardware and software required for data formatting, encryption, error-control coding, modulation and transmission. The components of the transmitter assembly for in-depth examination are the encoder, modulator, and number of transmitting antennas.

3.1.1 System Parameters.

The ground-station contains a large receiver array, equipment necessary for reconstructing the original data, and storage equipment. The pieces of the ground-station of

Table 3.1: Airship System Parameters

Operating Altitude	60,000 ft
Horizontal Transmission Distance	100 miles
Maximum Receiver Spacing	3 miles
Airship Length	600 ft
Transmitter Frequency	12 GHz
SNR at the Demodulator	4.77 dB
Available Bandwidth	5 GHz
Minimum Data Rate	1 Tbps
Maximum Bit-Error Rate	1×10^{-5}

interest are the number of receiver antennas, the demodulator and the decoder. Issues such as amplifiers, data storage and transfer, and decryption have been ignored. Finally, the channel has been assumed to be a frequency-selective MIMO channel in the sense that the constructive and destructive interference resulting from cross-talk between transmitter-receiver combinations, and multipath from ground-bounce and stationary scatters, causes fading. With this in mind, a short summary of system parameters, many of which were selected to easily compare with previous research in [2], is presented in Table 3.1.

3.1.2 Antenna Configurations.

Following the conclusions in [2], wherein (2.32) was multiplied by the system's bandwidth for the sake of effectively comparing the capacities of SISO, Multiple-Input Single-Output (MISO), Single-Input Multiple-Output (SIMO), and MIMO systems together, three antenna configurations were utilized. Previous research suggested configurations of antennas likely to produce throughput of 1 Tbps based on capacity results for ratios of transmitter to receivers, $[N_t{:}N_r] \in \{[1{:}1], [1{:}2], [1{:}4], [1{:}8]\}$. The suggested

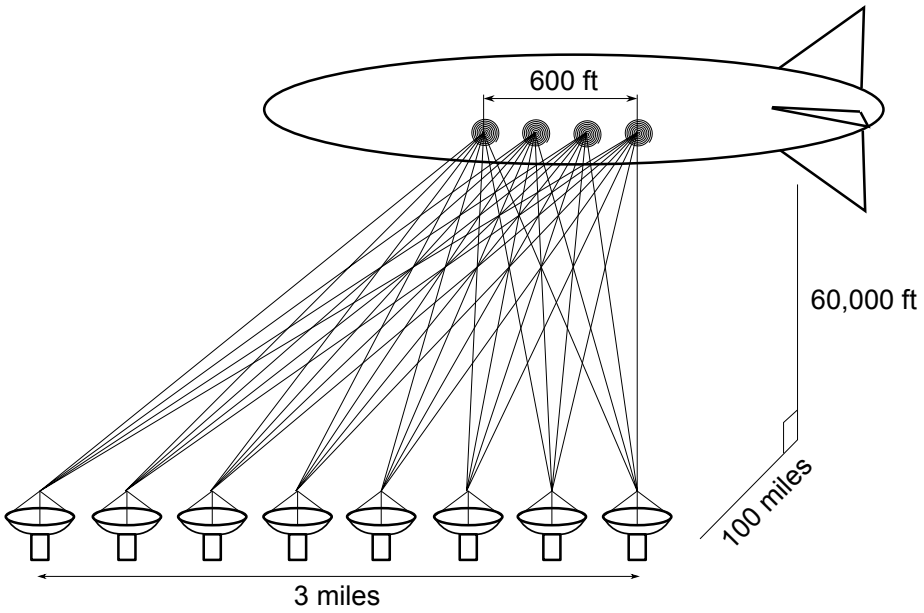


Figure 3.1: Physical representation of MIMO transmission pathways

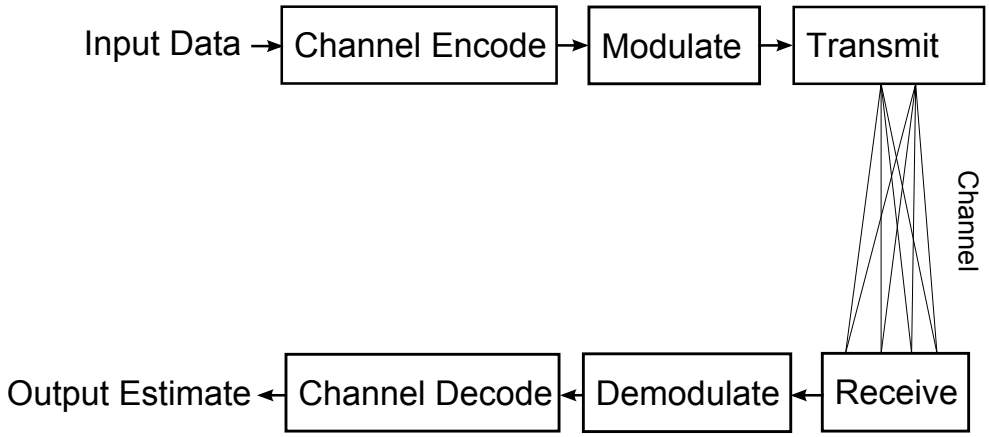


Figure 3.2: Experimental system block-diagram

numbers of antennas were [98:98], [80:160], [62:248] and [44:352], corresponding to the above ratios. However, the [98:98] case was not examined in this study with the assumption that larger numbers of transmitters on-board the airship would be increasingly difficult to implement, with diminishing return with regard to performance. The maximum receiver

spacing was evenly divided by the number of receivers for a given configuration. In practice, this would minimize interference between individual receivers and prevent mutual coupling between receiver elements. Transmit antennas were likewise separated along the full length of the transmitter array on the airship.

Another factor worthy of consideration when addressing antennas is the orientation between the transmitter and receiver. It is assumed for the sake of experimentation that the receiver and transmitter arrays are both linear and perfectly parallel to each other. As a consequence, the impacts of antenna mis-match and incitent channel angle of arrival are not examined. Further, it has been assumed that any array phasing and mutual coupling concerns are non-issues.

3.2 Error Correction Codes

Previous research focused heavily on identifying possible R-S codes which would achieve a throughput rate of 1 Tbps. The system, as modeled, did not quite attain this throughput rate in many instances due to the fact that it was implemented with minimum number of antennas required for each ratio, discounting the impact of the FEC code rate. This section seeks to address specific codes which were expected to out-perform the R-S codes in [2]. As such, some of the higher code rates have been chosen as they are close to the top-performing R-S codes in the previous research. The decoder block of the system seeks to return demodulator output to the original data received by the encoder by means of hard-decision output.

3.2.1 Non-Uniform Interleaver.

The interleaver implemented for turbo code experiments followed the non-uniform interleaving scheme presented by Berrou et al. [7] with one minor alteration. It is again emphasized that the primary reason for interleaving in this instance is not the reduction of errors due to noisy channel conditions, but for the sake of reducing the correlation between the data and parity sequences in the decoding process. The affects of interleaving

are not examined in this study. The formulas for implementing the interleaver have been reproduced below:

$$m_r = ((129 \cdot (m + n)))_{256} \tag{3.1}$$

$$\xi = ((m + n))_8 \tag{3.2}$$

$$n_r = (([P(\xi) \cdot (n + 1)] - 1))_{256} \tag{3.3}$$

$$P(\xi) = [17, 37, 19, 29, 41, 23, 13, 7], \qquad \xi \in [0, 7] \tag{3.4}$$

In the equations above, $((\cdot))_N$ represents a modulo N operation as before, and P is a set of numbers relatively prime with 256. Data are assembled into a 256×256 matrix from left to right, top to bottom, then read into an interleaving matrix according to (3.1)-(3.4). This matrix is then read back out from top to bottom, left to right. The memory size of this interleaver (65,536 elements) ultimately determined the data block size for all turbo encoding and decoding operations.

3.2.2 Convolutional Coding.

To the greatest extent possible, maximal free distance codes as expressed in [21] and [6, p. 517] were employed. This was done as these codes are known to produce results which most closely reflect the optimally achievable BER performance for convolutional codes. Table 3.2 provides a list of the codes which were selected, along with their properties. The data block size used for convolutional code experiments was 65,520 bits as it is divisible by all base two logarithm values of the modulation schemes employed as presented in Section 3.3. The Viterbi algorithm was used to decode all convolutional codes

3.2.3 Turbo Encoding.

Following Berrou et al. [7], a RSC code with a constraint length of five and generator polynomial $[37,21]_8$ was used for both encoders, as seen in 3.3. This generator does not provide maximal free distance for its constraint length [6, p. 517], however, this may be

Table 3.2: Selected short constraint length, maximal free distance convolutional codes

Code Rate	**Constraint Length**	**Generator (Octal)**	**Free Distance**
1/3	5	25, 33, 37	12
1/2	5	23, 35	7
2/3	4	17, 6, 15	3
4/5	8	237, 274, 156, 255, 337	3

insignificant as turbo codes suffer from poor minimum distance once they are constructed [6, p. 553].

Each encoder was implemented using MATLAB®'s `poly2trellis` and `convenc` commands. The data stream was passed into the first encoder, and the parity output was saved. The data stream was then interleaved and passed to the second encoder to create the second parity stream. The original data sequence, and both parity sequences, were then passed on to be punctured and concatenated as dictated by the code rate.

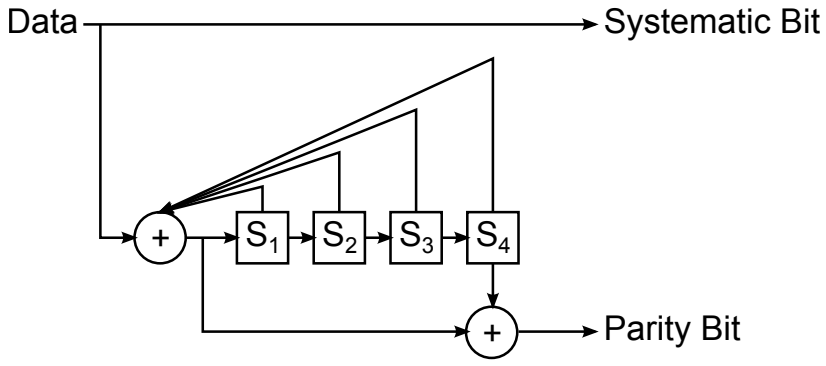


Figure 3.3: RSC encoder with constraint length of 5 and generator polynomial $[37,21]_8$

Given that, for the discussion of puncturing, a base code rate of $1/n = 1/3$ was used with puncture period T_p, matrices of dimension $n \times T_p$ were constructed to define puncturing behavior, based on the relationships in (3.5). These code rates were selected based on the

data block size being utilized by the interleaver, as they are all achievable given the prime factorization of the interleaver memory size (65,536 bits total as mentioned previously) [6, p. 521].

$$R_p = \frac{T_p}{nT_p - N}, \quad N \in [0, (n-1)T_p - 1] \qquad \mathbf{P} = \begin{bmatrix} p_{1,1} & p_{1,2} & \cdots & p_{1,T_p} \\ p_{2,1} & p_{2,2} & \cdots & p_{2,T_p} \\ \vdots & \vdots & \ddots & \vdots \\ p_{n,1} & p_{n,2} & \cdots & p_{n,T_p} \end{bmatrix} \tag{3.5}$$

The value of N in (3.5) represents the number of bits deleted out of the total number of bits generated by the encoder. Example puncturing matrices with turbo codes in mind are below. For experimentation, the rate ⅓ code was punctured to rate ½, using (3.6). The relevant puncturing matrices are given in (3.6)-(3.8). The generated data stream was read into the first row of each matrix, and parity streams into subsequent rows, T_p at a time. Coded bits were read back out top to bottom, left to right, but only if the value in the puncture matrix was one. Each location a zero is present in the matrix, represented a bit to be deleted from the final concatenated code.

$$\mathbf{P}_{1/2} = \begin{bmatrix} 1 & 1 \\ 1 & 0 \\ 1 & 0 \end{bmatrix} \tag{3.6}$$

$$\mathbf{P}_{2/3} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \tag{3.7}$$

$$\mathbf{P}_{4/5} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{3.8}$$

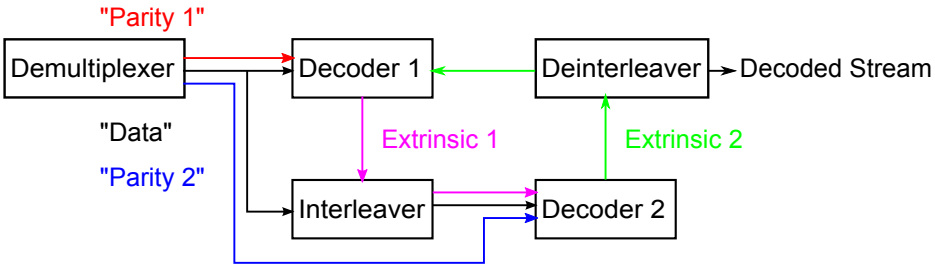


Figure 3.4: Iterative decoding loop for turbo decoding

The puncture matrix for the rate ½ code generally takes advantage of the redundant parity provided by the decoder. The puncturing for the rate ⅔ code achieves this to a lesser degree, but the pattern for the rate ⅘ code looks, by mere inspection, as if it will not perform much better than transmitting the data uncoded. It should be noted that in the case of both of the latter two matrices that a shorter puncture period could have been utilized (by only drawing from the first parity sequence, or the second row of the matrix, in each case), but doing so would destroy the benefit of turbo coding in the first place, and reduce the codec to a suboptimal convolutional scheme. Iteratively decoding such a scheme would produce highly correlated results from iteration to iteration, would only reinforce the first decoding step in each cycle, and would be akin to having a "yes-man" in the circuit.

3.2.4 Turbo Decoding.

The process of decoding a turbo code is arguably the most involved of any presented anywhere else in this document. Significant measures must be taken to ensure that computational overflow is avoided, data and parity sequences are matched, and the correct metrics are calculated for the specific encoder used to encode the data in the first place. The iterative decoding loop has been reproduced in Fig. 3.4. This begins with a discussion of the multiplexing and concatenation of the encoded data.

3.2.4.1 Demultiplexing and Puncture Removal.

The first step in the turbo decoding process was to separate the received signal back into its constituent data and parity sequences. For the base rate ⅓ code, this simply meant placing every third bit into a different bin. However, not all parity values which are generated are required to be transmitted for the BCJR algorithm to work effectively. This allowed generated codes to be punctured from base rate ⅓ to other desired code rates as described previously. The parity streams, y_{k1} and y_{k2}, were filled with zeros in appropriate locations to account for this puncturing. Following a single bit through (2.8), (2.10), (2.12) and (2.5) shows that there is no overall impact to the BCJR algorithm output for the bit in question from this zero value. In effect, a punctured convolutional code may be decoded with the BCJR algorithm, but the loss of parity introduces additional errors. But this is generally true of the overall impact of increasing the code rate, anyway.

