Multilevel Polar Codes for Grassmannian Signalling

by

Philip Reinhard Balogun

A thesis submitted to the Faculty of Graduate and Postdoctoral
Affairs in partial fulfillment of the requirements
for the degree of

Master of Applied Science

in

Electrical and Computer Engineering

Carleton University
Ottawa, Ontario

© 2016

Philip Reinhard Balogun

Abstract

Polar codes, ever since their introduction, have been shown to be very effective for various wireless communication channels. This, together with their relatively low implementation complexity, has made polar codes an attractive coding scheme for wireless communications. On the other hand, within the realm of non-coherent wireless MIMO communication, Grassmannian signalling has been shown to approach the ergodic capacity of frequency-flat block fading channels at high SNR. In this paper, we combine these two components together using multilevel polar coding to create a system that produces the best known performance for a coded non-coherent channel. This combination requires that the signal constellation be set-partition labelled, so, a novel set partitioning algorithm, which works for regular and irregular multidimensional constellations, such as the Grassmannian constellation, is proposed. Finally, we develop a methodology for designing polar codes for a noisy channel and this is used to further improve the performance of our system. In simulation, we compare the error rate performance of our design with that of existing schemes and show that significant gains are possible over the previously known best technique, which is based on turbo codes. We then provide a complexity analysis of our receiver in comparison with other existing methods and show that it is able to provide these gains at a considerably lower complexity.

Acknowledgements

I would like to express my sincere gratitude to my supervisors for all the time and effort they invested in me throughout the duration of my research. To Prof. Ian Marsland, I deeply appreciate all the time you made out to meet with me on a consistent basis to discuss the research and to clarify many concepts to me. Thank you for providing such detailed reviews on my thesis – the final work would certainly not be what it is without you. To Prof. Halim Yanikomeroglu, thank you very much for all the encouragement you gave to me, financial and otherwise, and thank you for providing a comfortable working space. To Dr. Ramy Gohary, I appreciate all the time you spent explaining the Grassmannian concepts to me and all the resources you provided to improve my understanding. I thank you all for believing in me, I could not have asked for a better team of supervisors.

I would also like to thank my research colleagues for all the support, encouragement and advice they gave me throughout my research process.

I would like to thank my family and friends for the support and prayers that kept me going. And last but not least, I'd like to thank my lord and saviour, Jesus Christ, because without His favour in my life, none of this would be possible in the first place.

Contents

List of Tables	vi
List of Figures	vii
1 Introduction	1
2 Background	4
2.1 Polar Codes	4
2.1.1 Channel Polarization	4
2.1.2 Encoding	6
2.1.3 Decoding	7
2.2 Multilevel Codes	8
2.2.1 Transmitter	9
2.2.2 Receiver	10
2.3 Set-Partitioning Mapping	11
2.4 Grassmannian Signalling	12
3 System Design	16
3.1 System Set-up	16
3.1.1 Set Partitioning Algorithm	17
3.2 System Design Methodology	23

4	Simulation Results and Analysis	28
4.1	Performance of Designed System	28
4.2	Comparison With Other Existing Techniques	32
4.3	Results With a Training Based Scheme	33
4.4	Results With Coherent Golden Codes	36
4.5	Complexity Analysis	37
4.5.1	De-mapping Complexity Analysis	38
4.5.2	Bit LLR Calculation Complexity Analysis	40
4.5.3	Decoding Complexity Analysis	41
5	Conclusion	45
	References	47

List of Tables

4.1	De-mapping complexity comparison of Grassmannian constellation and Golden codes.	39
4.2	Example of de-mapping complexity comparison of Grassmannian constellation and Golden codes. $N_t = N_r = 2$, $T = 4$, $M = 4096$, and $N' = 512$	40
4.3	LLR Calculation, $b =$ number of de-mapping iterations, $m = \log_2 M$	41
4.4	Decoding complexity of turbo and multi-level polar codes. $n' = \log_2 N'$, $k =$ constraint length, $N =$ Frame length, $i =$ total number of decoder iterations, $d =$ number of decoders, $b =$ number of de-mapping iterations.	42
4.5	Decoding complexity example of turbo and multi-level polar codes. $n' = 9$, $k = 5$, $N = 6144$, $i = 32$, $d = 2$, $b = 4$	43
4.6	Comparison of the total number of receiver operations for Grassmannian and Golden codes. $n' = \log_2 N'$	43
4.7	Example of the total number of receiver operations for Grassmannian and Golden codes.	44

List of Figures

1.1	Typical coded modulation system.	2
2.1	Examples of channel polarization using a binary erasure channel with erasure probability = 0.5.	5
2.2	Polar code graph.	6
2.3	BICM block diagram.	8
2.4	Block diagram for multilevel codes.	9
2.5	Ungerboek's method of set partitioning a 16-QAM constellation.	12
2.6	MIMO channel.	12
3.1	Sample 16-QAM Constellation.	20
3.2	Distance table for partitioning a 16-QAM constellation.	21
3.3	Set partitioning a 16-QAM signal using our algorithm.	22
3.4	Bit error rate per bit of a 16-QAM constellation.	23
3.5	Bit error rate per bit of a 256 point Grassmannian constellation.	24
3.6	Effect of design SNR on the FER performance of our system. $N' = 512$, rate $R = 4/5$, $M = 4096$	26
4.1	Comparison of the coded vs uncoded schemes for a 4096 point Grassmannian constellation.	29
4.2	Polar codes with 4096 point Grassmannian constellations.	30

4.3	4096 point Grassmannian constellation with polar codes of different sub-code lengths with code rate $4/5$	31
4.4	Different codes running with 4096 point Grassmannian constellation with rate $R=4/5$. All BICM figures use quasi-Gray labelling for the constellation. Multilevel code uses set partitioned labelling. Un-optimized BICM codes are optimized for a BPSK AWGN channel only.	32
4.5	256 point Golden code with (a) set partitioning and (b) Gray labelling. . . .	35
4.6	Performance of Golden codes and Grassmannian constellations with the multilevel polar coding. Sub-code frame length = 256. Rate = $4/5$	36
4.7	Polar codes with 256 point coherent Golden codes. Rate = $1/2$. $N = 8192$. . .	37

Chapter 1

Introduction

Coherent communication, in which perfect channel state information (CSI) is available at the receiver, dominates current cellular technologies. To obtain this CSI, training based estimation methods using pilot or reference signals are used. For single-input single-output systems under slow fading conditions, this method has proved extremely effective as these training symbols may be transmitted quite sparingly and therefore do not consume a lot of resources. Future technologies requiring signal detection in multiple-input multiple-output (MIMO) and/or fast fading scenarios, however, would significantly increase the signalling overhead of such coherent systems. This is because the training symbols would need to be transmitted more often and with enough power for the system to correctly estimate the channel. As a result, non-coherent systems which do not require any channel estimation, are therefore attractive as alternatives in such scenarios. The challenge then becomes finding a low-complexity, rate-efficient method of enabling non-coherent communication over MIMO channels.

A possible candidate for non-coherent MIMO communication, specifically designed for block-fading channels, are Grassmannian constellations [1]. These multidimensional constellations work with the observation that the distortion caused by a fading channel does not change the subspace in which a transmitted signal resides, it only rotates and scales the bases

Fig. 1.1: Typical coded modulation system.

of the subspace [2]. These constellations exploit such characteristics and consider orthogonal subspaces to differentiate the transmitted symbols at the receiver [3]. It was shown in [2] that these constellations are able to approach the ergodic capacity at high signal-to-noise ratios (SNRs). To improve the error rate performance at lower SNRs it is advisable to use coded modulation with Grassmannian signalling. This involves combining error correcting codes with a signal constellation as shown in Fig. 1.1. These error correcting codes, as the name suggests, allows a receiver to detect and correct some errors incurred during transmission by exploiting carefully inserted redundancies in the transmitted data. The field of channel coding is one that has existed since the 1950s when Richard Hamming pioneered the first error-correcting code, the hamming (7,4) code [4]. Ever since then, better and better codes have been created such as the near capacity approaching low density parity check (LDPC) and turbo codes. However, the decoding of these codes can be computationally intensive.

In the literature, Grassmannian constellations have been combined in a bit interleaved coded modulation (BICM) fashion with turbo codes [5]. It was shown that at a high data rate, the scheme was able to outperform training based methods. Further work in this area showed that by improving the labelling for the Grassmannian constellations, the system was able to outperform even training based Golden codes [6].

Polar codes, a recent invention by Erdal Arikan, are a class of error-correcting codes with the proven ability to achieve the capacity of binary input, memoryless output symmetric channels [7]. In addition, they require relatively low encoding and decoding complexity. Although polar codes have been used with BICM, generally speaking, they perform worse than other codes such as LDPC and turbo codes [8]. However, it has been observed [9] that polar codes perform better when used with multilevel coding (MLC) [10] than with BICM

[11].

In this thesis we aim to improve the error rate performance of Grassmannian signalling over non-coherent Rayleigh block fading MIMO channels by using polar codes in a multilevel coding scheme. MLC requires that the constellation uses set partition (SP) labelling to produce the best results [12]. However, the current set partitioning algorithms can only be used for regular constellations, and so are unsuitable for Grassmannian constellations.