3.2.4.2 Iterative Decoding.

Once the demodulated signal has been demultiplexed, a few operations which must only be performed one time were accomplished: An interleaved duplicate of the data stream, x_k, was stored in memory along with a L_k^c based on (2.17) for later use in the decoding cycle.

Each decoder computed its own estimate of the originally encoded sequence using the branch and state metrics defined in Section 2.3.7.1. The initial trellis state for $\bar{\alpha}_k$ was assumed to be the all zero state. The final state for the $\bar{\beta}_k$ computations was assumed uniformly likely. It should be noted that this decoding process is very slow as each decoding step depends on the output from the previous one, either going forward in time for $\bar{\alpha}_k$, or backward in time for $\bar{\beta}_k$. These values could have been conducted in parallel by segmenting the data and parity sequence, however the cost of doing so would have been the introduction of more error since the initial trellis state of each segment would have had to have been assumed or defined as equally likely.

Table 3.3: Branch and State Metric Computations

1. Compute $\delta_k^{i,s}$ $\forall$ k of the received encoded message block using (2.8)
2. Set $\bar{\alpha}_{k=1}^{s}$, set $\aleph_{k=1}$ (or compute using (2.11)), and save both
3. Using (2.10), compute α_k^s $\forall$ s for the next (forward in time) k
4. Compute $\aleph_k$ for this value of k using (2.11) and save it
5. Use (2.13) to compute $\bar{\alpha}_k^s$
6. Repeat steps 3-5 for all remaining values of k
7. Set β_{K+1}^{s}, and scale by $\aleph_K$
8. Compute (backward in time) remaining values of β_k^s and $\bar{\beta}_k^s$ using (2.12) and (2.13)

Each decoder required three inputs: the data sequence, interleaved or not, a parity sequence, and an a priori likelihood of the decoded bit's value. For the first pass through the first decoder, it was assumed that ones and zeros were transmitted with equal probability. For the second decoder, the soft decision LLR from the first decoder serves as this a priori likelihood. This "extrinsic" likelihood from each decoder was interleaved or deinterleaved as appropriate, and passed as input to the next decoder until four complete decoding iterations were concluded. At this point, the final estimate of the original data, $\hat{d}_k$, was computed using (2.17) with the previously computed L_k^c, L_k^a as the extrinsic value from the previous decoder, and L_k^e as the final decoder's extrinsic output. The decoding process, as compiled from [7], [10], and [5], is broken down, step-by-step in Tables 3.3 and 3.4.

3.2.4.3 Overflow Prevention.

For higher order QAM the LLR values produced by the demodulator, which are required input into the turbo decoder, vary wildly within each symbol. (Computation of these LLRs will be discussed in Section 3.4.) On frequent occasion, the first bit in a symbol

Table 3.4: Iterative Decoding Process

Preliminary Calculations

1. De-multiplex the received sequence into x_k and y_k (minimum of y_{k1} & y_{k2}).
2. Add zeros to each y_k to account for puncture "holes" as necessary
3. Interleave x_k and store it ($\acute{x}_k$).
4. Compute and store L_k^c.

Decoding Iterations

1. Feed x_k, y_{k1}, and Λ_k^a into decoder 1
2. Compute $\delta_k^{i,s}$ using decoder 1 input using (2.8)
3. Set $\bar{\alpha}_1^1 = 1$ and $\bar{\alpha}_1^{s\neq 1} = 0$ (all other states)
4. Initialize $\bar{\beta}_{K+1}^s$ to uniform probability across all states.
5. Compute $\bar{\alpha}_k^s$ and $\bar{\beta}_k^s$ (see Tab. 3.3)
6. Compute Λ_k^e from decoder 1 using (2.16), interleave ($\acute{\Lambda}_{k1}^e$), pass to decoder 2
7. Repeat steps 1-6 for decoder 2 using $\acute{x}_k$, y_{k2}, and $\Lambda_k^a = \acute{\Lambda}_{k1}^e$.
8. De-interleave extrinsic liklihood ($\Lambda_{k2}^e \rightarrow \grave{\Lambda}_{k2}^e$)
9. Set $\Lambda_k^a = \grave{\Lambda}_{k2}^e$
10. Iteration complete, repeat steps 1-9 for requisite number of iterations

Final Decoding Step

1. Compute $L_k^e = \log(\grave{\Lambda}_{k2}^e)$ from the final iteration of decoder 2
2. Use (2.17) to compute the final $L(\hat{d}_k)$ from the decoding process
3. Make hard decisions on $L(\hat{d}_k)$ using (2.18)

would have a very high LLR while the next bit in the signal would have a LLR a few orders of magnitude less than that of its neighbor. The iterative decoding process seeks to use both decoders to strengthen the LLR of a given bit, and thus, upon receipt of an already large LLR many times the branch metric or state metric would become too large to be represented as a double precision integer. The largest value MATLAB® can handle without overflow

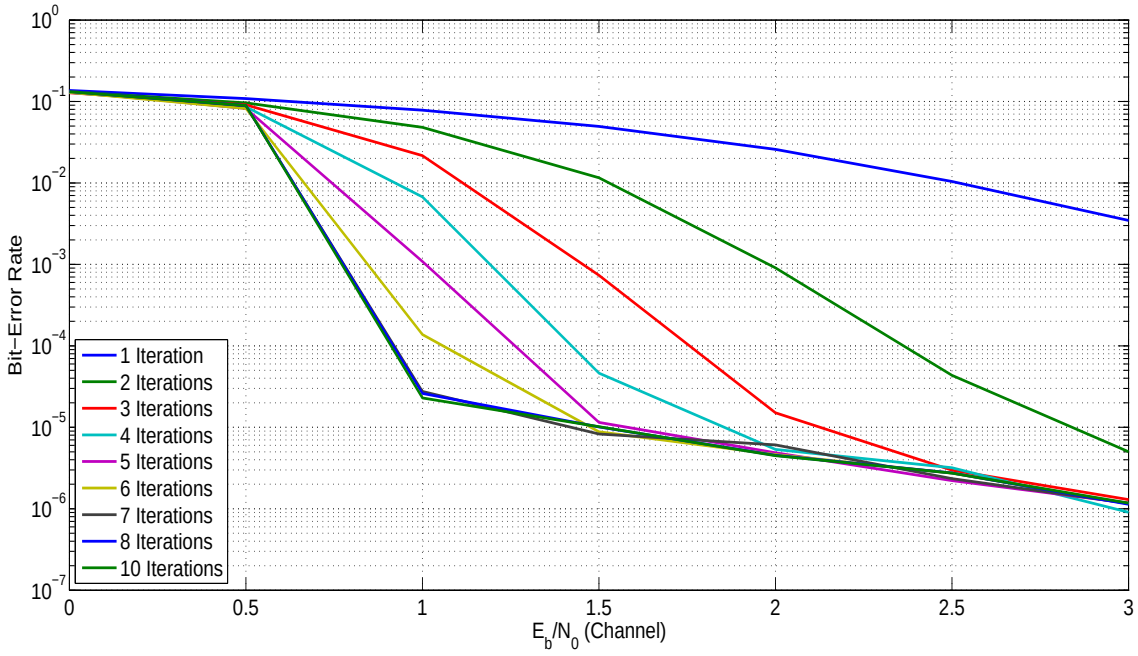


Figure 3.5: Performance of a rate ½ turbo code comprised of two $K = 5$ rate ½ RSC codes with generator $[37, 21]_8$

is on the order of 1×10^{308} and is slightly larger than the largest value in the Institute of Electrical and Electronics Engineers (IEEE) double precision standard (represented in code by `realmax`) [22]. Numbers larger than this are represented as ∞, and naturally produce computation errors when used with fractions or zeros. Not surprisingly, division by zero will produce ∞. It was assumed that the cause of a $\pm\infty$ branch metric value was the result of exceeding the double-precision standard due to the exponential calculation in (2.8), and as such, all occurrences of ∞ were replaced by 1×10^{300}.

3.2.5 Turbo Coder Performance.

Figure 3.5 shows the performance of the previously described turbo code as it is iteratively decoded for multiple iterations. The turbo decoder which was implemented for the overall system cycled through four decoder iterations. This was considered a reasonable

trade-off between the perceived point of diminishing returns with regard to increasing the number of decoding cycles, and the computational complexity of increasing that number of cycles. A noticeable characteristic of turbo codes is the error-floor that they experience due to their low minimum distance [6, p. 551]. Alternative theories about the cause of this phenomena center around interleaver design and the failure of the redundant decoders to converge on the same estimate of the original transmitted bit [10, pp. 261-262]. Regardless of its cause, the error correcting capability of this specific rate ½ code tapers off near 1-1.5 dB in the channel.

3.2.6 Low-Density Parity Check Codes.

The LDPC codes implemented for evaluation with this system were the Second Generation Digital Video Broadcasting Standard (DVB-S2) rate ⅓, ½, ⅔, and ⅘ codes. DVB-S2 codes make use of a $[{}^{k}/_{n} \cdot 64,800 \times 64,800]$ sparse parity matrix which contains ${}^{n}/_{k} \cdot 64,800$ parity bits, the majority of which are zero. The output length of the encoded data is fixed at 64,800 bits, and the parity matrix and number of parity bits add to make that size. The parity matrix only contains five to 8 bits per row depending on the code rate, as defined by the DVB-S2. Theses codes were implemented in MATLAB® using the `dvbs2ldpc` command [23], and nothing like the error floor described above was encountered in testing the individual LDPC.

LDPC codes are currently the primary competitor to turbo codes, a comparison of the $R = {}^{1}/_{2}$ turbo used in this study, decoded with ten iterations, and the DVB-S2 LDPC code for low SNRs is show in Fig. 3.6. The data points presented were obtained by generating random bits, BPSK modulating them, and decoding until a minimum of 500 errors came out of the simulation, for each code. This was not intended to be a universal comparison between the two coding types, but simply a reference point for later comparison of two of the codes used in this study. A less precise characterization of each kind of code examined is given in Fig. 3.7.

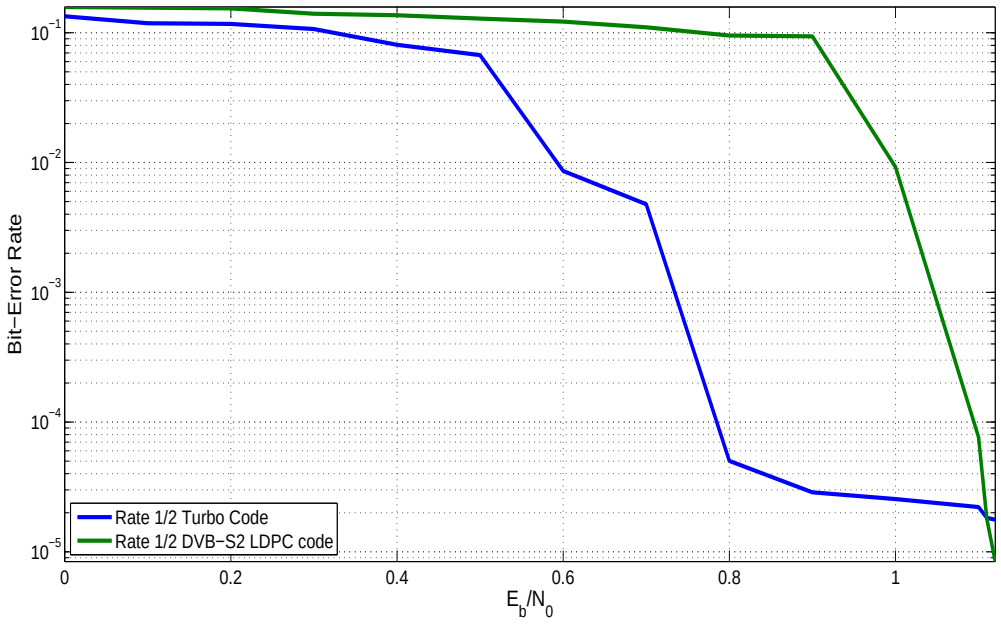


Figure 3.6: Comparison of rate ½ turbo and LDPC codes implemented for experimentation

3.3 Modulation

Because bandwidth and carrier frequency are fixed, increases in throughput were expected to come through utilizing strong FEC codes to enable the use of larger signalling constellations. Simply stated, the goal with modulation was to try to find the largest constellation feasible while maintaining the necessary BER performance as specified previously. BPSK was an obvious starting point as most error control codes are analyzed via BPSK. Furthermore, it is the most "reliable" since all of the energy in one symbol is dedicated to one bit. If a code did not meet requirements with BPSK, it was assumed that it would not work with a larger constellation.

Many of the signalling constellations from previous work in [2] were rebuilt for efficiency using MATLAB®'s `modem` toolbox, with the aim of obtaining soft demodulator output instead of the hard decisions that the previous system produced. The reason for this

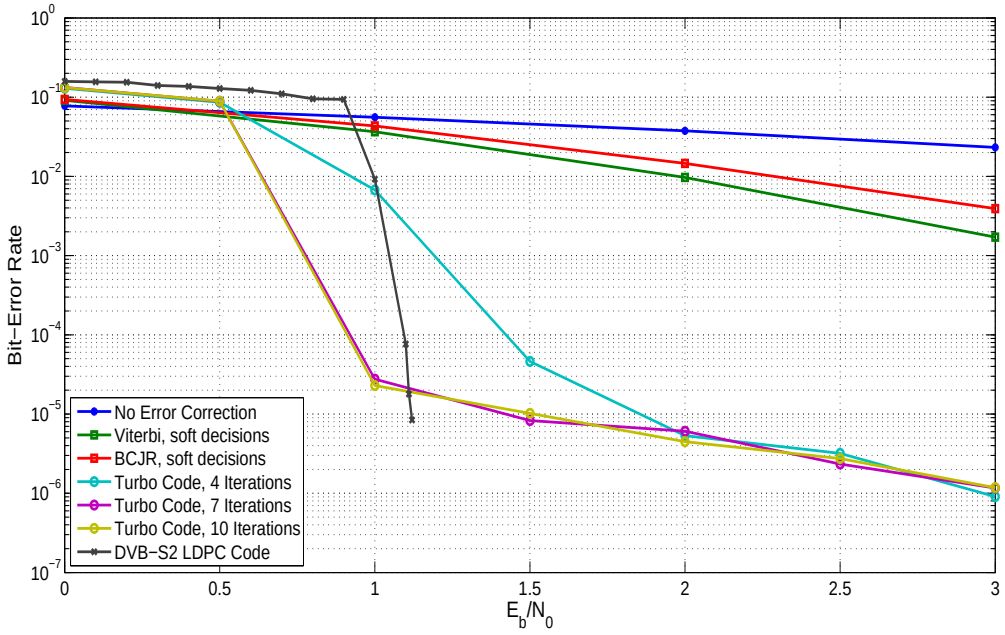


Figure 3.7: Comparison of uncoded BPSK to rate ½ codes implemented

is twofold: soft decision demodulation allows for higher fidelity in the decoding process, and both LDPC codes, and turbo codes require a soft input to function properly. Octal-PSK was of interest because it was the standard for satellite communications for quite some time. Even though the airship does not operate at anything near orbital altitudes, this modulation type warrants evaluation, if for no other reason than the ease of acquiring the equipment, were this system actually to be built. Higher order QAM was also used so that performance limits could be assessed.

Transmitter power was normalized to 1 W per transmitter by dividing the modulated waveform by the average energy in the signalling constellation. For rectangular QAM, this normalizing factor is represented by (3.9).

$$E_{av} = \frac{2(M-1)}{3} \tag{3.9}$$

Table 3.5: Average signal energy in MATLAB® signalling constellations

Modulation type	E_{av} **(W)**	E_{av} **(dBW)**
BPSK	1	0
4-QAM	2	3.01
8-PSK	1	0
16-QAM	10	10.00
32-QAM	20	13.01
64-QAM	42	16.23
128-QAM	82	19.14
256-QAM	170	22.30

Additionally, when the data sequence input into the modulator was not integer-divisible by the number of bits in a modulated symbol, block filling was accomplished with values most likely to produce a constellation point least susceptible to noise for the number of bits which had to be added. Block-filling patterns and examples of where these fall on the constellation are shown in Appendix B. Following demodulation, these bits were removed from consideration.

3.4 Demodulation

For the sake of reducing overall system complexity, physical-layer waveforms were not generated for transmission, and it was assumed that the demodulator effectively handled carrier synchronization, down-conversion, baseband filtering, and symbol constellation point mapping. From this point, soft decisions, in the form of LLRs, were generated by the demodulator. The impacts of antenna gain and directivity were not modeled, and were assumed to have been accounted for by the SNR at the receiver.

As stated before in Section 3.2.4.2, it was assumed that source formatting, encoding, and encryption were accomplished in a manner which produced equally likely message symbols, prior to entering the channel encoder block of the system. It was likewise assumed that throughput and BER metrics can be computed before decryption, source decoding and receiver formatting are accomplished as these aspects of a generic communication system are not the focus of this study. The major system blocks to be implemented were shown previously in Fig. 3.2.

The soft output of MATLAB®'s `demodulate` command generates a LLR given in basic form as

$$L(b) = \log\left(\frac{\Pr(b = 0|r = (\mathrm{Re}[r], \mathrm{Im}[r])}{\Pr(b = 1|r = (\mathrm{Re}[r], \mathrm{Im}[r])}\right), \tag{3.10}$$

where b represents an individual bit and r is the received modulated sequence, and derives its basis from MAP decoding of convolutional codes. The specific calculation accomplished, based on [24], is [23]

$$L(b) = \log\left(\frac{\sum_{r\in R_0} \exp\left[-\frac{1}{\sigma^2}\left((\mathrm{Re}[r] - \mathrm{Re}[R])^2 + (\mathrm{Im}[r] - \mathrm{Im}[R])^2\right)\right]}{\sum_{r\in R_1} \exp\left[-\frac{1}{\sigma^2}\left((\mathrm{Re}[r] - \mathrm{Re}[R])^2 + (\mathrm{Im}[r] - \mathrm{Im}[R])^2\right)\right]}\right). \tag{3.11}$$

This soft decision of received bits was then used as input into the decoder. This was desired due to the fact that both turbo decoders and LDPC decoders require soft input.

Equation (3.11) is shown in [24] to be approximately equal to (3.12) below. This method is used to compute channel values as it is less computationally intensive than using (3.11) and is less likely to create computational overflow when computing large or small exponential values.

$$L(b) = -\frac{1}{\sigma^2}\Big(\min_{r\in R_0}(\mathrm{Re}[r] - \mathrm{Re}[R])^2 + (\mathrm{Im}[r] - \mathrm{Im}[R])^2 - \min_{r\in R_1}(\mathrm{Re}[r] - \mathrm{Re}[R])^2 + (\mathrm{Im}[r] - \mathrm{Im}[R])^2\Big) \tag{3.12}$$

In (3.11), and (3.12) above, the soft-decision bit-stream LLR, $L(b)$, is implemented with the opposite decision rules than what have been discussed until now. Consequently, the

sign of either of these equations' output must be changed before use. In both cases, r is the received complex-valued signal, R is the expected location of an ideal symbol on the complex constellation map, and σ^2 is the variance of the noise.

3.5 Channel

As an update to the system model from previous research, a new channel was generated for each experiment: The previous system model assumed channel conditions did not change for the duration of each experiment (e.g. the MIMO channel matrix implemented for the duration of a given R-S code with octal-PSK throughput vs. BER computation did not vary until a new experiment was begun). This had the effect of treating the channel as if it was time invariant. While it is acceptable to do this for simulations, the ultimate goal is to take measured data and account for varying channel conditions.

3.5.1 MIMO Cross-Talk and Noise.

In order to achieve the desired SNR, noise power, σ_n^2, was scaled according to meet the conditions of (3.13) in the channel. As mentioned previously, the transmitter power P_t for each transmitter was normalized based on the constellation, implying that σ_n^2 be set to -4.77 dBW. With this in mind, the overall system SNR was achieved using

$$SNR = \frac{\sum_{n=1}^{N_t} P_{t,n}}{N_t \sigma_n^2} \, . \tag{3.13}$$

Adjusting (2.36) for noise at the receiver, $\mathbf{H}^{\dagger}$ becomes

$$\mathbf{H}^{\dagger} = \left(\mathbf{H}^H \mathbf{H} + \frac{N_t}{SNR} \mathbf{I} \right)^{-1} \mathbf{H}^H \, . \tag{3.14}$$

Equation (3.14) was defined in [2] as an implementation of a MMSE equalizer, and was applied to the received signal to separate out the original, individual transmissions using

$$\hat{\mathbf{x}} = \mathbf{H}^{\dagger} \mathbf{y} \, . \tag{3.15}$$

3.5.2 Fading and Multipath.

A slow fading channel was been assumed so that an entire block of data for any of the coding schemes could be transmitted before requiring updated channel coefficients. This assumption is generally made in practical systems, as the throughput cost incurred in estimating the channel too frequently can result in system shortfalls. As such, the channel coefficients in the simulated model were updated after each transmission cycle. For multipath rays, it was assumed that the LOS pathway experienced Ricean fading while each multipath ray experienced Rayleigh fading. The effects of local scatters and geographic features were ignored, and multipath rays were assumed to only come from ground-reflection.

3.5.3 Two-Ray Model.

The multipath ray generated to interfere with the main signal transmitted was modeled based on the distribution in (2.28). The individual distances between each transmitter and each receiver were computed and used to determine the delay between the original transmission, and the arrival of a multipath ray. The delay was computed as in Section 2.7.1, without using the small angle assumption. The LOS transmission was passed through a Ricean fading channel by applying (2.28) with $\sigma = 0.1$. The multipath ray was passed through a Rayleigh fading channel by applying (2.27) with $\sigma = 0.001$.

3.5.4 Tapped Delay Line Model.

The TDL channel model, referred to as the "five-ray" model in previous work determined that in a given sampling period, the oldest multipath component to be received, based on the longest transmission pathway to the receiver, was transmitted four samples before the present time sample. This was determined by computing the longest ground-reflection pathway from transmitter to receiver, and implemented by summing the effects of four previous, Rayleigh-faded, multipath components with the Ricean-faded LOS transmission. This delay model was verified and carried forward into this investigation,

so that a single LOS transmission was summed together with four delayed multipath rays. This approach is called the layered reception technique in [20, pp. 255-256]. The LOS and multipath rays were faded as in the previous section.

3.6 Experiment

Monte-Carlo simulations were designed so that a minimum of 1×10^8 randomly generated bits were passed through the system in order to be able to resolve BERs as small as 1×10^{-8}. Additional information regarding these precise numbers may be found in Appendix B. This was accomplished by looping through the system until the requisite number of bits had been passed through the channel, estimated, and compared to the original data. All system blocks in Fig. 3.2 were implemented in MATLAB® individually, and chained together for a single loop of the experiment. For the first pass through the system, the modulator, demodulator, encoder, and decoder were generated and saved. For subsequent passes, these components were pulled from memory and reused. The channel matrix was regenerated for each pass through the system. Upon completion of the simulation, performance metrics were gathered.

3.6.1 Throughput.

One of the chief goals of implementing the codes and MIMO is to improve the throughput of the overall communication system. Many will be familiar with the concept of throughput as it applies to computer architecture, measured in units such as floating point operations or instructions processed in some unit time [25, p. 530], but here it is defined as the amount of data passed through the communication system in bits per second (bps). Digital media are the contemporary standard for video data, and quantization issues are handled at or near the detector, hence it is assumed that data exist in digital form prior to entering the discussion.

The term "data" itself can be political in nature as well: Streamed data typically represent only a subset of the original data block, and may be something entirely different

than what is actually recorded to the detector's physical storage media — that is, if data are not received, processed, and relayed without any intermediate storage. Processing rules on either end of the system may dictate which pieces of data are transmitted in the first place, which are stored, and which are discarded. Therefore, throughput computations were only based on data selected for transmission, ignoring the fact that source coding and encryption typically modify the original data size. Referring to Fig. 3.2, "data" are collectively meant to be the digital elements entering the encoder block, and exiting the decoder. All transformations and permutations in-between were considered to be artifacts of the encoding and modulation process, and were accounted for in the following equation for throughput:

$$\phi = W \times \tfrac{k}{n} \times M \times N_t \, . \tag{3.16}$$

As before, M is the modulation constellation size, k/n is the code rate, and N_t is the number of transmitters. The available bandwidth for transmission is given by W. For a system's configuration to be acceptable, a minimum throughput of 1 Tbps must be met. From a quick glance at (3.16) it is obvious that one of the traditional throughput variables, Effective Isotropic Radiated Power (EIRP) has been ignored. This is intentional, as the impacts of the link are being addressed in the system's error performance as it pertains to the fixed SNR of 4.77 dB at the receiver, and other parameters given in Table 3.1.

3.6.2 Error Performance.

The simplest calculation of BER is given by (3.17) in which the numbers are assumed to be the totals from the entire experiment. As previously stated, an acceptable BER is 1×10^{-5}, and systems performing at a higher BER than this did not meet design specifications. Data were generated randomly with the assumption that the probability of a one or a zero was equally likely, and demodulated and decoded with this assumption in mind. BER was used as one of the primary discriminants for advancing to a higher

constellation size for modulation in many cases.

$$BER = \frac{\#\ errors}{\#\ total\ bits} \tag{3.17}$$

3.7 Conclusion

This chapter has provided a system model for simulating an large MIMO array with focus on FEC, modulation, and channel parameters. Design heuristics have been addressed and the information necessary to repeat experiments has been presented. At least one simulation was completed for each combination of code, antenna configuration, and multipath model, by means described in Section 3.6. Results from the experiments required by this chapter are presented in Chapter 4.

IV. Results

RESULTS from Monte Carlo simulations are presented in this chapter with the aim of demonstrating improved system performance, despite failure to achieve the goal of modeling a high-altitude 1 Tbps MIMO communication system. A subset of results are discussed, and the reader is referred to appendices for the full set of metrics worthy of mention.

4.1 Antenna Configurations

In each of the figures presented in this chapter, data points sitting on the 1×10^{-8} do not actually hold this value. These experiments did not produce any errors despite the minimum number of bits being passed through the system (1×10^8 per Section 3.6). For the sake of plots, they have taken on the smallest measurable BER value. Differences between their actual values cannot be inferred as much larger sample sizes are required to generate the necessary errors from simulation to do so. Results for modulation and FEC code providing at least 500 Gbps of throughput have been presented in Tables 4.1-4.3, and plotted in Figs. 4.1-4.6. By simple count, it appears that the [62:248] configuration was capable of the best overall performance, with the [44:352] configuration following closely behind. The truth is that the latter was not ever able to improve the absolute system performance over previous research. Since it could only improve relative to the same modulation possibilities, the second-place configuration is actually the [80:160] configuration, based purely on throughput. Figures 4.7-4.12 have been constructed to illustrate this fact; they only show results which absolutely improve over the previous system.

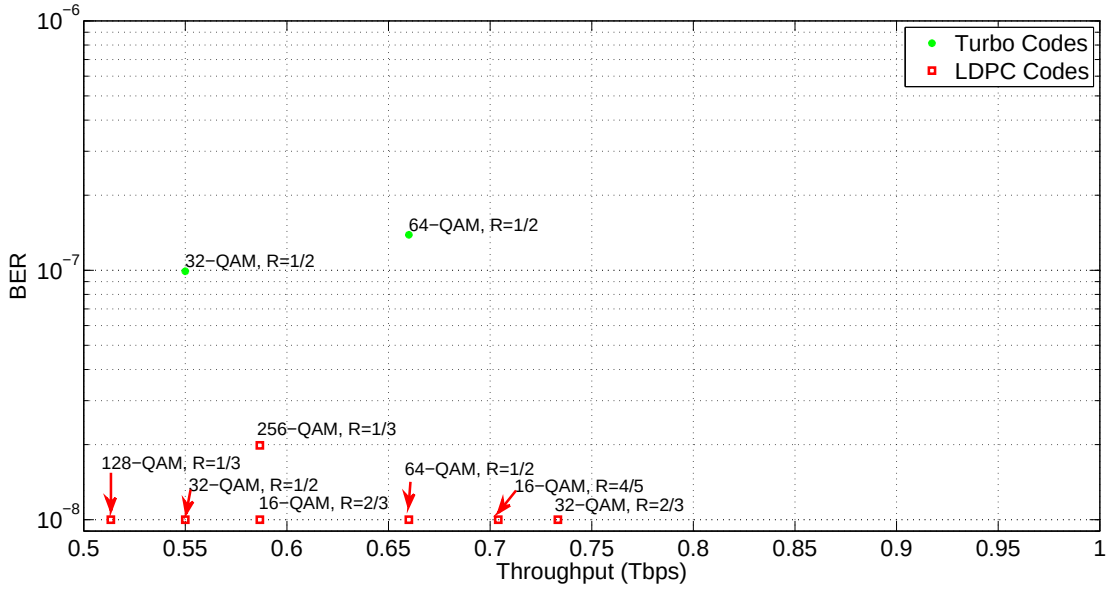


Figure 4.1: All code and modulation combinations producing 500 Gbps when applied to the [44:352] antenna configuration with two-ray multipath