In the next chapter, we give an overview of the existing technologies used in our system. In Chapter 3, we propose a novel methodology for designing multilevel polar codes that work effectively with a multidimensional Grassmannian signalling and we propose a novel set partitioning algorithm that works for such constellations and is also applicable to arbitrary, not necessarily structured, multidimensional signalling schemes. In Chapter 4, we show the benefits of the new system and compare it with other existing schemes in simulation results. Finally, in Chapter 5, the conclusions and suggestions for future work are given.

Chapter 2

Background

2.1 Polar Codes

Polar codes are a recent invention by Erdal Arikan that represent a class of error-correcting codes with the ability to achieve the capacity of discrete memoryless channels. These codes are able to achieve this by using an effect known as channel polarization, where channels are categorized as being either good or bad. By recursively applying such polarization transformation over the resulting channels, the reliabilities of the synthesized channels will show significant difference: the “good ones get better and the bad get worse” [8]. The channels get more distinctly polarized as the code length is increased and the good channels can be chosen to transmit information bits over while the others are frozen (set to a zero). The polarization effect, the encoding, and the decoding methods will be further elaborated upon in the following sections.

2.1.1 Channel Polarization

The channel polarization effect is best described using an illustration [8]. Fig. 2.1a shows a 2 input - 2 output encoder using two binary erasure channels (W, W) each with erasure probability of 0.5 and therefore, a channel capacity of $I(W) = 0.5$. The two independent

Fig. 2.1: **Examples of channel polarization using a binary erasure channel with erasure probability = 0.5.**

channels are combined using one modulo-2 addition operation ($\oplus$) and this produces a compound channel with a total capacity of $2I(W)$. After splitting the compound channel into two channels W^- and W^+ , the capacity of each channel was derived in [7] as $I(W^-) = I(W)^2$ and $I(W^+) = 2I(W) - I(W)^2$. In this example, we see that polarization of the channels has already started with $I(W^-) = 0.25$ and $I(W^+) = 0.75$.

This polarization effect can be seen even more when the number of channels is increased as shown in Fig. 2.1b. In this case, we have two independent copies of the channels W^- and W^+ that receive inputs that have been combined as shown in Fig. 2.1a. Channels W^{--} and W^{-+} are combined such that the compound channel has a capacity of $2I(W^-)$. Therefore, $I(W^{--}) = I(W^-)^2 = 0.0625$ and $I(W^{-+}) = 2I(W^-) - I(W^-)^2 = 0.4375$. Similarly, the channels W^{++} and W^{+-} are combined and the capacity of the compound channel is $2I(W^+)$. The capacities are then evaluated as $I(W^{+-}) = I(W^+)^2 = 0.5625$ and $I(W^{++}) = 2I(W^+) - I(W^+)^2 = 0.9375$.

The capacities can be evaluated recursively for any block size in the same manner as shown in the above example. This process can be repeated for larger and larger block lengths and the channels will become more and more polarized as their capacities tend either towards 1 or 0. The polarization effect works not only for a BEC, it can be generalised to any discrete memoryless channel and even channels with continuous outputs such as the AWGN channel

Fig. 2.2: *Polar code graph.*

[13]. Furthermore, for the AWGN channel, the polarization effect applies not only to the capacity but also to the bit error rate (BER). That is, the BER tends to zero for the good sub-channels and 0.5 for the bad ones.

2.1.2 Encoding

Polar codes are linear block codes and therefore can be represented as $\mathbf{c} = \mathbf{u} \mathbf{G}_N$, where $\mathbf{c}$ is the codeword, $\mathbf{u} \in \{0, 1\}^N$ is the message word, $\mathbf{G}_N$ is the $N \times N$ generator matrix and $N = 2^n$ is the length of the code. This generator matrix is an n -th Kronecker power of $\mathbf{F}$ which is given as $\mathbf{F} = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$ with $\mathbf{G}_2 = \mathbf{F}$, $\mathbf{G}_4 = \begin{bmatrix} \mathbf{F} & 0 \\ 0 & \mathbf{F} \end{bmatrix}$ and $\mathbf{G}_N = \begin{bmatrix} \mathbf{G}_{N/2} & 0 \\ 0 & \mathbf{G}_{N/2} \end{bmatrix}$.

Fig. 2.2 ([14]) shows the pictorial representation of an $N = 2^n = 8$ polar encoder where

i, j represents each node in the graph with $1 \leq i \leq N$ rows and $0 \leq j \leq n$ columns. The message bits $\mathbf{u}$ are passed through the graph from left to right one column at a time and the corresponding bits $v_{i,j}$ are calculated according to the butterfly patterns shown in the figure. Each bit at the top right edge of each butterfly is calculated by adding the bits of both of the left sides together, while the bit at the bottom right corner is equal to that at the bottom left node. The value calculated at the node at row i and column j is

$$v_{i,j} = \begin{cases} v_{i,j-1} \oplus v_{i+2^{j-1},j-1}, & \left\lfloor \frac{i}{j-1} \right\rfloor \equiv 0 \pmod{2} \\ v_{i,j-1}, & \text{otherwise} \end{cases} \quad (2.1)$$

for all $j > 0$. When the process is complete, $\mathbf{c}$ is a linear combination of $\mathbf{u}$ as described in (2.1). As shown in Figs. 2.1a, 2.1b, and 2.2, the length of $\mathbf{c}$ and that of $\mathbf{u}$ are the same. However, with polar codes, some elements of $\mathbf{u}$ are set to zero while the others carry the information. Unlike other block codes which are designed to maximize the minimum Hamming distance between codewords, the channel selection principle used for polar codes involves selecting the channels that minimize the error rate of the channels that carry information. We are able to choose the better channels due to the process of channel polarization that occurs in the encoder.

2.1.3 Decoding

The basic successive cancellation decoding, developed by Arikan, was used in this thesis. SC decoding can be viewed as a soft/hard message passing algorithm over the trellis of polar codes [8]. This process is illustrated in Fig. 2.2 where the soft symbols (LLRs) are generally propagated from right to left. Each node at the upper left corner of a butterfly will be referred to as an upper node and each node at the bottom left corner will be referred to as a lower node. The LLRs at each node need to be calculated in order to properly decode $u_{i,0}$. These calculations are shown in (2.2); the first formula applies to the upper nodes while the

Fig. 2.3: **BICM block diagram.**

second formula applies to the lower nodes.

$$\lambda_{i,j} = \begin{cases} \ln \left[\frac{\exp\{\lambda_{i,j+1}\} \exp\{\lambda_{i+2^j,j+1}\} + 1}{\exp\{\lambda_{i,j+1}\} + \exp\{\lambda_{i+2^j,j+1}\}} \right], & \lfloor \frac{i-1}{2^j} \rfloor \equiv 0 \pmod{2} \\ \lambda_{i,j+1} + (1 - 2v_{i-2^j,j})\lambda_{i-2^j,j+1}, & \text{otherwise} \end{cases} \quad (2.2)$$

As the equation shows, the upper node calculations require the knowledge of the LLRs in the higher stages (columns to the right) while those of the lower node require $v_{i-2^j,j}$ which is a hard decision on the LLRs of the corresponding upper nodes. As such, the upper node LLRs are calculated first by traversing the trellis from right to left. When the lowest stage ($j = 0$) is reached, a hard decision is made on the bit that was sent with $u_{1,0} = v_{1,0} = 0$ if $\lambda_{1,0} > 0$, otherwise $u_{1,0} = 1$. The exception occurs for frozen bits which are set to 0. Once $v_{1,0}$ is determined, $v_{2,0}$ can then be determined using the lower node calculation (2.2). The decoder then traverses in the reverse direction to apply the hard decisions made to calculate $v_{1,1}$ and $v_{2,1}$ using (2.1). Once these are calculated, the lower LLRs $\lambda_{3,1}$ and $\lambda_{4,1}$ can be determined and thus $v_{3,0}$ and $v_{4,0}$ can be calculated. This recursive pattern continues until all the message bits are decoded.

2.2 Multilevel Codes

Bit-interleaved coded modulation (BICM) [11] is the most widely used technique for combining powerful error correcting codes, such as turbo and LDPC codes, with non-binary modulation schemes such as QAM. Fig. 2.3 shows the structure of a BICM system. This system involves concatenating a length N encoder with a modulator over a signal constel-

Fig. 2.4: *Block diagram for multilevel codes.*

lation of size M , through a bit interleaver and a mapper with Gray labelling [15]. With BICM the interleaver should be designed taking into account the structure of the code and the constellation mapping [15]. An extension of BICM is BICM with iterative detection and decoding (BICM-IDD), whereby detection and decoding are performed iteratively, with feedback from the decoder used to improve the reliability of the detector. BICM-IDD was used with Grassmannian signalling in [5] and was the best known technique for non-coherent communication over the MIMO fading channel.