4.2 Coding Methods

4.2.1 Convolutional Codes.

Convolutional coding have been proven through the course of their history, and as expected, their use consistently allowed for the system to operate at or above the minimum BER. They frequently performed so well in terms of BER that they could not be distinguished from their turbo and LDPC counterparts for small modulation constellation sizes. Additional characterization with larger samples sizes would likely eventually show convolutional codes to be inferior to the other two code types employed, but a few cases which can be seen in Figs. 4.2, 4.4, and 4.5 even allowed for relatively high order QAM to be used without exceeding maximum allowable BER. The best performing convolutional code was the rate ½ code applied to the [62:248] antenna array with 16-QAM, which

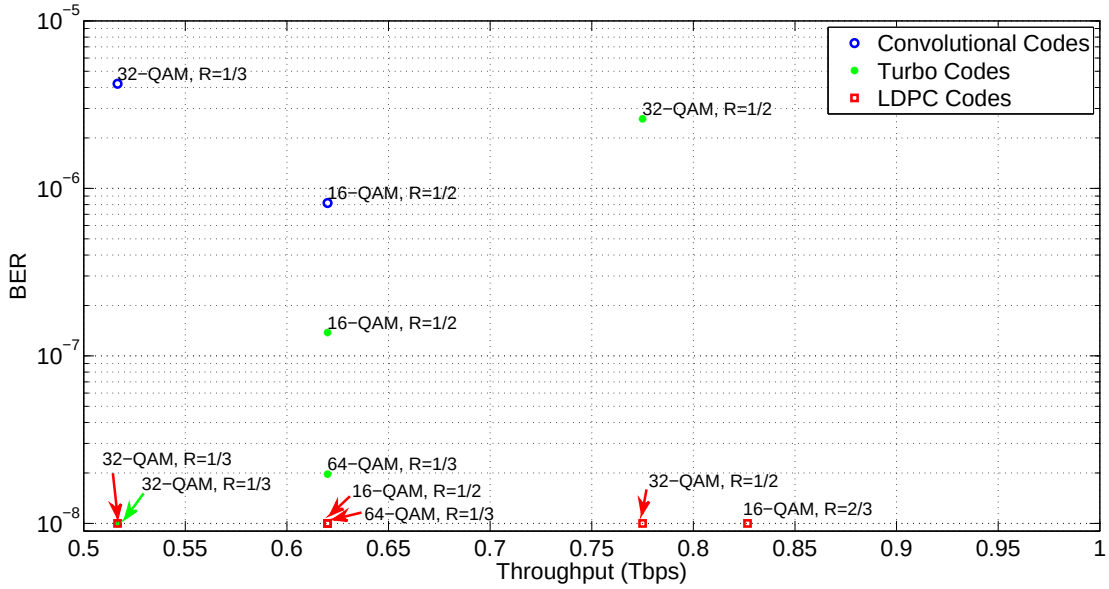


Figure 4.2: All code and modulation combinations producing 500 Gbps when applied to the [62:248] antenna configuration with two-ray multipath

showed a best-case BER of 8.17×10^{-7}, but a throughput of merely 0.62 Tbps (see Fig. 4.2 or 4.5).

4.2.2 Turbo Codes.

Turbo codes demonstrated excellent performance as expected, but they were unable to provide enough coding gain so as to allow for large enough modulation constellations to achieve 1 Tbps while maintaining the minimum BER standard. The top performing code, modulation, and antenna configuration combination was the rate ½ code applied to the [62:248] configuration with 32-QAM, which produced a BER of 2.6×10^{-6} and a throughput of 0.78 Tbps in the best case. The effort invested in generating and decoding these codes produced results allow for this specific configuration to be an acceptable option for absolutely out-performing the previous system, using the code and modulation combinations seen in Figs. 4.8 and 4.11. Making use of additional decoding iterations may

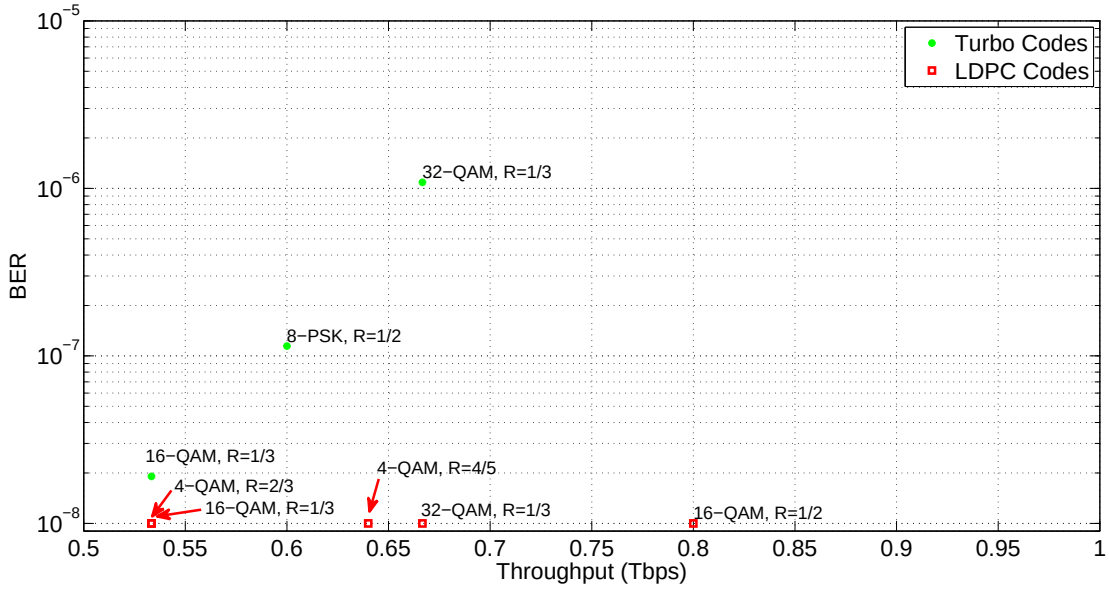


Figure 4.3: All code and modulation combinations producing 500 Gbps when applied to the [80:160] antenna configuration with two-ray multipath

have availed more code possibilities for consideration, and should be investigated, but as can be seen in Figs. 4.1 and 4.4, the application of high-order QAM was possible given these codes.

4.2.3 LDPC Codes.

The LDPC codes used in this study performed extremely well, and as evident in all figures in this chapter, consistently provided the most code and modulation possibilities that met the maximum BER requirement. It is probably not a fair to compare them to the turbo codes which were used; they follow an industry standard and thus are subject to more "optimal" design conditions as well as scrutiny and collaborative input. Furthermore, the LDPC implemented by MATLAB® makes use of 50 decoding iterations before assigning hard decisions. The turbo code used here only used four. Though they are fundamentally different decoders (turbo decoders being the more complicated of the two), the disparity in

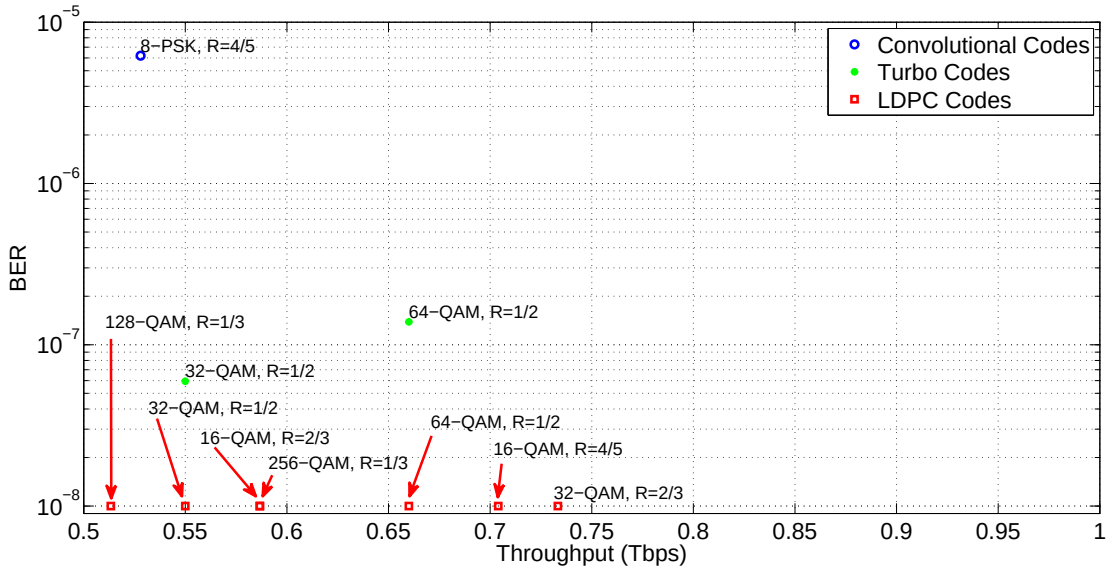


Figure 4.4: All code and modulation combinations producing 500 Gbps when applied to the [44:352] antenna configuration with TDL multipath

the number of decoder iterations is noteworthy. The majority of the system configurations which produced absolute improvement over previous research incorporated the DVB-S2 LDPC. This trend can be seen in Figs. 4.7-4.12. The best of these arrangements came from the [62:248] array using a rate ⅔ code and 16-QAM, and achieved an small, unresolvable BER of $<1 \times 10^{-8}$ at 0.83 Tbps of throughput, as shown in Figs. 4.8 and 4.11.

During the system building process, when the LDPC codes were employed, errors were not discovered in small groups of one or two per iteration, rather the more common occurrence was that if errors were to appear at all, they appeared in large groups in tens at time. Data are not presented here for further inquiry, but this was a notable difference in the performance of the LDPC code design from the turbo code design. When paired with the understanding of how the LDPC decoder works (optimizing parity equations via factor

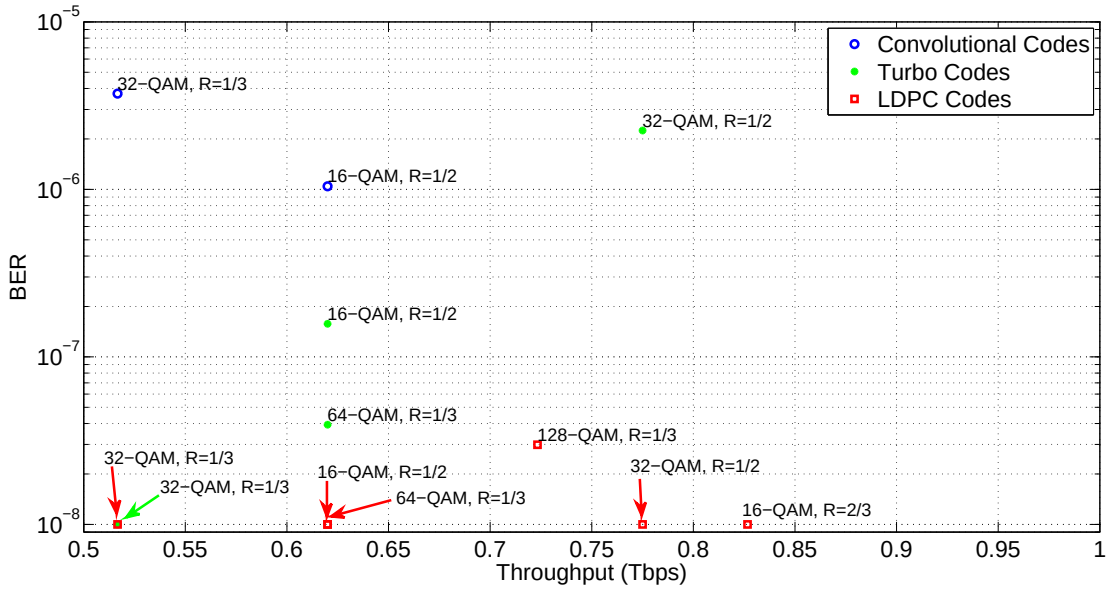


Figure 4.5: All code and modulation combinations producing 500 Gbps when applied to the [62:248] antenna configuration with TDL multipath

graphs or tanner graphs) it stands to reason as to why many bits would fail at a time: The decoding process utilizes more codependent relationships than does turbo decoding.

4.3 Anomalies

A noticeable pattern in the data is that the BER performance of 8-PSK was much worse than the next power of two constellation employed, 16-QAM. This was surprising, because the minimum Euclidean distances in the two constellations are 0.77 and 0.74 for the normalized 8-PSK constellation and the normalized 16-QAM constellation respectively (greater minimum distance implies lower frequency of error). Though this is not a huge difference, it is enough that the results should at least be similar, as opposed to the glaring difference experienced. Oddly enough, this problem seemed to be isolated to 8-PSK modulated signals encoded with LDPC codes. It is well known, however, that QAM

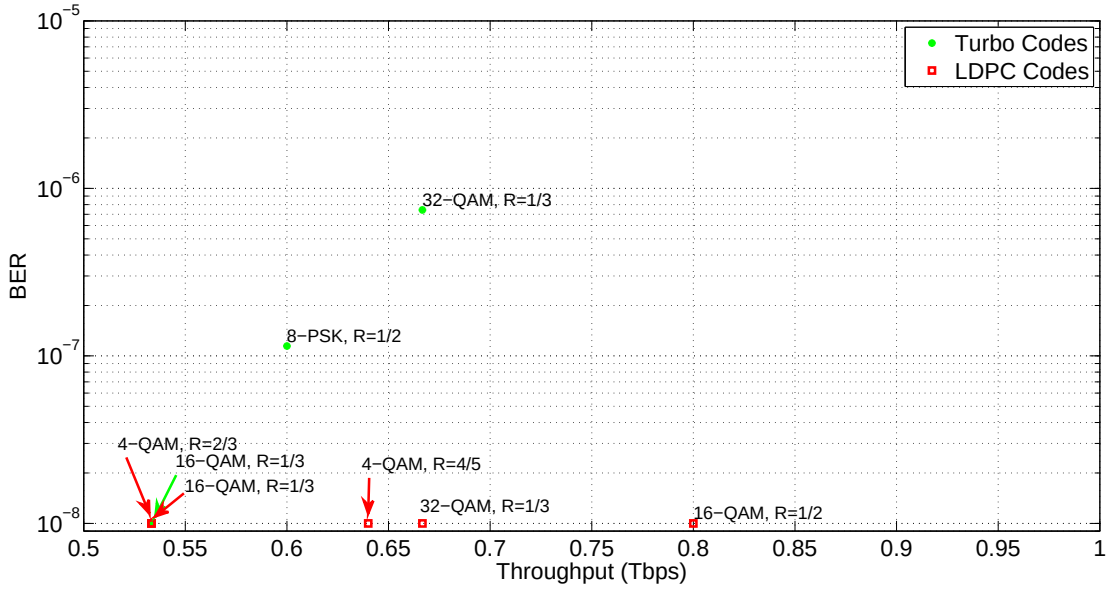


Figure 4.6: All code and modulation combinations producing 500 Gbps when applied to the [80:160] antenna configuration with TDL multipath

constellations out-perform PSK of the same size [6, p. 200], however, based on the aforementioned minimum distance issue mentioned above, it does not stand to reason that 16-QAM would out-perform 8-PSK. Further investigation of this phenomenon is warranted, as it could be something as simple as a bug in the code used to build the system, all the way up to a major incompatibility between the two for some unknown reason. This issue does not change the overall conclusion of this document: higher constellations of much higher order were implemented with better throughput results.

Another surprising trend experienced in the data involved the difference between the two-ray and TDL models for rate ⅓ turbo codes. While this code rate demonstrated the best BER performance for the TDL channel model, for the two-ray model, the rate ½ turbo codes actually out performed this lower code rate in terms of BER achieved and allowable QAM constellations. The exact reason for this remains a mystery, as repetitions of the

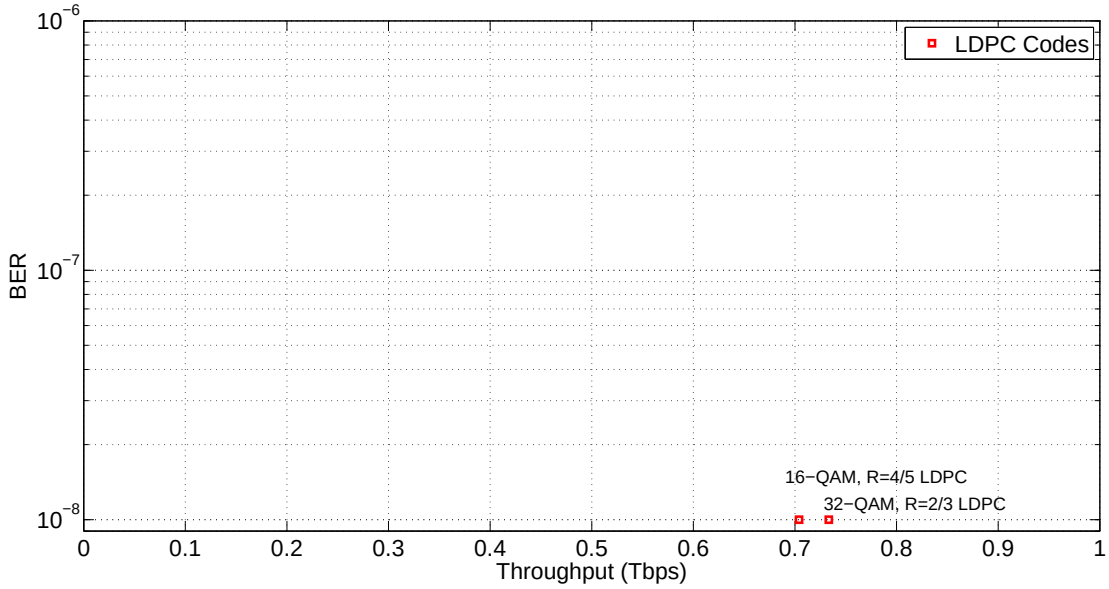


Figure 4.7: All codes showing absolute system performance improvement when applied to the [44:352] antenna configuration with two-ray multipath

experiment produced the same result. However, it may be surmised that one of the many variables related to the turbo code itself are to blame. The underlying convolutional code may have contributed through its innate characteristics. The (relatively small) number of decoding iterations the number of decoding iterations, set at four, could have prevented decoder-pair decisions from converging at this point, given the full parity set. The more likely scenario is that the impacts of multipath were mostly beneficial to the rate ⅓ code's performance; multipath improved the BER in this case.

4.4 Multipath Model Assessment

In general, the multipath effects between the two models were not significant. Some differences can be noted by comparing the tables in Appendix C. The major ones have

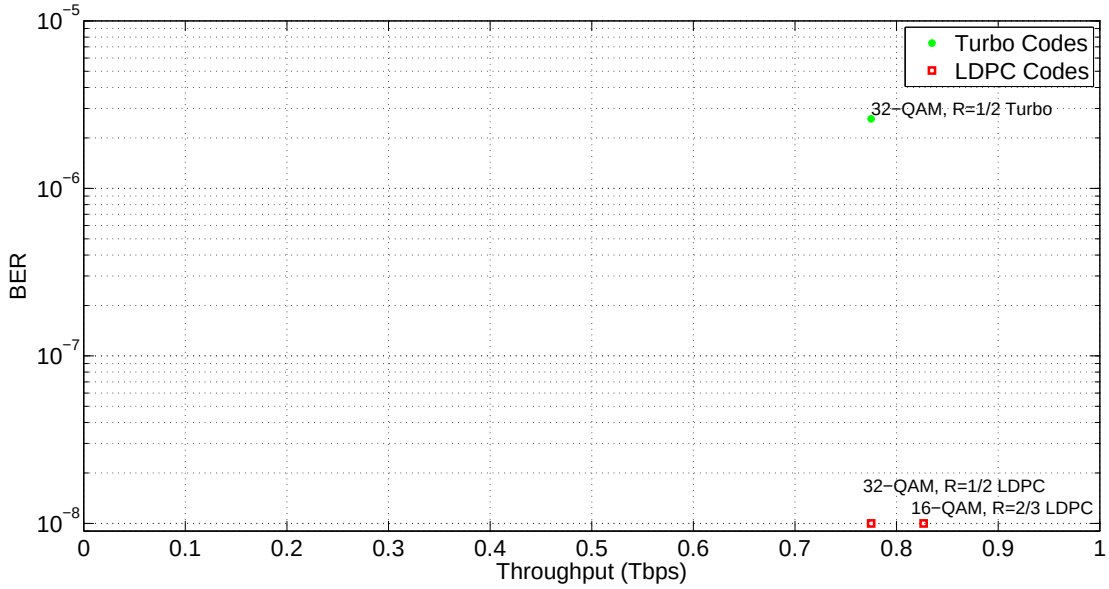


Figure 4.8: All codes showing absolute system performance improvement when applied to the [62:248] antenna configuration with two-ray multipath

been reproduced in Table 4.4. The data point in red indicates one which provided high throughput, but due to the uncertainty associated with its BER performance, it was not considered to be a case which showed absolute improvement over the previous mode, even though it appears in Fig. 4.11. With the exception of these notable differences, the rest of the results were not significant enough to warrant further discussion.

The acceptable configurations summary makes use of the worst-case BER between the two multipath models, as results between it and the two-ray ground reflection model were generally the same, even to three digits of BER in the majority of cases. There was some noticeable variation in the results for higher-order QAM modulations in general, as expected, but a few significant differences were observed in a few cases. The acceptable modulation and code combinations which showed these significant deviations are presented in Table 4.4. In one case, the variation between the two models resulted in the code being

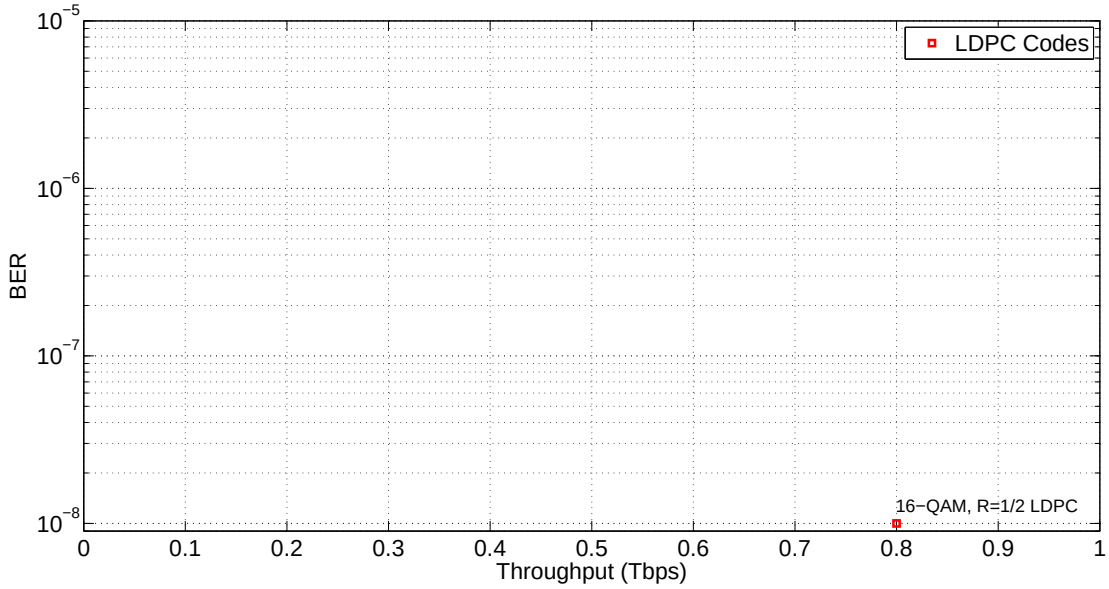


Figure 4.9: All codes showing absolute system performance improvement when applied to the [80:160] antenna configuration with two-ray multipath

removed from consideration, but not in the other. The argument can be made that the TDL method of computing multipath is more "correct" in its approach, and for this reason, the contestable configuration has not been included in the acceptable set. Additional data in Tables C.9 and C.18 can be juxtaposed to observe this occurrence.

4.5 Summary Comments

One of the most obvious trends is that the case of [62:248] antennas, representative of a 1:4 ratio, warrants the best overall results. The overall best-performing system configuration was the rate ⅔ LDPC code applied to the [62:248] system and 16-QAM. The second best overall performer came from the [80:160] configuration wherein a rate ½ LDPC was applied, and 16-QAM was modulated. Making use of the best-case turbo code implemented in this study is also reasonable, however the error floor significantly reduces overall BER performance.

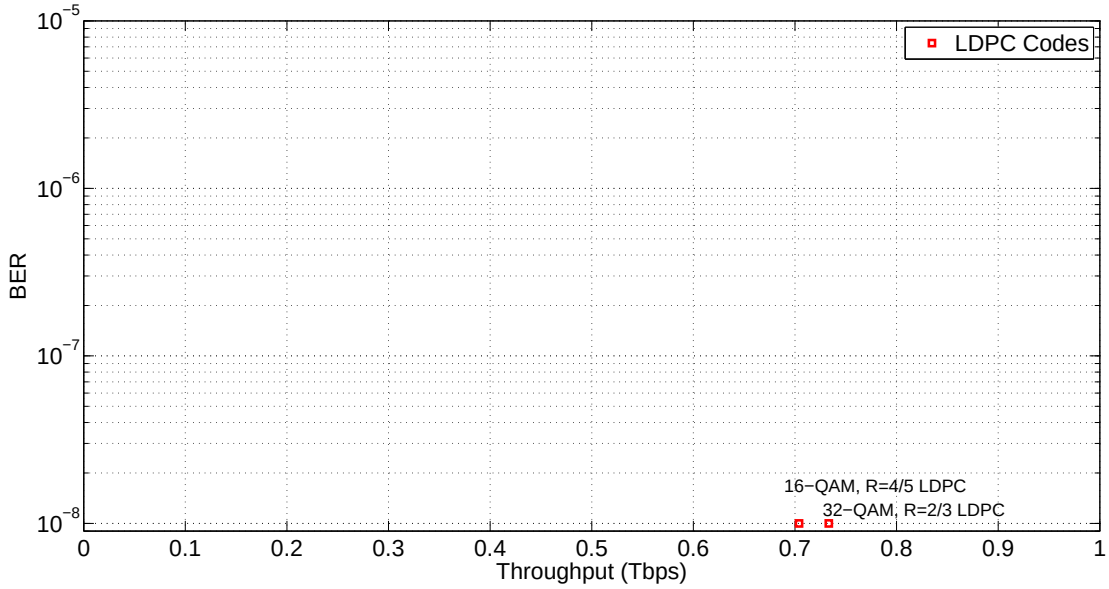


Figure 4.10: All codes showing absolute system performance improvement when applied to the [44:352] antenna configuration with TDL multipath

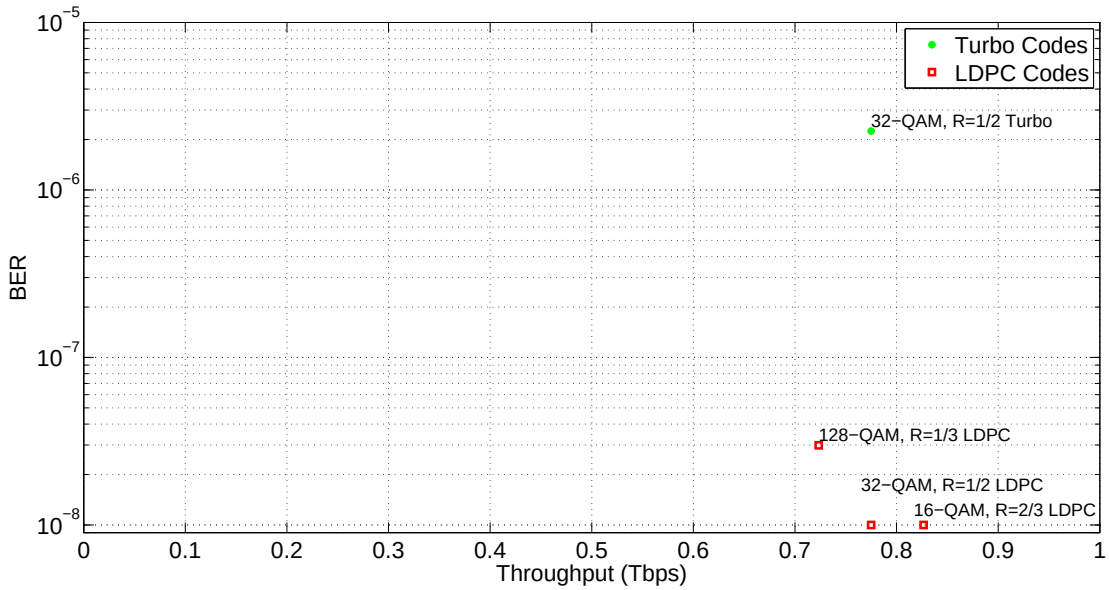


Figure 4.11: All codes showing absolute system performance improvement when applied to the [62:248] antenna configuration with TDL multipath