Although BICM and BICM-IDD work well with turbo and LDPC codes, it was shown in [9, 16] that, for polar codes, a different type of coded modulation technique, multilevel coding, works better. Multilevel coding, as shown in Fig. 2.4, involves concatenating m encoders each of length N' with a modulator over a signal constellation of size M where $M = 2^m$ and $N = mN'$.

2.2.1 Transmitter

The MLC approach developed by Imai *et al* in [10], involves protecting each address bit $B_{i,l}$ of the signal point by a single binary code at level l [12]. That is, for a signal constellation

with M points and $m = \log_2 M$ bits per signal point, there are m encoders – one for each bit level. Any code, such as block codes or convolutional codes, may be used as the component codes. Each component code is designed to complement the reliability of the bit levels which it encodes, therefore a suitable mapping to the constellation is required to ensure the variance in bit level reliabilities. It was further shown in [12] that the set-partitioning (SP) label for a constellation used in conjunction with multilevel codes is the best technique because SP labelling leads to large bit level variances compared to other types of labelling schemes such as Gray labelling. Each of the m bit levels can therefore be encoded such that the code bits of the first encoder will be mapped to the least reliable bit level while the m^{th} encoder's code bits will be mapped to the most reliable bit level. Each component code has a length N' and therefore the encoders output a total of mN' bits to the mapper. Because each bit level can be thought of as its own channel with a certain capacity, the component code rates are chosen such that the high capacity channels (channels with few error rates) use a higher rate code than those with low capacity. Each component code has a code rate R_i that is chosen such that the overall rate is $R = \frac{1}{m} \sum_{i=1}^m R_i$ [16].

2.2.2 Receiver

The codewords are mapped to the signal constellation symbols with the least reliable bits being mapped to the most significant bit position and vice versa. The receiver uses a multi-stage decoding (MSD) technique ([10]) in which each code is decoded individually starting from the least reliable level and using the decisions made in the prior decoding levels/stages.

The MSD receiver used in this system uses soft decision decoding that calculates the log likelihood ratio (LLR) of each bit given the received signal matrix as [5]

$$\lambda_l = \ln \frac{\Pr\{c_l = 0 | \mathbf{Y}, c_1, \dots, c_{l-1}\}}{\Pr\{c_l = 1 | \mathbf{Y}, c_1, \dots, c_{l-1}\}} = \ln \frac{\sum_{\mathbf{X} \in \chi_{l,0}} \Pr\{\mathbf{Y}|\mathbf{X}\}}{\sum_{\mathbf{X} \in \chi_{l,1}} \Pr\{\mathbf{Y}|\mathbf{X}\}} \quad (2.3)$$

where $\mathbf{X}$ represents a matrix in the transmitted constellation, $\mathbf{Y}$ is the received signal matrix,

l represents the current bit level and the set $\chi_{l,k}$ contains all the matrices in the constellation with $B_{i,l} = k$.

As (2.3) shows, the de-mapping of each bit level relies on knowledge of the originally transmitted bits of the bit levels before it. When the LLRs for a particular level are calculated and the bits are decoded, the receiver passes those bits back into an encoder to obtain an approximation of the bits that were transmitted. These approximate codeword bits are then used in the de-mapping of the next bit level and so on (Fig. 2.4).

The de-mapping process can be the computational bottleneck of the system, especially for large constellations, as every single point in the constellation needs to be compared with each received signal matrix. Methods exist that reduce the complexity of the de-mapper for the Grassmannian constellation such as the list based de-mapper [5], however these methods are not explored in this thesis.

2.3 Set-Partitioning Mapping

As stated previously, the labelling of the signal constellation used for MLC is very important. Ungerboeck proposed a set partitioning algorithm [17] for regular constellations, such as PSK or QAM, that involves dividing the signal points into two subsets in such a way that the minimum Euclidean distance between any two points in a subset is greater than the minimum distance in the whole constellation. All the points in one subset are assigned a bit value of 0 in the first bit position, while all the points in the other subset are assigned a 1. Each subset is in turn divided into two subsets, again with increasing minimum distance between points within a subset, and all points within one of the new subsets are assigned a bit value of 0 in the second position, while the points within the other new subset are assigned a 1. This process is repeated until each subset contains only one point, and all points have been assigned a unique value of $m = \log_2 M$ bits, where M is the number of points in the constellation (see Fig. 2.5). Forney later provided a formalized algorithm for partitioning

Fig. 2.5: Ungerboeck's method of set partitioning a 16-QAM constellation.

Fig. 2.6: MIMO channel.

constellations when the signal points fall on a regular multidimensional lattice [18], but there are no general-purpose algorithms that work with irregular constellations.

2.4 Grassmannian Signalling

Fig. 2.6 shows a typical MIMO channel model with two transmit and two receive antennas with fading channel coefficients $h_{1,1}$, $h_{1,2}$, $h_{2,1}$, and $h_{2,2}$. In order to estimate those coefficients, a natural method to use is a training-based technique. Here, symbols known by the transmitter and receiver are sent over the channel and the receiver can use them to estimate

the channel. The information symbols can then be sent over the channel, for as long as the channel coefficients remain unchanged. This method works well for situations where the channel remains constant for long periods of time (that is, the channel coherence time, T , is large) and frequent channel estimation, which increases the processing overhead, is not required. Furthermore, training based schemes fall short of attaining the full channel capacity [2]. Under fast fading conditions, as considered in this thesis, such a training based scheme might not be ideal. Alternatively, within the realm of non-coherent MIMO communication, Grassmannian signalling has been shown to approach the ergodic capacity of frequency-flat block fading channels. Here, a block is defined as T channel uses (i.e. the block length = T). It was shown in [2, 3] that, for this system with N_t transmit and N_r receive antennas and a coherence time of T operating at high SNRs, the ergodic capacity of the channel can be achieved by codebooks of unitary matrices that are isotropically distributed on the (compact) Grassmann manifold provided that T satisfies

$$T \geq \min\{N_t, N_r\} + N_r, \quad (2.4)$$

In addition, in order to attain the maximum number of independent channels exploited by the transmitter (the communication degrees of freedom), for such a system, the number of transmit antennas, N_t , should be

$$N_t = \min \left\{ \left\lfloor \frac{T}{2} \right\rfloor, N_r \right\}. \quad (2.5)$$

The received signal is given by

$$\mathbf{Y} = \mathbf{X}\mathbf{H} + \mathbf{W} \quad (2.6)$$

or

$$\begin{bmatrix} y_{1,1} & \cdots & y_{1,N_r} \\ y_{2,1} & \cdots & y_{2,N_r} \\ \vdots & \ddots & \vdots \\ y_{T,1} & \cdots & y_{T,N_r} \end{bmatrix} = \begin{bmatrix} x_{1,1} & \cdots & x_{1,N_t} \\ x_{2,1} & \cdots & x_{2,N_t} \\ \vdots & \ddots & \vdots \\ x_{T,1} & \cdots & x_{T,N_t} \end{bmatrix} \begin{bmatrix} h_{1,1} & \cdots & h_{1,N_r} \\ h_{2,1} & \cdots & h_{2,N_r} \\ \vdots & \ddots & \vdots \\ h_{N_t,1} & \cdots & h_{N_t,N_r} \end{bmatrix} + \begin{bmatrix} w_{1,1} & \cdots & w_{1,N_r} \\ w_{2,1} & \cdots & w_{2,N_r} \\ \vdots & \ddots & \vdots \\ w_{T,1} & \cdots & w_{T,N_r} \end{bmatrix} \quad (2.7)$$

where $\mathbf{H}$ is an $N_t \times N_r$ fading channel matrix with independent elements which change independently after every block of T channel uses. Each channel coefficient, $h_{i,j}$, has a complex Gaussian distribution, $\mathcal{CN}(0, 1)$ with a mean of 0 and a variance of, $\mathbf{E}[|h_{i,j}|^2] = 1$ and the real and imaginary parts are independent. Similarly, $\mathbf{W}$ is a $T \times N_r$ additive white Gaussian noise matrix where its elements are independent and have complex Gaussian distribution $\mathcal{CN}(0, \sigma_W^2)$ with variance $\mathbf{E}[|w_{i,j}|^2] = \sigma_W^2$ and the real and imaginary parts are independent.