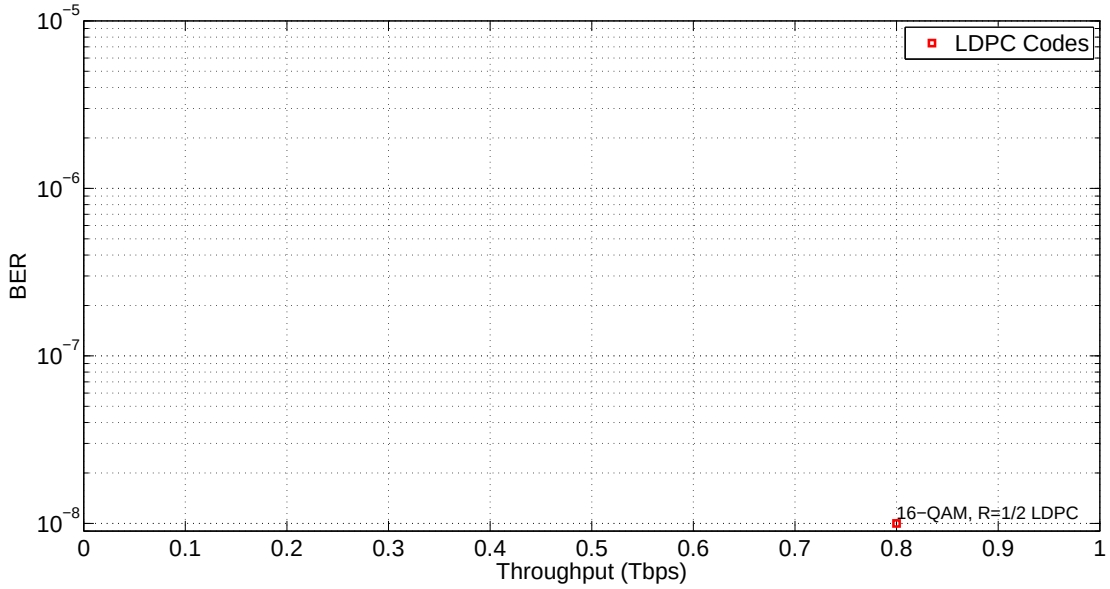


Figure 4.12: All codes showing absolute system performance improvement when applied to the [80:160] antenna configuration with TDL multipath

Table 4.1: Code and modulation combinations for $N_t = 44,\ N_r = 352$ achieving minimum throughput of 500 Gbps

Code Type	**Rate**	**Modulation**	**BER, 2-Ray**	**BER, TDL**	**Throughput (Tbps)**
Convolutional	4/5	8-PSK	6.66e-6	6.19e-6	0.52
Turbo	1/2	32-QAM	9.91e-8	5.95e-8	0.55
		64-QAM	1.39e-7	1.39e-7	0.66
LDPC	1/3	128-QAM	<1e-8	<1e-8	0.51
		256-QAM	1.99e-8	<1e-8	0.59
	1/2	32-QAM	<1e-8	<1e-8	0.55
		64-QAM	<1e-8	<1e-8	0.66
	2/3	16-QAM	<1e-8	<1e-8	0.59
		32-QAM	<1e-8	<1e-8	0.73
	4/5	16-QAM	<1e-8	<1e-8	0.70

Table 4.2: Code and modulation combinations for $N_t = 62,\ N_r = 248$ achieving minimum throughput of 500 Gbps

Code Type	Rate	Modulation	BER, 2-Ray	BER, TDL	Throughput (Tbps)
Convolutional	1/3	32-QAM	4.21e-6	3.73e-6	0.52
	1/2	16-QAM	8.17e-7	1.04e-6	0.62
Turbo	1/3	32-QAM	<1e-8	<1e-8	0.52
		64-QAM	1.97e-8	3.97-8	0.62
	1/2	16-QAM	1.38e-7	1.57e-7	0.62
		32-QAM	2.60e-6	2.25e-6	0.78
LDPC	1/3	32-QAM	<1e-8	<1e-8	0.52
		64-QAM	<1e-8	<1e-8	0.62
	1/2	16-QAM	<1e-8	<1e-8	0.62
		32-QAM	<1e-8	<1e-8	0.78
	2/3	16-QAM	<1e-8	<1e-8	0.83

Table 4.3: Code and modulation combinations for $N_t = 80,\ N_r = 160$ achieving minimum throughput of 500 Gbps

Code Type	Rate	Modulation	BER, 2-Ray	BER, TDL	Throughput (Tbps)
Turbo	1/3	16-QAM	1.91e-8	<1e-8	0.53
		32-QAM	1.09e-6	7.44e-7	0.67
	1/2	8-PSK	1.14e-7	1.14e-7	0.60
LDPC	1/3	16-QAM	<1e-8	<1e-8	0.53
		32-QAM	<1e-8	<1e-8	0.67
	1/2	16-QAM	<1e-8	<1e-8	0.80
	2/3	4-QAM	<1e-8	<1e-8	0.53
	4/5	4-QAM	<1e-8	<1e-8	0.64

Table 4.4: Significant differences between two-ray and TDL multipath results

Antenna Configuration	**Code**	**Modulation**	**Two-Ray BER**	**TDL BER**
$N_t = 44,\ N_r = 352$	Rate ⅓ Conv.	32-QAM	2.97e-8	<1e-8
	Rate ⅔ Conv.	8-PSK	1.98e-8	9.91e-8
	Rate ½ Turbo	32-QAM	9.91e-8	5.94e-8
$N_t = 62,\ N_r = 248$	Rate ⅓ Turbo	64-QAM	1.97e-8	3.97e-8
	Rate ⅓ LDPC	128-QAM	4.33e-5	2.99e-8

V. Conclusions and Recommendations

THE overall goal of this inquiry was not quite successfully achieved. This chapter summarizes results and makes recommendations from them, should a system with similar goals be desired or designed. Trade-offs are discussed for systems engineers and program managers to consider should the previous condition be established. Finally, the limitations of the system are re-emphasized and potential areas for future work are identified.

5.1 Recommendations

Results from this study and previous work in [2] indicate that the number of antennas in the system should be increased over the [44:352], [62:248], or [80:160] transmitter to receiver configurations investigated. This could be accomplished at either, or both, the transmitter or receiver arrays as gains resulting from this increase should enable higher throughput up to a point. The best antenna configuration overall from this study was the [62:248] arrangement, which was able to achieve 0.827 Tbps of throughput at a BER of less than 1×10^{-8} when a rate ⅔ LDPC code and 16-QAM were utilized. The worst configuration, based solely on the best throughput obtained, was the [44:352] configuration.

The best case examined was the DVB-S2 rate ⅔ LDPC code applied to the [62:248] array with 16-QAM modulation, because it most nearly attained the minimum throughput requirement and exhibited a low BER. Both R-S and convolutional codes should be abandoned in favor of either turbo codes or LDPC codes, based on the results in Chapter 4. Turbo codes should not be used unless the [62:248] configuration, or larger variant of the 1:4 arrangement, is employed. However, as the turbo codes used in this study were not chosen for being the most efficient or powerful, it is indeterminate as to whether turbo codes or LDPC codes perform better in this context as an absolute.

5.2 Trade-Off Considerations

When either the [62:248] configuration or the [44:352] configuration are employed, there are at least two options that improve on the system over previous work in [2]. Systems engineers will find that the option of employing the [44:352] antenna configuration and a rate ⅘ LDPC code with 16-QAM, or a rate ⅔ LDPC code with 32-QAM both show absolute improvement over the previous system best case, with the latter being the higher throughput option. Finally, the [62:248] configuration may be utilized with one of three combinations which all outperform these previous two options: In order of increasing overall performance, a rate ½ turbo code with 32-QAM modulator, a rate ½ LDPC code with 32-QAM modulator, or a rate ⅔ LDPC code with 16-QAM modulation may be selected. If the [80:160] array is to be used, the only viable option (while still showing absolute improvement) is to use the rate ½ LDPC code with 16-QAM. This final arrangement performs better than all others except for the rate ⅔ LDPC code with 16-QAM applied to the [62:248] configuration.

5.3 Relationship to Previous Research

Previous research addressed and constrained a very open-ended problem, eliminated dead-end inquiries, and identified some key areas for further investigation [2]; this study improved upon high-throughput options recommended therein. The most significant aspect of the results presented here is that they reflect an improvement over the previous research in terms of both throughput and BER. The previous model was able to achieve either the throughput criterion or the BER criterion, but it was unable to achieve both even under the best conditions. The error control codes used in this study enabled the best case throughput to improve by 18.6%, but indicated that coding itself is not the limiting factor. This last point is evident in that no 1 Tbps options were identified despite very powerful FEC codes.

5.4 Future Research

Airships exist today which are capable of supporting hardware which carries out the communication system functions presented in this document. Components also exist to construct a system as it was modeled here, but despite the real demand for a system like this one, there are many other factors which must be examined before it would be wise to press ahead with construction. For example, all experiments assumed that the airship was a static object with respect to the receiver's frame of reference. Winds aloft are known to be as high as 70 knots, but the effects of antenna alignment and positioning with respect to the ground receiver array were omitted under this assumption. This and other factors should remove any expectations of realizing the ideal results presented without further research.

Link budget analysis should be performed. While one of the key strengths of the error correction codes examined in this study is that they perform well at low SNR, the higher order modulation schemes employed are highly sensitive to noise affects. The assumption that a SNR of 4.77 dB can be achieved at the input to the demodulator should be examined.

Before changes are made to antenna, FEC coding, or modulation configurations, more work should be devoted to characterizing capacity limitations, ideally using actual channel measurements. Additionally, only ICD was used to estimate received MIMO symbols. Future work should attempt to estimate received symbols using a ML detector so that additional symbol detection gain may be obtained.

The most effective forward error correction schemes incorporate known or anticipated channel effects into their design, but this was not accomplished in this increment. The Bell Labs Layered Space-Time (BLAST) algorithm should be examined, with particular focus on Turbo-BLAST. This algorithm attempts to account for Rayleigh-faded MIMO channels, however appears to focus on increasing spatial diversity via redundancy [26],[27]. Investigation into the applicability of this, and other tailored coding methods in the context of this problem is warranted.

Appendix A: Viterbi Algorithm Example

The following sequence of figures shows how the Viterbi algorithm decodes convolutional codes from the trellis standpoint. Once the decoder has progressed through the trellis to the point that multiple paths are entering the same trellis node, redundant path elimination begins. Paths with greater cumulative Hamming weights (in parentheses next to the output for each branch) from the received sequence (denoted under the time sample value) are removed from consideration. The Viterbi algorithm only maintains a maximum number of paths equal to the number of decoder states [5, p. 405]. In Fig. A.6 this rule appears to be broken, but it is at this point that all paths but the most likely are eliminated. The trellis utilized is the previously discussed ½, $[75]_8$ NSC code.

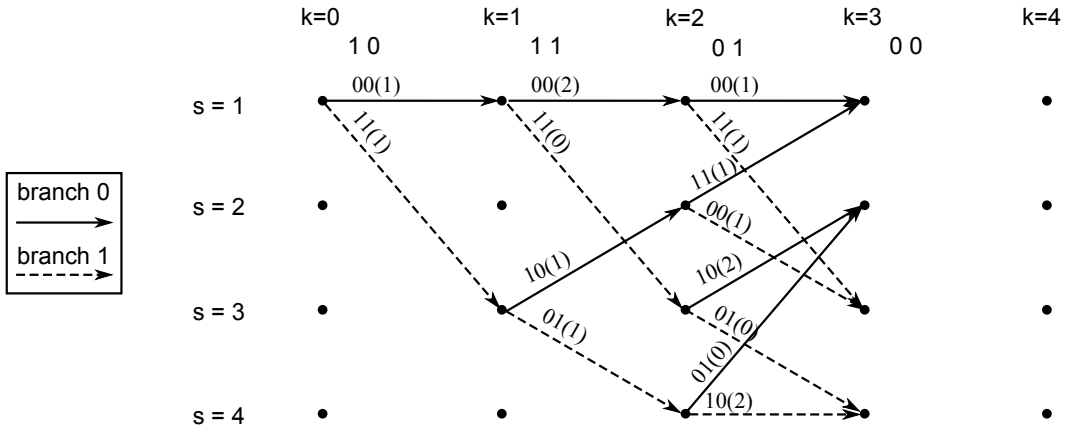


Figure A.1: Viterbi algorithm: Advancing to the point of having multiple inbound paths for at least one state

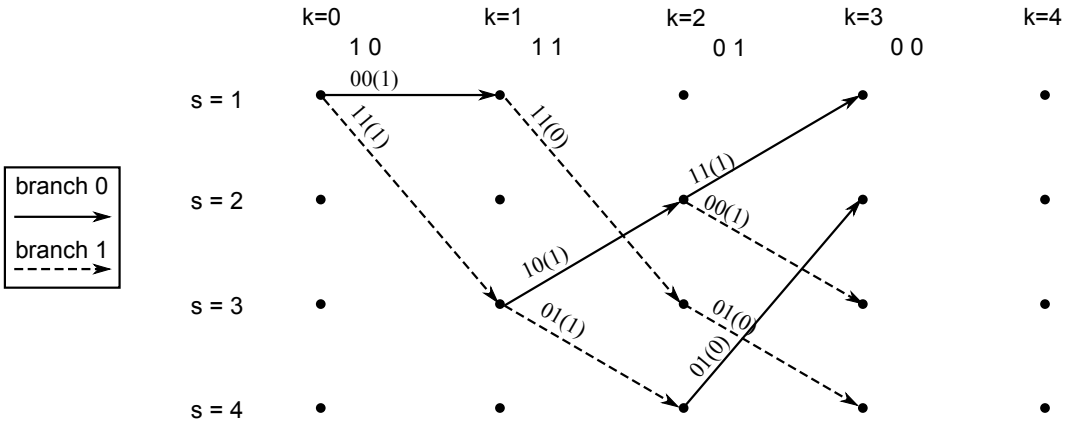


Figure A.2: Viterbi algorithm: Path reduction based on Hamming distance

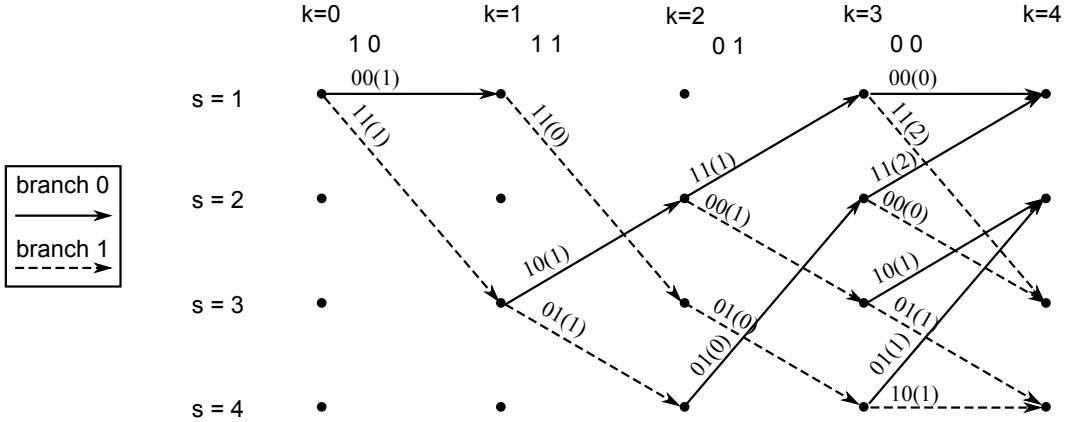


Figure A.3: Viterbi algorithm: Advancing one more time step

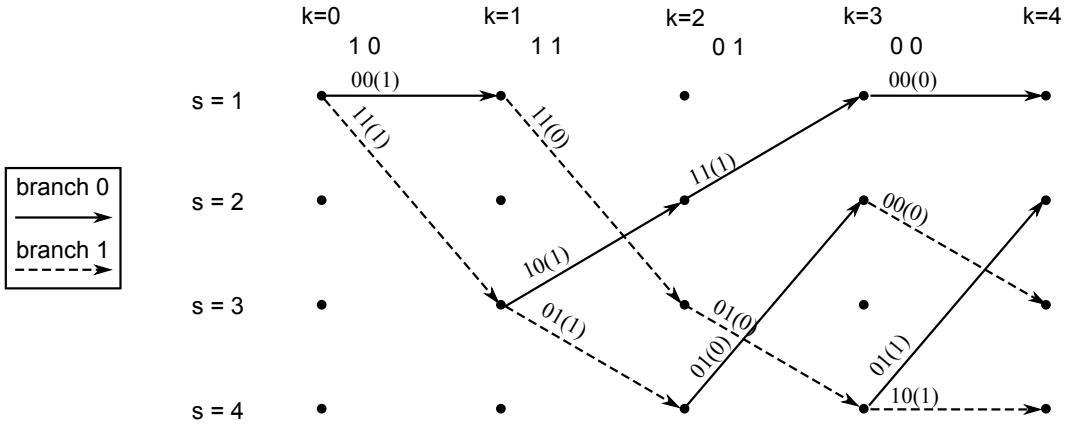


Figure A.4: Viterbi algorithm: Removing unlikely branches

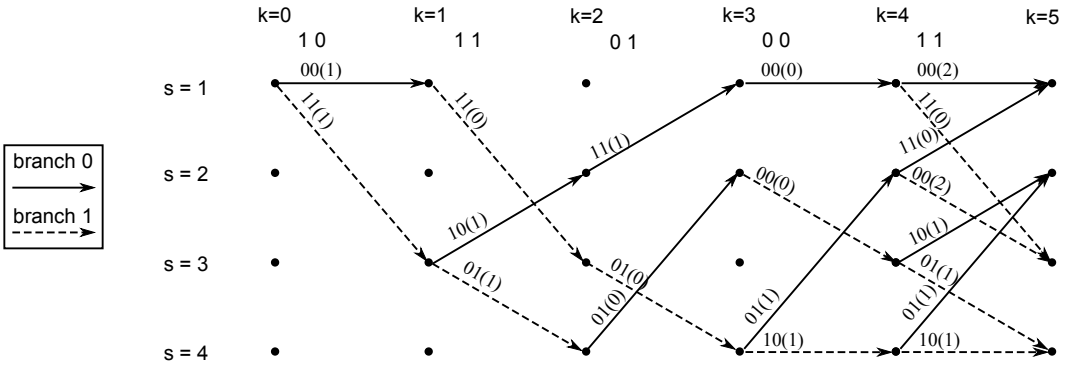


Figure A.5: Viterbi algorithm: Advancing another time step

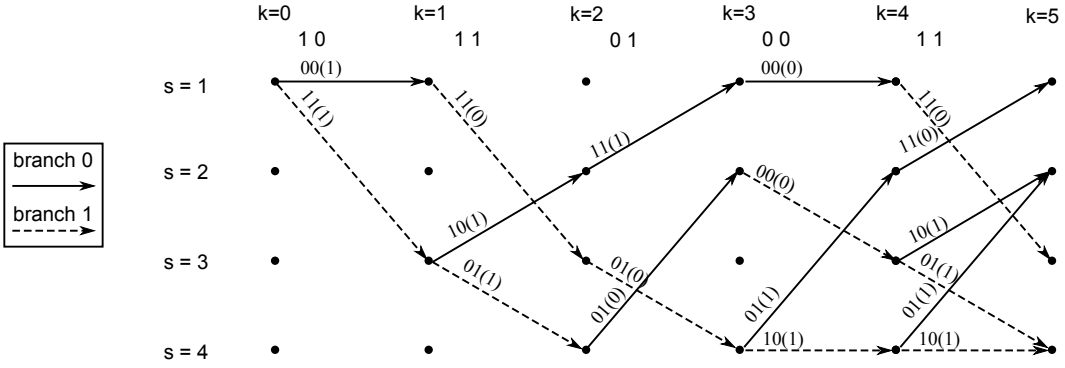


Figure A.6: Viterbi algorithm: At first glance, the algorithm may seem to have failed

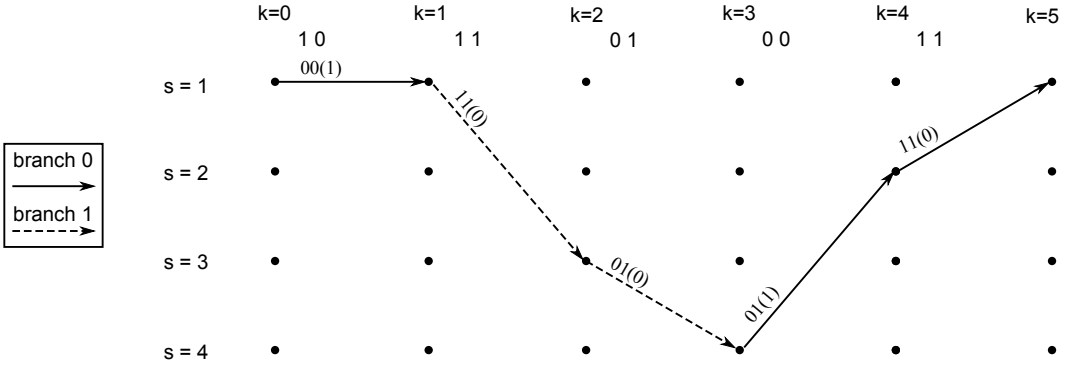


Figure A.7: Viterbi algorithm: Final path selected

Appendix B: Implentation Details

Tables presented in this section represent the bits added to the encoded sequence of Section 3.3 when the sequence length is not divisible by $\log_2(M)$ where M is the constellation size. Figure B.1 shows the upper left quadrant of a Gray-coded 128-QAM constellation. The point the arrow is aimed at represents the constellation point selected when the input block has a remainder of one. Points were chosen in this fashion where possible to avoid collisions with other symbols when noise was introduced into the signal content.

Table B.1: Modulator input block fitting for 8-PSK

Remainder after division	**Additional bits required**	**Sequence appended**
2	1	1 bit at random
1	2	2 bits at random

Table B.2: Modulator input block fitting for 32-QAM

Remainder after division	**Additional bits required**	**Sequence appended**
3	2	0 0
2	3	0 0 0
1	4	0 0 0 0

B.1 Data Block Size for Individual Experiments

Table B.3: Modulator input block fitting for 64-QAM

Remainder after division	**Additional bits required**	**Sequence appended**
4	2	0 0
2	4	0 0 0 0

Table B.4: Modulator input block fitting for 128-QAM

Remainder after division	**Additional bits required**	**Sequence appended**
6	1	1
4	3	0 0 1
2	5	0 0 0 0 1
1	6	0 0 0 0 0 1

Table B.5: LDPC code input block size (bits)

Code Rate	$\mathbf{N_t = 44,\ N_r = 352}$	$\mathbf{N_t = 62,\ N_t = 248}$	$\mathbf{N_t = 80,\ N_r = 160}$
1/3	100,742,400	100,440,000	100,224,000
1/2	101,217,600	100,440,000	101,088,000
2/3	100,742,400	101,779,200	100,224,000
4/5	100,362,240	102,850,560	103,680,000

Table B.6: Turbo code input block size (bits)

Code Rate	$\mathbf{N_t = 44,\ N_r = 352}$	$\mathbf{N_t = 62,\ N_t = 248}$	$\mathbf{N_t = 80,\ N_r = 160}$
1/3	100,925,440	101,580,800	104,857,600
1/2	100,925,440	101,580,800	104,857,600

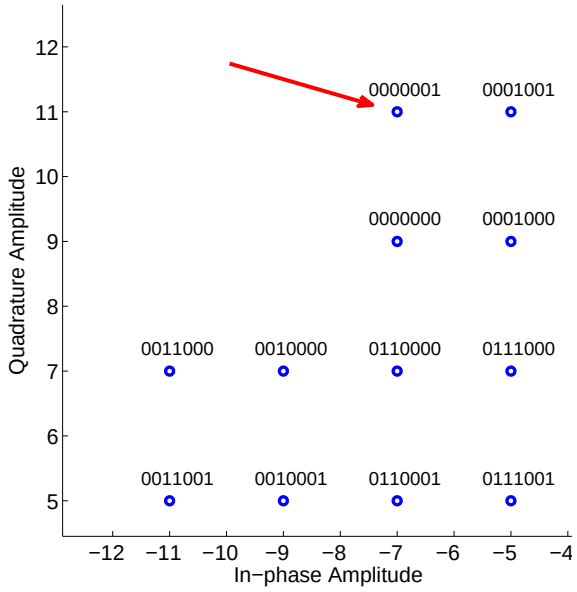


Figure B.1: Example constellation point selection, $M = 7$, final encoded bit = 0

Table B.7: Convolutional code input block size (bits)

Code Rate	$\mathbf{N_t = 44,\ N_r = 352}$	$\mathbf{N_t = 62,\ N_t = 248}$	$\mathbf{N_t = 80,\ N_r = 160}$
1/3	100,900,800	101,556,000	104,832,000
1/2	100,900,800	101,556,000	104,832,000
2/3	100,900,800	101,556,000	104,832,000
4/5	100,900,800	101,556,000	104,832,000

Appendix C: Complete Results

C.1 Throughput Calculations

Table C.1: Throughput (Tbps) results, $N_t = 44,\ N_r = 352$

Code Rate	M=2	M=4	M=8	M=16	M=32	M=64	M=128	M=256
1/3	0.0733	0.1467	0.2200	0.2933	0.3667	0.4400	0.5133	0.5867
1/2	0.1100	0.2200	0.3300	0.4400	0.5500	0.6600	0.7700	0.8800
2/3	0.1467	0.2933	0.4400	0.5867	0.7333	0.8800	1.0267	1.1733
4/5	0.1760	0.3520	0.5280	0.7040	0.8800	1.0560	1.2320	1.4080

Table C.2: Throughput (Tbps) results, $N_t = 62,\ N_r = 248$

Code Rate	M=2	M=4	M=8	M=16	M=32	M=64	M=128	M=256
1/3	0.1033	0.2067	0.3100	0.4133	0.5167	0.6200	0.7233	0.8267
1/2	0.1550	0.3100	0.4650	0.6200	0.7750	0.9300	1.0850	1.2400
2/3	0.2067	0.4133	0.6200	0.8267	1.0333	1.2400	1.4467	1.6533
4/5	0.2480	0.4960	0.7440	0.9920	1.2400	1.4880	1.7360	1.9840

Table C.3: Throughput (Tbps) results, $N_t = 80,\ N_r = 160$

Code Rate	M=2	M=4	M=8	M=16	M=32	M=64	M=128	M=256
1/3	0.1333	0.2667	0.4000	0.5333	0.6667	0.8000	0.9333	1.0667
1/2	0.2000	0.4000	0.6000	0.8000	1.0000	1.2000	1.4000	1.6000
2/3	0.2667	0.5333	0.8000	1.0667	1.3333	1.6000	1.8667	2.1333
4/5	0.3200	0.6400	0.9600	1.2800	1.6000	1.9200	2.2400	2.5600

C.2 Two-Ray Ground Reflection Multipath Model BER Results

The following tables present the results from all tests run, regardless of usefulness, for the two-ray multipath model. In each table, $< 1 \times 10^{-8}$ indicates that no errors were encountered in the simulation, and that the BER is smaller than can be accurately resolved by the simulation. Places where "NC" appears in the table are those or which no data was collected for the configuration in question, ususally due to the fact that the previous level of M experienced a BER high enough to rule it, and larger constellation sizes, out of the question for actual fielding.

Table C.4: Convolutional code BER results, two-ray multipath, $N_t = 44,\ N_r = 352$

R	**M=2**	**M=4**	**M=8**	**M=16**	**M=32**	**M=64**	**M=128**	**M=256**
1/3	<1e-8	<1e-8	<1e-8	<1e-8	2.97e-8	1.98e-8	2.63e-4	2.59e-4
1/2	<1e-8	<1e-8	<1e-8	<1e-8	1.02e-5	8.79e-5	NC	NC
2/3	<1e-8	<1e-8	1.98e-8	2.18e-5	5.99e-3	NC	NC	NC
4/5	<1e-8	<1e-8	6.66e-6	5.40e-4	NC	NC	NC	NC

Table C.5: Turbo code BER results, two-ray multipath, $N_t = 44,\ N_r = 352$

R	**M=2**	**M=4**	**M=8**	**M=16**	**M=32**	**M=64**	**M=128**
1/3	<1e-8	<1e-8	<1e-8	<1e-8	<1e-8	<1e-8	3.38e-1
1/2	<1e-8	<1e-8	<1e-8	<1e-8	9.91e-8	1.38e-7	2.18e-1

Table C.6: LDPC code BER results, two-ray multipath, $N_t = 44,\ N_r = 352$

R	**M=2**	**M=4**	**M=8**	**M=16**	**M=32**	**M=64**	**M=128**	**M=256**
1/3	<1e-8	<1e-8	<1e-8	<1e-8	<1e-8	<1e-8	<1e-8	1.99e-8
1/2	<1e-8	<1e-8	1.19e-3	<1e-8	<1e-8	<1e-8	5.59e-2	1.79e-1
2/3	<1e-8	<1e-8	2.43e-3	<1e-8	<1e-8	1.29e-3	1.38e-1	1.79e-1
4/5	<1e-8	<1e-8	2.53e-3	<1e-8	3.95e-4	8.48e-2	1.33e01	1.69e-1

Table C.7: Convolutional code BER results, two-ray multipath, $N_t = 62,\ N_r = 248$

R	**M=2**	**M=4**	**M=8**	**M=16**	**M=32**	**M=64**
1/3	<1e-8	<1e-8	<1e-8	<1e-8	4.21e-6	9.84e-5
1/2	<1e-8	<1e-8	<1e-8	8.17e-7	2.88e-3	2.90e-2
2/3	<1e-8	<1e-8	3.91e-5	5.15e-3	NC	NC
4/5	<1e-8	2.66e-7	2.99e-3	5.67e-2	NC	NC

Table C.8: Turbo code BER results, two-ray multipath, $N_t = 62,\ N_r = 248$

R	**M=2**	**M=4**	**M=8**	**M=16**	**M=32**	**M=64**	**M=128**
1/3	<1e-8	<1e-8	<1e-8	<1e-8	<1e-8	1.97e-8	3.50e-1
1/2	<1e-8	<1e-8	<1e-8	1.38e-7	2.60e-6	1.54e-1	NC