$\mathbf{X}$ is the transmitted $T \times N_t$ matrix that is ‘unitary’ in the sense that $\mathbf{X}^\top \mathbf{X} = \mathbf{I}_{N_t}$. The superscript $^\top$ denotes conjugate transpose and $\mathbf{I}_{N_t}$ is the $N_t \times N_t$ identity matrix. These matrices are isotropically distributed, and $E_{\mathbf{X}} = \|\mathbf{X}\|^2 = N_t$ where $\|\cdot\|$ is the Frobenius norm which is defined as $\sqrt{\text{Tr}(\mathbf{X}^\top \mathbf{X})}$. The average SNR can then be calculated as

$$\rho = \frac{\mathbf{E}[\|\mathbf{X}\mathbf{H}\|^2]}{\mathbf{E}[\|\mathbf{W}\|^2]}, \quad (2.8)$$

where

$$\mathbf{E}[\|\mathbf{X}\mathbf{H}\|^2] = \mathbf{E}[\text{Tr}\{\mathbf{H}^\top \mathbf{X}^\top \mathbf{X} \mathbf{H}\}] = \mathbf{E}[\text{Tr}\{\mathbf{H}^\top \mathbf{H}\}] = N_t N_r, \quad (2.9)$$

and

$$\mathbf{E}[\|\mathbf{W}\|^2] = T N_r \sigma_W^2, \quad (2.10)$$

so that

$$\rho = \frac{N_t N_r}{T N_r \sigma_W^2} = \frac{N_t}{T \sigma_W^2}. \quad (2.11)$$

For uncoded communication, the receiver uses maximum likelihood (ML) detection by searching the entire constellation to find the signal which maximizes the likelihood function $\Pr(\mathbf{Y}|\mathbf{X})$ where [2]

$$\begin{aligned}\Pr(\mathbf{Y}|\mathbf{X}) &= \frac{\exp \left\{ \frac{-1}{\sigma_W^2} \text{Tr} \left(\mathbf{Y}^\top \left(\mathbf{I}_T - \frac{1}{1+\sigma_W^2} \mathbf{X}\mathbf{X}^\top \right) \mathbf{Y} \right) \right\}}{(\pi\sigma_W^2)^{TN_r} \left(1 + \frac{1}{\sigma_W^2} \right)^{N_t N_r}} \\ &\propto \exp \left\{ \frac{\|\mathbf{X}^\top \mathbf{Y}\|^2}{\sigma_W^2 (1 + \sigma_W^2)} \right\}\end{aligned}\tag{2.12}$$

Having to search through the entire constellation to find the best candidate introduces a high computational cost to this ML detector. We will further discuss the computational load of this method in Chapter 4.

Another way to view how the receiver detects the signals is as follows: when $\mathbf{X}$ is multiplied with $\mathbf{H}$, its basis vectors, which span an N_t -dimensional subspace, are rotated and scaled within the same sub-space. Adding noise to this introduces perturbations within the subspace. The receiver then has to decide which sub-space was transmitted on, irrespective of its basis.

Chapter 3

System Design

3.1 System Set-up

We now have all the components needed for the design of our new system. As stated in the previous chapter, polar codes work well in an MLC system. This is due to the fact that MLCs partition the M point channel into $m = \log_2 M$ bit channels which each have different reliabilities. Therefore, each corresponding encoder is designed at a rate that matches the reliability of the bit channel. This requires that the rates for the component encoders be easily fine-tunable, which is an advantageous property that polar codes have. This point was highlighted by Seidl *et al.* in [16] where multilevel polar codes were presented for regular M -ary channels. For the system presented in this thesis, we therefore use polar codes as our component codes (see Fig. 2.4). Grassmannian constellations are used as our non-coherent signals as they provide better error rate performance compared with the best known training based scheme, the Golden codes [2]. As a whole, the polar MLC and Grassmannian constellation system works as follows: the channel is first partitioned into m bit levels, then, each level is ‘assigned’ polar codes which must be specifically designed for that bit channel. The novelties in the code design and the set partitioning for our system will be discussed in the proceeding sections. After the design is complete, the system can

then be set-up with the low rate polar codes mapped to the least reliable Grassmannian channel and vice versa. At the receiver, the likelihood expression, $P\{\mathbf{Y}|\mathbf{X}\}$, is determined where

$$P\{\mathbf{Y}|\mathbf{X}\} = \beta \exp \left\{ \frac{\|\mathbf{X}^\top \mathbf{Y}\|}{\sigma_W^2(1 + \sigma_W^2)} \right\}. \quad (3.1)$$

where β is a constant of proportionality. Once this is done, the MSD begins and the LLRs for the first level of reliability are calculated (2.12). These are then passed to the polar decoder for the first level which decodes the bits. Estimates of the coded bits that were sent are made which are then used to decode the bits of the second level and so on. The m^{th} level decoder makes use of the estimates of the coded bits from all the levels before it to properly decode the bits of that level.

This combination of polar codes and Grassmannian constellations itself is a novel one.

3.1.1 Set Partitioning Algorithm

The set partitioning of the signal constellation is central to designing a good multilevel coding system. Ungerboeck's algorithm works well for regular constellations, and could also be used with small irregular ones, but implementation becomes problematic for large irregular multidimensional constellations. We therefore propose a novel, formal algorithm that works with any arbitrary constellation and has low implementation complexity. This algorithm works in the opposite order from Ungerboeck's, starting with M subsets, at 'depth' $l = 1$, containing just one point each that are paired together to form $M/2$ subsets of two points each. These subsets at depth $l = 2$ are in turn combined to form $M/4$ subsets of four points each, and so on. This process continues until there is just one subset of M points at depth $l = m$. The depth therefore can be seen as number of times the subsets have been combined.

Given a constellation $\mathcal{C} = \{\mathbf{X}_i | i \in \{1, \dots, M\}\}$, the ultimate objective of set partitioning

is to create a mapping between sequences of $\log_2 M$ bits and points in $\mathcal{C}$. Let $B_{i,l}$ be the l^{th} bit of the mapping for $\mathbf{X}_i$. Furthermore, let $\mathbf{S}_{l,i}$ be the set containing the indices of the constellation points in the i^{th} subset at depth l . Note that $|\mathbf{S}_{l,i}| = 2^{l-1}$ for $i \in \{1, \dots, M/2^{l-1}\}$ and $l \in \{1, \dots, \log_2 M\}$. Starting with M subsets that each contain one index point, $\mathbf{S}_{1,i} = i$, the algorithm combines subsets at depth $l-1$ to produce subsets at depth l . This combination is based on the inter-subset distance table,

$$\mathcal{D}_l(i, j) = \min_{a \in \mathbf{S}_{l,i}, b \in \mathbf{S}_{l,j}} d(\mathbf{X}_a, \mathbf{X}_b) \quad (3.2)$$

where $d(\mathbf{X}_a, \mathbf{X}_b)$ is the distance between constellation points $\mathbf{X}_a$ and $\mathbf{X}_b$. That is, $\mathcal{D}_l(i, j)$ contains the distance between the two closest points in subsets i and j at depth l which is the standard method for determining the distance between two subsets of points [19]. Whereas for coherent detection the Euclidean distance is normally used, it was shown in [2] that for non-coherent detection of Grassmannian constellations, the chordal Frobenius norm is more appropriate. The distance between $\mathbf{X}_a$ and $\mathbf{X}_b$ is given by

$$d(\mathbf{X}_a, \mathbf{X}_b) = \sqrt{2N_t - 2 \sum_{k=1}^{N_t} \sigma_k} \quad (3.3)$$

where σ_k are the singular values of $\mathbf{X}_a^T \mathbf{X}_b$.

When combining subsets it is desirable to pair subsets that are as far apart as possible. That is, a greedy algorithm could pair subset i with subset $\arg \max_j \mathcal{D}_l(i, j)$. We note, however, that at depth l there will always be at least one pairing with a distance of only $\Delta_l = \min_i \max_j \mathcal{D}_l(i, j)$. Since the system performance depends mostly on this minimum distance, Δ_l , and making the most greedy choices for each i tends to make Δ_{l+1} smaller, we have found it better to pair subset i with the subset that is closest to i but with distance no less than Δ_l . This will ensure that the minimum distance is still Δ_l , without being needlessly greedy. Furthermore, because of numerical rounding errors in calculating the

chordal Frobenius norm, a more robust approach is to include a slight error tolerance and accept pairs with distances greater than $\Delta_l - \epsilon$ for some small ϵ (e.g. $\epsilon = 10^{-3}$). The set partitioning algorithm is then formally given by the following pseudo-code:

Set partitioning algorithm

- Initialize $\mathbf{S}_{1,i} \leftarrow \{i\}, M_1 \leftarrow M, \mathcal{D}_1(i, j) \leftarrow d(\mathbf{X}_i, \mathbf{X}_j)$
- for $l = 1, 2, \dots, \log_2 M$
 - $\Delta_l \leftarrow \min_{i \in \{1, \dots, M_l\}} \max_{j \in \{1, \dots, M_l\}} \mathcal{D}_l(i, j)$ * determine the minimum pairing distance
 - for $i = 1, 3, \dots, M_l - 1$ * $M_l = M/2^{l-1}$ is the number of subsets
 - * $j \leftarrow \arg \min_{j \in \{i+1, \dots, M_l\}, \mathcal{D}_l(i, j) > \Delta_l - \epsilon} \mathcal{D}_l(i, j)$ * find index/subset to pair with
 - * $B_{a,l} \leftarrow 0 \quad \forall a \in \mathbf{S}_{l,i}$ * label bit = 0 for all elements in the subset i
 - * $B_{b,l} \leftarrow 1 \quad \forall b \in \mathbf{S}_{l,j}$ * label bit = 1 for all elements in the subset j
 - * $\mathbf{S}_{l+1, (i+1)/2} \leftarrow \mathbf{S}_{l,i} \cup \mathbf{S}_{l,j}$ * subsets are paired into 1 set
 - * swap row j of $\mathcal{D}_l$ with row $i + 1$
 - * swap column j of $\mathcal{D}_l$ with column $i + 1$
 - $M_{l+1} \leftarrow M_l/2$ * number of the new set is half that of old
 - $\mathcal{D}_{l+1}(i, j) \leftarrow \min\{\mathcal{D}_l(2i-1, 2j-1), \mathcal{D}_l(2i-1, 2j), \mathcal{D}_l(2i, 2j-1), \mathcal{D}_l(2i, 2j)\} \quad \forall i, j \in 1, 2, \dots, M_{l+1}$

The two ‘swap’ operations in the algorithm keep the paired subsets next to each other in the table. By doing this, when the algorithm searches for pairs for i , it only searches through the as-of-yet unpaired indices $j \in \{i + 1, \dots, M_l\}$. Therefore, for each iteration of i , the set j gets smaller by two indices. This also helps to keep track of the paired sets because at the end of the iteration through i , every odd numbered index i is paired with the corresponding even numbered index $i + 1$. Thus the inter-subset distance table for the next depth, $\mathcal{D}_{l+1}(i, j)$ can easily be populated by considering the distance between all the points in one set to all

Fig. 3.1: Sample 16-QAM Constellation.

the points in the other. The last line of the algorithm does this by taking advantage of the side-by-side pairings of the rows and columns of $\mathcal{D}_l$.

Considering an example using a 16-QAM constellation as shown in Fig. 3.1 we obtain a distance metric table for $\mathcal{D}_1$ shown in Fig. 3.2. Here, there are $M_1 = 16$ sets of $\mathbf{S}_{1,i}$ each containing 1 point. The maximum distance from each point, highlighted in the figure, is obtained and the minimum of these is selected. In this example, $\Delta_1 = 4\sqrt{2}$. Point $i = 1$ is paired with point $j = 11$ then i is assigned a bit value of 0 while j is assigned a bit value of 1 in the first bit position (i.e. the least significant bit). These two points are then concatenated into one set $\mathbf{S}_{2,1}$. Row $i + 1 = 2$ is swapped with row $j = 11$ such that the paired sets are next to each other and row 2 is now in the eleventh position (shown on top of the table). The same is done for the columns. In the next iteration, $i = 3$ and is paired with $j = 9$. The bits are assigned, the sets are concatenated and the rows and columns are swapped. This continues till $i = 15$ and the paired sets are $\mathbf{S}_{2,1} = \{1, 11\}$, $\mathbf{S}_{2,2} = \{3, 9\}$, $\mathbf{S}_{2,3} = \{5, 15\}$, $\mathbf{S}_{2,4} = \{7, 13\}$, $\mathbf{S}_{2,5} = \{4, 10\}$, $\mathbf{S}_{2,6} = \{2, 12\}$, $\mathbf{S}_{2,7} = \{8, 14\}$ and $\mathbf{S}_{2,8} = \{6, 16\}$. At the end, there are $M_2 = 8$ sets of $\mathbf{S}_{2,i}$ which each contain 2 points. The distance table $\mathcal{D}_2$,

	4				2				8				6			
$\mathcal{D}_1$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	0	2	4	6	2	$2\sqrt{2}$	$2\sqrt{5}$	$2\sqrt{10}$	4	$2\sqrt{5}$	$4\sqrt{2}$	$2\sqrt{13}$	6	$2\sqrt{10}$	$2\sqrt{13}$	$2\sqrt{18}$
2	2	0	2	4	$2\sqrt{2}$	2	$2\sqrt{2}$	$2\sqrt{5}$	$2\sqrt{5}$	4	$2\sqrt{5}$	$4\sqrt{2}$	$2\sqrt{10}$	6	$2\sqrt{10}$	$2\sqrt{13}$
3	4	2	0	2	$2\sqrt{5}$	$2\sqrt{2}$	2	$2\sqrt{2}$	$4\sqrt{2}$	$2\sqrt{5}$	4	$2\sqrt{5}$	$2\sqrt{13}$	$2\sqrt{10}$	6	$2\sqrt{10}$
4	6	4	2	0	$2\sqrt{10}$	$2\sqrt{5}$	$2\sqrt{2}$	2	$2\sqrt{13}$	$4\sqrt{2}$	$2\sqrt{5}$	4	$2\sqrt{18}$	$2\sqrt{13}$	$2\sqrt{10}$	6
5	2	$2\sqrt{2}$	$2\sqrt{5}$	$2\sqrt{10}$	0	2	4	6	2	$2\sqrt{2}$	$2\sqrt{5}$	$2\sqrt{10}$	4	$2\sqrt{5}$	$4\sqrt{2}$	$2\sqrt{13}$
6	$2\sqrt{2}$	2	$2\sqrt{2}$	$2\sqrt{5}$	2	0	2	4	$2\sqrt{2}$	2	$2\sqrt{2}$	$2\sqrt{5}$	$2\sqrt{5}$	4	$2\sqrt{5}$	$4\sqrt{2}$
7	$2\sqrt{5}$	$2\sqrt{2}$	2	$2\sqrt{2}$	4	2	0	2	$2\sqrt{5}$	$2\sqrt{2}$	2	$2\sqrt{2}$	$4\sqrt{2}$	$2\sqrt{5}$	4	$2\sqrt{5}$
8	$2\sqrt{10}$	$2\sqrt{5}$	$2\sqrt{2}$	2	6	4	2	0	$2\sqrt{10}$	$2\sqrt{5}$	$2\sqrt{2}$	2	$2\sqrt{13}$	$4\sqrt{2}$	$2\sqrt{5}$	4
9	4	$2\sqrt{5}$	$4\sqrt{2}$	$2\sqrt{13}$	2	$2\sqrt{2}$	$2\sqrt{5}$	$2\sqrt{10}$	0	2	4	6	2	$2\sqrt{2}$	$2\sqrt{5}$	$2\sqrt{10}$
10	$2\sqrt{5}$	4	$2\sqrt{5}$	$4\sqrt{2}$	$2\sqrt{2}$	2	$2\sqrt{2}$	$2\sqrt{5}$	2	0	2	4	$2\sqrt{2}$	2	$2\sqrt{2}$	$2\sqrt{5}$
11	$4\sqrt{2}$	$2\sqrt{5}$	4	$2\sqrt{5}$	$2\sqrt{5}$	$2\sqrt{2}$	2	$2\sqrt{2}$	4	2	0	2	$2\sqrt{5}$	$2\sqrt{2}$	2	$2\sqrt{2}$
12	$2\sqrt{13}$	$4\sqrt{2}$	$2\sqrt{5}$	4	$2\sqrt{10}$	$2\sqrt{5}$	$2\sqrt{2}$	2	6	4	2	0	$2\sqrt{10}$	$2\sqrt{5}$	$2\sqrt{2}$	2
13	6	$2\sqrt{10}$	$2\sqrt{13}$	$2\sqrt{18}$	4	$2\sqrt{5}$	$4\sqrt{2}$	$2\sqrt{13}$	2	$2\sqrt{2}$	$2\sqrt{5}$	$2\sqrt{10}$	0	2	4	6
14	$2\sqrt{10}$	6	$2\sqrt{10}$	$2\sqrt{13}$	$2\sqrt{5}$	4	$2\sqrt{5}$	$4\sqrt{2}$	$2\sqrt{2}$	2	$2\sqrt{2}$	$2\sqrt{5}$	2	0	2	4
15	$2\sqrt{13}$	$2\sqrt{10}$	6	$2\sqrt{10}$	$4\sqrt{2}$	$2\sqrt{5}$	4	$2\sqrt{5}$	$2\sqrt{5}$	$2\sqrt{2}$	2	$2\sqrt{2}$	4	2	0	2
16	$2\sqrt{18}$	$2\sqrt{13}$	$2\sqrt{10}$	6	$2\sqrt{13}$	$4\sqrt{2}$	$2\sqrt{5}$	4	$2\sqrt{10}$	$2\sqrt{5}$	$2\sqrt{2}$	2	6	4	2	0

$\mathcal{D}_2$	1,11	3,9	5,15	7,13	4,10	2,12	8,14	6,16
1,11	0	4	2	2	2	2	$2\sqrt{2}$	$2\sqrt{2}$
3,9	4	0	2	2	2	2	$2\sqrt{2}$	$2\sqrt{2}$
5,15	2	2	0	4	$2\sqrt{2}$	$2\sqrt{2}$	2	2
7,13	2	2	4	0	$2\sqrt{2}$	$2\sqrt{2}$	2	2
4,10	2	2	$2\sqrt{2}$	$2\sqrt{2}$	0	4	2	2
2,12	2	2	$2\sqrt{2}$	$2\sqrt{2}$	4	0	2	2
8,14	$2\sqrt{2}$	$2\sqrt{2}$	2	2	2	2	0	4
6,16	$2\sqrt{2}$	$2\sqrt{2}$	2	2	2	2	4	0