Table C.9: LDPC code BER results, two-ray multipath, $N_t = 62,\ N_r = 248$

R	**M=2**	**M=4**	**M=8**	**M=16**	**M=32**	**M=64**	**M=128**	**M=256**
1/3	<1e-8	<1e-8	<1e-8	<1e-8	<1e-8	<1e-8	4.33e-5	2.36e-1
1/2	<1e-8	<1e-8	1.47e-2	<1e-8	<1e-8	1.46e-1	2.15e-1	2.45e-1
2/3	<1e-8	<1e-8	2.02e-2	<1e-8	9.28e-2	1.53e-1	2.05e-1	2.36e-1
4/5	<1e-8	<1e-8	2.06e-2	2.00e-2	1.07e-1	1.45e-1	1.94e-1	2.28e-1

Table C.10: Convolutional code BER results, two-ray multipath, $N_t = 80,\ N_r = 160$

R	**M=2**	**M=4**	**M=8**	**M=16**	**M=32**	**M=64**
1/3	<1e-8	<1e-8	1.24e-7	2.80e-5	3.77e-3	3.45e-2
1/2	<1e-8	<1e-8	3.70e-5	5.77e-3	1.68e-1	3.35e-1
2/3	<1e-8	2.54e-5	1.07e-2	1.19e-1	NC	NC
4/5	9.54e-8	2.80e-4	1.21e-1	3.19e-1	NC	NC

Table C.11: Turbo code BER results, two-ray multipath, $N_t = 80,\ N_r = 160$

R	**M=2**	**M=4**	**M=8**	**M=16**	**M=32**	**M=64**
1/3	<1e-8	<1e-8	<1e-8	1.91e-8	1.09e-6	3.31e-3
1/2	<1e-8	<1e-8	1.14e-7	1.18e-5	2.12e-1	NC

Table C.12: LDPC code BER results, two-ray multipath, $N_t = 80,\ N_r = 160$

R	**M=2**	**M=4**	**M=8**	**M=16**	**M=32**	**M=64**	**M=128**	**M=256**
1/3	<1e-8	<1e-8	<1e-8	<1e-8	<1e-8	2.38e-1	2.92e-1	3.05e-1
1/2	<1e-8	<1e-8	5.88e-2	<1e-8	1.92e-1	2.38e-1	2.81e-1	3.00e-1
2/3	<1e-8	<1e-8	6.55e-2	1.09e-1	1.86e-1	2.23e-1	2.69e-1	2.91e-1
4/5	<1e-8	<1e-8	6.58e-2	1.16e-1	1.78e-1	2.13e-1	2.62e-1	2.86e-1

Table C.13: Convolutional code BER results, TDL multipath, $N_t = 44,\ N_r = 352$

R	**M=2**	**M=4**	**M=8**	**M=16**	**M=32**	**M=64**
1/3	<1e-8	<1e-8	<1e-8	<1e-8	<1e-8	1.98e-8
1/2	<1e-8	<1e-8	<1e-8	<1e-8	1.06e-5	NC
2/3	<1e-8	<1e-8	9.91e-8	2.16e-5	6.03e-3	NC
4/5	<1e-8	<1e-8	6.19e-6	5.44e-4	4.68	NC

Table C.14: Turbo code BER results, TDL multipath, $N_t = 44,\ N_r = 352$

R	**M=2**	**M=4**	**M=8**	**M=16**	**M=32**	**M=64**	**M=128**
1/3	<1e-8	<1e-8	<1e-8	<1e-8	<1e-8	<1e-8	3.32e-1
1/2	<1e-8	<1e-8	<1e-8	<1e-8	5.94e-8	1.38e-7	8.53e-5

Table C.15: LDPC code BER results, TDL multipath, $N_t = 44,\ N_r = 352$

R	**M=2**	**M=4**	**M=8**	**M=16**	**M=32**	**M=64**	**M=128**	**M=256**
1/3	<1e-8	<1e-8	<1e-8	<1e-8	<1e-8	<1e-8	<1e-8	<1e-8
1/2	<1e-8	<1e-8	1.20e-3	<1e-8	<1e-8	<1e-8	5.44e-2	1.79e-1
2/3	<1e-8	<1e-8	2.43e-3	<1e-8	<1e-8	1.20e-3	1.38e-1	1.79e-1
4/5	<1e-8	<1e-8	2.54e-3	<1e-8	3.66e-4	8.48e-2	1.33e01	1.69e-1

Table C.16: Convolutional code BER results, TDL multipath, $N_t = 62,\ N_r = 248$

R	**M=2**	**M=4**	**M=8**	**M=16**	**M=32**	**M=64**
1/3	<1e-8	<1e-8	<1e-8	9.84-e9	3.73e-6	1.02e-4
1/2	<1e-8	<1e-8	<1e-8	1.04e-6	2.84e-3	NC
2/3	<1e-8	1.97-8	4.05e-5	5.15e-3	8.79e-2	NC
4/5	<1e-8	4.92e-8	2.99e-3	5.66e-2	2.88e-1	NC

Table C.17: Turbo code BER results, TDL multipath, $N_t = 62,\ N_r = 248$

R	**M=2**	**M=4**	**M=8**	**M=16**	**M=32**	**M=64**	**M=128**
1/3	<1e-8	<1e-8	<1e-8	<1e-8	<1e-8	3.97e-8	3.44e-1
1/2	<1e-8	<1e-8	<1e-8	1.57e-7	2.24e-6	1.54e-1	NC

Table C.18: LDPC code BER results, two-ray multipath, $N_t = 62,\ N_r = 248$

R	**M=2**	**M=4**	**M=8**	**M=16**	**M=32**	**M=64**	**M=128**	**M=256**
1/3	<1e-8	<1e-8	<1e-8	<1e-8	<1e-8	<1e-8	2.99e-8	2.36e-1
1/2	<1e-8	<1e-8	1.47e-2	<1e-8	<1e-8	1.46e-1	2.15e-1	2.45e-1
2/3	<1e-8	<1e-8	2.02e-2	<1e-8	9.26e-2	1.53e-1	2.05e-1	2.36e-1
4/5	<1e-8	<1e-8	2.06e-2	2.00e-2	1.07e-1	1.45e-1	1.94e-1	2.28e-1

Table C.19: Convolutional code BER results, TDL multipath, $N_t = 80,\ N_r = 160$

R	**M=2**	**M=4**	**M=8**	**M=16**	**M=32**
1/3	<1e-8	<1e-8	9.54e-8	2.33e-5	NC
1/2	<1e-8	<1e-8	3.38e-5	5.86e-3	1.68e-1
2/3	<1e-8	2.59e-5	1.07e-2	1.19e-1	NC
4/5	1.34e-7	2.84e-4	1.21e-1	5.64e-2	4.3e-1

Table C.20: Turbo code BER results, TDL multipath, $N_t = 80,\ N_r = 160$

R	**M=2**	**M=4**	**M=8**	**M=16**	**M=32**	**M=64**
1/3	<1e-8	<1e-8	<1e-8	<1e-8	7.44e-7	3.17e-3
1/2	<1e-8	<1e-8	1.14e-7	1.23e-5	2.12e-1	2.51e-1

Table C.21: LDPC code BER results, TDL multipath, $N_t = 80,\ N_r = 160$

R	**M=2**	**M=4**	**M=8**	**M=16**	**M=32**	**M=64**	**M=128**	**M=256**
1/3	<1e-8	<1e-8	<1e-8	<1e-8	<1e-8	2.39e-1	2.92e-1	3.04e-1
1/2	<1e-8	<1e-8	5.88e-2	<1e-8	1.92e-1	2.38e-1	2.81e-1	2.99e-1
2/3	<1e-8	<1e-8	6.54e-2	1.09e-1	1.86e-1	2.23e-1	2.69e-1	2.91e-1
4/5	<1e-8	<1e-8	6.57e-2	1.16e-1	1.77e-1	2.13e-1	2.62e-1	2.86e-1

Bibliography

[1] Stew Magnuson, "Military 'Swimming in Sensors and Drowning in Data'," *National Defense Industrial Association Business and Technology Magazine*, vol. XCV, no. 686, January 2010.

[2] Adam Brueggen, "Trade-offs in a 1 Tbps Multiple-Input and Multiple-Output (MIMO) Communication System Between an Airship and Ground Receive Antennas," M.S. thesis, Air Force Institute of Technology, 2012.

[3] Office of Spectrum Manangement National Telecomunications & Information Administration, "Federal Spectrum Use Summary: 30 MHz – 3000 GHz," Tech. Rep., 2010.

[4] Dennis Roddy, *Satellite Communications*, McGraw-Hill, New York, 4th edition, 2006.

[5] Bernard Sklar, *Digital Communications: Fundamentals and Applications*, Prentice Hall PTR, Upper Saddle River, NJ, 2nd edition, 2001.

[6] John G. Proakis and Masoud Salehi, *Digital Communications*, McGraw Hill Higher Education, Boston, MA, 5th edition, 2008.

[7] Claude Berrou and Alaine Glavieux, "Near Optimum Error Correcting Coding and Decoding: Turbo-Codes," *IEEE Transactions on Communications*, vol. 44, no. 10, pp. 1261–1271, October 1996.

[8] Steven M. Kay, *Fundamentals of Statistical Signal Processing: Estimation Theory*, Prentice Hall PTR, Upper Saddle River, NJ, 1993.

[9] Emmanuel C. Ifeachor and Barrie W. Jervis, *Digital Signal Processing*, Pearson Education Ltd., New York, NY, 2nd edition, 2002.

[10] Andrea Goldsmith, *Wireless Communications*, Cambridge University Press, Cambridge, 2005.

[11] Christian Schlegel, *Trellis Coding*, The Institute of Electrical and Electronics Engineers, Inc., 1997.

[12] Alister Burr, "Turbo-Codes: The Ultimate Error Control Codes?," *Electronics Communication Engineering Journal*, vol. 13, no. 4, pp. 155 –165, aug 2001.

[13] Lajos Hanzo, Osamah Alamri, Mohammed El-Hajjar, and Nan Wu, *Near-Capacity Multi-Functional MIMO Systems: Sphere-Packing, Iterative Detection and Cooperation*, John Wiley & Sons Ltd, Chichester, UK, 2010.

[14] Steven M. Kay, *Fundamentals of Statistical Signal Processing: Detection Theory*, Prentice Hall PTR, Upper Saddle River, NJ, 1998.

[15] Stefania Sesia, Issam Toufik, and Matthew Baker, *LTE-The UMTS Long Term Evolution: From Theory to Practice*, John Wiley & Sons Ltd, Chichester, UK, 2nd edition, 2011.

[16] Ferrel Stremler, *Introduction to Communication Systems*, Addison Wesley, Reading, MA, 3rd edition, 1990.

[17] Wiley J. Larson and James R. Wertz, *Space Mission Analysis and Design*, Microcosm Press, Hawthorn, CA, 3rd edition, 1999.

[18] Theodore S. Rappaport, *Wireless Communications*, Prentice Hall PTR, Upper Saddle River, NJ, 2nd edition, 2002.

[19] Gilbert Strang, *Linear Algebra and Its Applications*, Thompson Higher Education, Belmont, CA, 3rd edition, 2006.

[20] Gregory D. Durgin, *Space-Time Wireless Channels*, Prentice Hall PTR, Upper Saddle River, NJ, 2003.

[21] David G. Daut, James W. Modestino, and Lee D. Wismer, "New Short Constraint Length Convolutional Code Constructions for Selected Rational Rates," *IEEE Transactions on Information Theory*, vol. 28, no. 5, pp. 794 – 800, September 1982.

[22] "IEEE Standard for Floating-Point Arithmetic," *IEEE Std 754-2008*, pp. 1 –58, 29 2008.

[23] Inc. Mathworks, "Communications System Toobox," 2011, MATLAB® (R2011a).

[24] Andrew J. Viterbi, "An Intuitive Justification and a Simplified Implementation of the MAP Decoder for Convolutional Codes," *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 2, pp. 260–264, February 1998.

[25] M. Morris Mano and Charles R. Kime, *Logic and Computer Design Fundamentals*, Pearson Education, Inc., Upper Saddle River, NJ, 3rd edition, 2004.

[26] Mathini Sellathurai and Simon Haykin, "TURBO-BLAST for Wireless Communications: Theory and Experiments," *IEEE Transactions on Signal Processing*, vol. 50, no. 10, pp. 2538–2546, October 2002.

[27] Mathini Sellathurai and Simon Haykin, "Turbo-BLAST: Performance Evaluation in Correlated Rayleigh-Fading Environment," *IEEE Journal on Selected Areas in Communications*, vol. 21, no. 3, pp. 340–349, April 2003.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704–0188

The public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704–0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202–4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number. **PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.**

1. REPORT DATE *(DD–MM–YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From — To)*
21–03–2013	Master's Thesis	Oct 2011–Mar 2013

4. TITLE AND SUBTITLE

Improving Bandwidth Utilization in a 1 Tbps Airborne MIMO Communications Downlink

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Hill, Jonathan D., Captain, USAF

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/EN)
2950 Hobson Way
WPAFB, OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT-ENG-13-M-25

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Intentionally Left Blank

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

DISTRIBUTION STATEMENT A.
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

13. SUPPLEMENTARY NOTES

This work is declared a work of the U.S. Government and is not subject to copyright protection in the United States.

14. ABSTRACT

FEC techniques are compared for different MIMO configurations of a high altitude, extremely wide bandwidth radio frequency downlink. Monte Carlo simulations are completed in MATLAB® with the aim of isolating the impacts of turbo codes and LDPC codes on system throughput and error performance. The system is modeled as a transmit-only static array at an altitude of 60,000 feet, with no interferers in the channel. Transmissions are received by a static receiver array. Simulations attempt to determine what modulation types should be considered for practical implementation, and what FEC codes enable these modulation schemes. The antenna configurations used in this study are [44:352], [62:248], and [80:160] transmitters to receivers. Effects from waveform generation, mixing, down-conversion, and amplification are not considered.

Criteria of interest were BER and throughput, with the maximum allowable value of the former set at 1×10^{-5}, and the latter set at a 1 terabits per second (Tbps) transfer rate for a successful configuration. Results show that the best performing system configuration was unable to meet both criteria, but was capable of improving over Brueggen's 2012 research, which used Reed-Solomon codes and a MIMO configuration of [80:160], by 18.6%. The best-case configuration produced a throughput rate of 0.83 Tbps at a BER of less than 1×10^{-8}, by implementing a rate ⅔ LDPC code with QAM constellation of 16 symbols.

15. SUBJECT TERMS

MIMO, Turbo Code, Low Density Parity Check Code, Forward Error Correction, Airship

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Dr. Richard K. Martin (ENG)
U	U	U	UU	105	19b. TELEPHONE NUMBER *(include area code)* (937) 255-3636 x4625 richard.martin@afit.edu