	5,15,7,13			
$\mathcal{D}_3$	1,11,3,9	5,15,7,13	4,10,2,12	8,14,6,16
1,11,3,9	0	2	2	$2\sqrt{2}$
5,15,7,13	2	0	$2\sqrt{2}$	2
4,10,2,12	2	$2\sqrt{2}$	0	2
8,14,6,16	$2\sqrt{2}$	2	2	0

	1,11,3,9, 8,14,6,16		4,10,2,12, 5,15,7,13
$\mathcal{D}_4$	1,11,3,9, 8,14,6,16		4,10,2,12, 5,15,7,13
1,11,3,9, 8,14,6,16	0		2
4,10,2,12, 5,15,7,13	2		0

Fig. 3.2: Distance table for partitioning a 16-QAM constellation.

shown in the figure, is generated by taking the minimum distance between each point in the new sets and all the sets. The maximum distance from each set is obtained and the minimum of these is selected with $\Delta_2 = 4$. Sets $\mathbf{S}_{2,1}$ and $\mathbf{S}_{2,2}$ are concatenated with all the points in the former set assigned a value of 0 and all the points in the latter a value of 1 in the second bit position. No swaps are necessary here because the paired set is already adjacent. In the second iteration, $i = 3$ and the sets $\mathbf{S}_{2,3}$ and $\mathbf{S}_{2,4}$ are concatenated and

Fig. 3.3: Set partitioning a 16-QAM signal using our algorithm.

the bits are assigned. This continues till all the sets are paired and concatenated and the bits are assigned. In the next level/depth, there are $M_3 = 4$ sets where $\mathbf{S}_{3,1} = \{1, 11, 3, 9\}$, $\mathbf{S}_{3,2} = \{5, 15, 7, 13\}$, $\mathbf{S}_{3,3} = \{4, 10, 2, 12\}$, and $\mathbf{S}_{3,4} = \{8, 14, 6, 16\}$. By taking the minimum distance between these sets, $\mathcal{D}_3$ is formed. The minimum of the maximum distance from each set is obtained and at this level, $\Delta_3 = 2\sqrt{2}$. $\mathbf{S}_{3,1}$ is paired with $\mathbf{S}_{3,4}$ and sets $\mathbf{S}_{3,2}$ and $\mathbf{S}_{3,4}$ are swapped in the distance table. The bits are labelled appropriately and the next two sets are paired, concatenated and labelled accordingly. In the last level with $M_4 = 2$ sets where $\mathbf{S}_{4,1} = \{1, 11, 3, 9, 8, 14, 6, 16\}$ and $\mathbf{S}_{4,2} = \{4, 10, 2, 12, 5, 15, 7, 13\}$, $\Delta_4 = 2$. The two sets are paired, all points in $\mathbf{S}_{4,1}$ are labelled 0 and $\mathbf{S}_{4,2}$ are labelled 1 in the fourth bit position, and these are concatenated giving a final set $\mathbf{S}_{5,1}$ which contains all of the points in the constellation. Thus, the set partitioning is done and a visual example can be seen in Fig. 3.3.

Fig. 3.4 shows the BER per bit level of a 16-QAM constellation partitioned using our algorithm. This bit variance can be obtained in simulation when the receiver detects the signals one bit at a time, assuming the previous bits have been detected correctly. Because the bits at the lesser depths have a greater distance apart in the signal space than those at

Fig. 3.4: *Bit error rate per bit of a 16-QAM constellation.*

the lower depths, i.e. $\Delta_l > \Delta_{l+1}$, those bits show a better performance.

We see therefore that our algorithm converges to the Ungerboeck solution when used on a 16-QAM constellation. This shows that while our algorithm may not necessarily be optimal, it is a good one.

The algorithm is applied to a 256 point Grassmannian constellation and the BERs per bit are shown in Fig. 3.5. All bit levels can distinctly be seen here.

3.2 System Design Methodology

Due to the polarization effect that occurs in the polar codes, data should be transmitted over the bit channels that have the best channel capacity while the others are not used. With a simple binary erasure channel, it is easy to calculate the capacities of each bit channel, however, with our system, this computation is not trivial. Therefore, code design was carried out using MATLAB simulations. The code design component of the system is very similar to the set-up described above where an M point Grassmannian constellation is combined with $m = \log_2 M$ component polar encoders each with length N' (where $mN' = N$). Because

Fig. 3.5: *Bit error rate per bit of a 256 point Grassmannian constellation.*

the positions of the frozen bit channels are not yet known however, the component codes all operate at a rate $R_i = 1$. The design methodology involves simulating the transmission of a large number of message words through the system at a specified design SNR, and recording the bit positions where the first errors occur. When polar codes are decoded using the successive cancellation decoder described in Chapter 2, the message bits are recovered one at a time, in order from the first to the last message bit. For the purpose of code design, the “first error” bit is defined as the first message bit to be decoded incorrectly in a message word (if a message word is decoded correctly then there is no first error bit for that word). The first error probability for a bit is the probability of error for a bit, given that all previous bits are known (i.e. are either frozen or have been decoded correctly). This error represents the error of the bit channel and does not include propagated errors while decoding. In order to record only first errors, the receiver would need to stop decoding a frame as soon as the first error is detected and keep track of how many times each bit position in the frame was

reached as well as how many times an error occurred there. With this method, the first error probabilities would likely not be accurately estimated in the higher level codes without simulating an extremely large number of message words. Therefore, we use a system where the receiver knows the bits that were transmitted and keeps track of every bit position in the received frame that an error occurred. Once an error is detected and registered, the receiver corrects the error, thereby preventing that error from propagating. In this way, the next error detected can be counted as a ‘first’ error for that particular bit position in the frame, so, the first errors can be calculated wherever they occur in every received frame. Eventually, with enough simulated bits, the total number of errors that occurred for each bit channel will be recorded and their respective first error probabilities can be calculated. This method of calculating first errors is necessary because our system uses a multilevel design where the de-mapping of each successive level depends on the correct detection of all the previous levels. Depending on what coding rate was required, the channels with the smallest first error probabilities would be chosen to transfer our data bits over.

The performance of the system can then be evaluated using normal Monte Carlo simulation. However, as shown in Fig. 3.6, the system performance is heavily dependent on the design SNR. Although the designed code tends to work well at the design SNR, its performance can be quite poor at other SNRs. It is often preferable to design a code with a given rate that achieves a given target FER at the lowest possible SNR. For example, from Fig. 3.6, to achieve a target FER of 10^{-2} , a design SNR of between 10 and 10.5 dB should be used. Since the optimal design SNR is not known in advance, we use a bisection search to find it. This search works with the observation that a code designed at a particular SNR will perform the best at that SNR. That is, for example, as shown in Fig. 3.6, the code designed at 9.5 dB has the best FER performance at that SNR while the code designed at 10.5 dB performs the best at that SNR. Note also that the two SNRs produce different optimal FERs (e.g. the 10.5 dB code is optimal at a FER of 3×10^{-3}). Therefore, the bisection search starts by choosing two SNRs, one low and one high, as our starting points

Fig. 3.6: **Effect of design SNR on the FER performance of our system.** $N' = 512$, rate $R = 4/5$, $M = 4096$.

such that when the code designed at the lower SNR (SNR_l) is simulated at that SNR, it produces an optimal FER greater than our target FER. Conversely, when the code designed at the higher SNR (SNR_h) is simulated at that SNR, the resulting FER is less than our target FER. A design SNR (SNR_d) is chosen such that $SNR_d = (SNR_l + SNR_h)/2$. The code is designed and simulated at SNR_d and its optimal FER is obtained. If the FER is equal to our target FER (with a small tolerance of $\pm\epsilon$), SNR_d is our target design SNR. If the FER is greater than our target FER, $SNR_l = SNR_d$, else, $SNR_h = SNR_d$. The algorithm repeats by calculating a new SNR_d with the new values of SNR_l and SNR_h until a design SNR produces a frame error rate approximately equal to our target FER. In order to prevent a possible infinite loop, a maximum number of iterations, $imax$, is included and once this number is reached, the algorithm ends with an error message. The algorithm is formally given in the pseudo-code below

Bisection algorithm

- $\mathcal{F}(x)$ is the function that designs and simulates at a given SNR, x , producing the optimal FER. SNR_l and SNR_h are chosen such that $\mathcal{F}(SNR_l) > \text{target FER} > \mathcal{F}(SNR_h)$.
- for $i = 1, 2, 3, \dots, imax$
 - $SNR_d = (SNR_l + SNR_h)/2$
 - if $\mathcal{F}(SNR_d) = \text{target FER} \pm \epsilon$, target SNR = SNR_d . stop program
 - $\mathcal{F}(SNR_d) > \text{target FER}$? $SNR_l = SNR_d : SNR_h = SNR_d$
- output error

This method differs from what is normally done where the FER is optimised for a chosen SNR [20].

Chapter 4

Simulation Results and Analysis

In this chapter, the bit error rate performance for the system is presented. All simulations were carried out using MATLAB and unless otherwise stated, all channels were simulated as block fading channels where one block is defined as a symbol of m bits for a 2×2 MIMO channel with a coherence time of $T = 4$. Where a Grassmannian constellation is used, $M = 4096$ while the Golden code constellation uses $M = 256$.

4.1 Performance of Designed System

To illustrate the performance gain obtained by our system, we compare the performance of the multi-level polar code, designed as described in the previous chapter for a target FER of 10^{-2} , with that of a BICM polar coded system. Fig. 4.1 shows the BER curves for these systems. A 4096 point Grassmannian constellation is used for both coding techniques and the error rate curve for an uncoded 4096 point Grassmannian constellation with quasi-Gray labelling is also shown in the figure for reference. The MLC polar system has a total frame length of $N = 3072$ (12 component codes each with $N' = 256$ bits) and the constellation is mapped using labels that were created by our set partitioning labelling algorithm. The BICM based system was used with a slightly longer frame length of $N = 4096$ and both coded schemes were simulated at a code rate of $R = \frac{1}{2}$. As expected, the coded systems show a

Fig. 4.1: Comparison of the coded vs uncoded schemes for a 4096 point Grassmannian constellation.

significant improvement over the uncoded case at moderately high SNRs. Furthermore, we see that MLC scheme improves over the BICM scheme by about 4 dB and it improves over the uncoded scheme by about 13 dB at a BER of 10^{-4} .

Having considered the performance gain of our polar MLC system over a BICM based one, we now consider the effect of choosing larger frame sizes and different coding rates for our codes. Fig. 4.2 shows the BER performance comparison between the designed scheme with $N = 49152$, and $N = 3072$ for both rate $\frac{1}{2}$ and $\frac{4}{5}$. Increasing N improves the bit error rate performance, as expected, by about 1.5 dB for the higher rate code and by 2 dB for the lower rate one at a BER of 10^{-4} . The difference between the rate $\frac{1}{2}$ and $\frac{4}{5}$ is approximately 2.25 dB for the longer code and 2.75 dB for the shorter at the same BER.

As with every coding based system, increasing the frame size improves the bit error rate performance, but not above the SNR threshold set by the channel capacity. In Fig. 4.3 we show the effect of increasing the frame size. The figure shows the results of a system with

Fig. 4.2: *Polar codes with 4096 point Grassmannian constellations.*

a 4096 point Grassmannian constellation combined with rate 4/5 polar codes of varying sub-code lengths. These sub-codes range from $N' = 8$ to $N' = 8192$ which corresponds to total code lengths of $N = 96$ to $N = 98304$ bits respectively. As expected, the longer the length of the polar codes, the better the BER performance. However, there must be a limit to how good these codes can get and we find this limit by considering the channel capacity. The SNR limit at capacity was calculated using the expression computed for Grassmannian constellations with equal number of transmit and receive antennas found in [1], which is

$$C = N_t \left(1 - \frac{N_t}{T}\right) \log_2 \rho + c_{N_t, N_t} \quad (4.1)$$

where

$$c_{N_t, N_t} = \frac{1}{T} \log_2 |G(T, N_t)| + N_t \left(1 - \frac{N_t}{T}\right) \log_2 \frac{T}{\pi e} + \left(1 - \frac{N_t}{T}\right) \mathbf{E}[\log_2 \det \mathbf{H} \mathbf{H}^T] \quad (4.2)$$

Fig. 4.3: 4096 point Grassmannian constellation with polar codes of different sub-code lengths with code rate $4/5$.

where ρ is the average SNR and $|G(T, N_t)|$, the volume of the Grassmann manifold, is given as

$$|G(T, N_t)| = \frac{\prod_{i=T-N_t+1}^T \frac{2\pi^i}{(i-1)!}}{\prod_{i=1}^{N_t} \frac{2\pi^i}{(i-1)!}}. \quad (4.3)$$

The SNR threshold for the constellation at rate $4/5$ (2.4 bits per channel use) is calculated to be 11.6 dB and this corresponds to an E_b/N_0 value of about 7.8 dB. As Fig. 4.3 shows, with a sub-code length of $N' = 8192$, this design is able to operate within 1.6 dB of the approximate non-coherent ergodic capacity at a BER of 10^{-4} . This is the closest to the channel capacity for a Grassmannian constellation that has been reached as far as we are aware.

Fig. 4.4: Different codes running with 4096 point Grassmannian constellation with rate $R=4/5$. All BICM figures use quasi-Gray labelling for the constellation. Multilevel code uses set partitioned labelling. Un-optimized BICM codes are optimized for a BPSK AWGN channel only.

4.2 Comparison With Other Existing Techniques

Thus far, the results seem to indicate that our new system performs quite well. However, it is useful to compare it with other existing coding techniques. Fig. 4.4 shows the comparison between our polar coding design and turbo, LDPC, and other BICM codes with a 4096 point Grassmannian constellation at a rate of $4/5$. The first curve shows the results of our multi-level polar code technique designed as described in Chapter 3 with $N = 24576$ ($N' = 2048$). The next curve shows the performance of a turbo coded system as designed in [6] using bit interleaved coded modulation with iterative detection and decoding (BICM-IDD), and quasi-Gray labelling for the constellation. It is of note that the results presented for that code give

the previous best known performance for a Grassmannian constellation until now. The third curve gives the performance of a BICM system using the standard DVB-S2 LDPC code with a Grassmannian constellation. The last two curves show the performance of a BICM polar code system with $N = 32768$; the first of the two uses a polar code designed for the AWGN channel while the other is designed for the Grassmannian channel. We see from the figure that despite the fact that the frame length used in our system is significantly shorter than all the others, we are able to outperform them all up to a BER of 10^{-4} . We also see that while the multilevel method works very well for polar codes with Grassmannian signalling, the single-level polar codes perform much worse than both turbo and LDPC codes using BICM. Thus, for polar codes, multilevel coding is the preferred coded modulation method. While not shown in the figure, LDPC and turbo codes are not as effective with multilevel coding because the method requires that the sub-code rates be easily fine-tunable. Since it is not trivial to create turbo or LDPC codes at a wide variety of code rates, single-level coding is used instead.

As was discussed in the previous chapter, the design of polar codes is dependent on the channel characteristics. The figure also shows what happens if we try to use polar codes designed for the AWGN channel for a Grassmannian channel (curve 4). Therefore, it is necessary to design the codes properly for the given channel. It is also of note that the polar encoder/decoder has significantly less complexity compared with the other two codes. Therefore, the polar scheme not only provides better error rate performance, but it does so with a much lower complexity. A more detailed complexity analysis is carried out later.

4.3 Results With a Training Based Scheme

The new system designed has thus far been shown to produce good error rate performance compared with other coding schemes. Next, we consider how our system fares with other non-coherent multidimensional constellations by adapting it for a training-based Golden

code constellation. The Golden code is a full-rate, full-diversity space-time block code with unbeaten performance in coherent systems. It is based on the Golden number $\frac{1+\sqrt{5}}{2}$ and is used for a 2×2 MIMO system. The training based Golden code has two phases: the first is the training phase where a known signal is sent from each transmitter and this is used to estimate the channel at the receiver. The next phase is an information bearing phase where data symbols are sent over the channel and the receiver can detect the signals using the estimated channel coefficients for as long as the channel remains unchanged.

During the training phase, a known signal is transmitted over only one of the antennas at a time, so the training phase requires N_t channel uses. The system model is given as,

$$\mathbf{Y}_{tr} = \kappa \mathbf{I}_{N_t} \mathbf{H} + \mathbf{W}_{tr} \quad (4.4)$$

where $\mathbf{H}$ is the $N_t \times N_r$ fading channel matrix (with the same probability distribution as in the Grassmannian system model) and $\mathbf{W}_{tr}$ is the AWGN matrix. κ is a scaling factor to control the transmit power during the training phase relative to the power during the information-bearing phase. Based on $\mathbf{Y}_{tr}$, the channel can be estimated as $\hat{\mathbf{H}} = \mathbf{Y}_{tr}/\kappa$, and the estimate is used for signal detection during the information-bearing phase.

The system model for the information-bearing phase is

$$\mathbf{Y}_X = \mathbf{X}\mathbf{H} + \mathbf{W}_X \quad (4.5)$$

where $\mathbf{X}$ given by [21],

$$\mathbf{X} = \frac{1}{\sqrt{5}} \begin{bmatrix} \alpha(a + b\theta) & \alpha(c + d\theta) \\ \gamma\bar{\alpha}(c + d\bar{\theta}) & \bar{\alpha}(a + b\bar{\theta}) \end{bmatrix}, \quad (4.6)$$

is an $N_t \times N_t$ matrix where $\theta = \frac{1+\sqrt{5}}{2}$ is the Golden number, $\bar{\theta} = 1 - \theta$, $\alpha = 1 + i(1 - \theta)$, and $\bar{\alpha} = 1 + i(1 - \bar{\theta})$. a, b, c , and d are M -QAM symbols, which are normalized to a symbol

Fig. 4.5: 256 point Golden code with (a) set partitioning and (b) Gray labelling.

energy of 1. The total number of points in the Golden code signal space is M^4 . The average signal energy $E_{\mathbf{X}} = \mathbf{E}[||\mathbf{X}||^2] = \sum_i \sum_j \mathbf{E}[|x_{i,j}|^2] = 4$ for any size of the underlying M -QAM constellation. To balance the power during the training and information-bearing phases, we use $\kappa = \sqrt{2}$ for our simulations. Using the channel estimate from training, the receiver uses the likelihood function

$$\Pr\{\mathbf{Y}|\mathbf{X}\} = \frac{1}{(\pi\sigma_W^2)^{N_t N_r}} \exp \left\{ -\frac{||\mathbf{Y}_X - \mathbf{X}\hat{\mathbf{H}}||^2}{\sigma_W^2} \right\} \quad (4.7)$$

for calculating the LLRs used by the polar decoders.

The multilevel polar code is deigned the same way as in Chapter 3, however, for the set partitioning algorithm, the distance metric used is the Euclidean distance. Fig. 4.5 shows the bit error rate per bit for the Golden code using our set partitioning algorithm in comparison with Gray labelling. As the curves show, the set partitioning algorithm is clearly able to separate the bits into distinct reliability levels. By implementing the same system design and methodology described in Chapter 3 to the Golden code, the error rate performance shown in Fig. 4.6 was obtained. The size of the component/ sub-codes used in the simulation was $N' = 256$. The results for the Grassmannian constellation are shown as well for reference.

Fig. 4.6: *Performance of Golden codes and Grassmannian constellations with the multilevel polar coding. Sub-code frame length = 256. Rate = 4/5.*

Although the Grassmannian constellations perform better than the Golden code, we see that our system is still able to produce good results with the latter. This result is supported by the fact that the Grassmannian constellation has a better block and frame error rate performance compared with the Golden code [2]. In other words, because of the multilevel design, constellations with fewer block errors result in fewer errors being propagated down the levels.

4.4 Results With Coherent Golden Codes

The benefit of our set partitioning algorithm and design methodology can be seen in Fig. 4.7 where we show the BER curves for coherent Golden codes. Here, we use a 256 point constellation at a rate of 1/2 and $N = 8192$. The first three curves show the results of a multi-level polar code technique. The first curve is with our set partitioned labels and optimised for a FER of 10^{-2} , the next curve is with our set partitioned labels but with polar

Fig. 4.7: *Polar codes with 256 point coherent Golden codes. Rate = 1/2. N = 8192.*

codes optimised for the AWGN channel, and the third curve is with gray labelling but also optimised for a FER of 10^{-2} . The last curve shows the performance of a BICM polar code system with gray labelling and optimised for a FER of 10^{-2} . We see from the figure that once again, a multi-level polar code system using set partitioned labelling and optimised for a particular FER, outperforms other polar coding and labelling combinations.

Altogether, the figure shows the benefit of designing polar codes for a specific channel and applying the right labelling technique. The design is able to work well for not only Grassmannian or non-coherent systems but for coherent ones as well.

4.5 Complexity Analysis

So far, we have seen that our system is able to produce better error rate performances compared with other coding and mapping systems. An important metric to therefore consider is the receiver complexity, especially compared with the other coded schemes. Precise compar-

isons of relative complexity between different algorithms is very subjective without a standard definition of complexity. For software implementations, the execution speed is a useful metric whereas for hardware implementations, the chip area and power draw may be more relevant. In this thesis we will use the number of floating point operations (flop) required to de-map and decode one message. Although the flop count is a useful and widely used metric, it does not distinguish between the relative costs of performing different operations, (for example, modern Pentium processors can perform additions twice as quickly as multiplications which in turn are 20 times faster than division [22]), which can also vary greatly depending on the type of processor being used and how carefully optimized the algorithm is to fully exploit the operator pipeline. Since these finer details are particularly subjective, in the following we restrict our attention to the flop count. Each complex addition requires two flop and each complex multiplication requires six flop. Logarithmic and exponential operations, which are typically calculated using rational polynomial interpolation, require 20 flop.

4.5.1 De-mapping Complexity Analysis

We first consider the de-mapping complexities of Grassmannian constellations and Golden code signals. As both schemes use the same multilevel polar coding technique, the encoding and decoding complexities, described in the next subsection, are therefore the same.

After the signals $\mathbf{Y}$ are received, the de-mapper works out the maximum likelihood probabilities as follows:

$$\Pr\{\mathbf{Y}|\mathbf{X}\} \propto \begin{cases} \exp\left\{\frac{\|\mathbf{X}^\top \mathbf{Y}\|^2}{\sigma_W^2(1+\sigma_W^2)}\right\} & \text{for the Grassmannian signal} \\ \exp\left\{\frac{\|\mathbf{Y}_X - \mathbf{X}\hat{\mathbf{H}}\|^2}{\sigma_W^2}\right\} & \text{for the Golden code signal} \end{cases} \quad (4.8)$$

The de-mapper calculates these values for every received block (i.e. after T channel uses). For the Grassmannian de-mapper, the value $\mathbf{X}^\top \mathbf{Y}$ is calculated first. Since $\mathbf{X}^\top$ is an $N_t \times T$ matrix and $\mathbf{Y}$ is a $T \times N_r$ matrix, the calculation has $N_t N_r (T - 1)$ complex additions

Table 4.1: De-mapping complexity comparison of Grassmannian constellation and Golden codes.

Method	Number of flop
Grassmannian	$N'M(8N_tN_rT + 2N_tN_r + 20)$
Golden Code	$N'M(8N_t^2N_r + 4N_tN_r + 20) + N'N_tN_r$

and N_tN_rT complex multiplications, for a total of $8N_tN_rT - 2N_tN_r$ flop. Next, the $\|\cdot\|^2$ operation is performed, where $\|\mathbf{A}\|^2 = \sum_{i=1}^{N_r} \sum_{j=1}^{N_t} |a_{i,j}|^2$ with $a_{i,j}$ being the element in row i and column j of $\mathbf{A}$. This requires $3N_tN_r$ flop to calculate all $|a_{i,j}|^2$, and $N_tN_r - 1$ flop to sum all the elements, for a total of $4N_tN_r - 1$ flop. To finally calculate $\Pr\{\mathbf{Y}|\mathbf{X}\}$, one division by the constant $\sigma_W^2(1 + \sigma_W^2)$ and one exponent operation (20 flop) are also needed. Therefore, calculation of $\Pr\{\mathbf{Y}|\mathbf{X}\}$ requires a total of $8N_tN_rT + 2N_tN_r + 20$ flop. This must be repeated M times, for each $\mathbf{X}$ in the constellation. Furthermore, since the transmission of one message word requires the transmission of N' Grassmannian symbols, the flop count must be multiplied by N' . The total flop count for the Grassmannian de-mapper is shown in Table 4.1.

Similarly, for the Golden code de-mapper, the $N_t \times N_r$ matrix, $\hat{\mathbf{H}} = \mathbf{Y}_{tr}/\kappa$ is determined first. This is done once per block by dividing each element in $\mathbf{Y}_{tr}$ by scaling factor κ , for a total of N_tN_r divisions. Next, $\|\mathbf{Y}_X - \mathbf{X}\hat{\mathbf{H}}\|^2$ is calculated for each $\mathbf{X}$ and for each transmission block. The $N_t \times N_t$ matrix, $\mathbf{X}$, is multiplied by the $N_t \times N_r$ matrix, $\hat{\mathbf{H}}$, requiring $N_tN_rN_t$ complex multiplications, and $N_tN_r(N_t - 1)$ complex additions, or $8N_t^2N_r - 2N_tN_r$ flop. Subtracting the result from $\mathbf{Y}_X$ requires an additional $2N_tN_r$ flop, and calculating $\|\cdot\|^2$ needs another $4N_tN_r - 1$ flop. Final calculation of $\Pr\{\mathbf{Y}|\mathbf{X}\}$ requires an additional 21 flop for the exponent and division by σ_W^2 . The total flop count for all M possibilities of $\mathbf{X}$ and for all N' blocks is given in Table 4.1.

As an example using our set-up with $N_t = N_r = 2$, $T = 4$, and using 4096 point constellations and a sub-code length $N' = 512$, we see the comparison of the number of operations required to de-map both a Grassmannian and a Golden code constellation in Table 4.2. As the table shows, the Grassmannian constellation method is significantly more

Table 4.2: *Example of de-mapping complexity comparison of Grassmannian constellation and Golden codes.* $N_t = N_r = 2$, $T = 4$, $M = 4096$, and $N' = 512$.

Method	Number of flop
Grassmannian	327,155,712
Golden Code	209,717,248

computationally intensive than the Golden code method and this is due to the fact that while information bits are transmitted over T timeslots for a Grassmannian constellation, the Golden code only transmits information bits over N_t timeslots. Therefore, while the Grassmannian approach has been shown to produce better error rate performances than its non-coherent Golden code counterpart under block fading conditions, there is a trade-off between complexity and performance.

4.5.2 Bit LLR Calculation Complexity Analysis

The decoders are fed soft symbols (the LLRs), which are calculated from $\Pr\{\mathbf{Y}|\mathbf{X}\}$. As shown in Chapter 2, the LLRs for the polar MLC are calculated as

$$\lambda_l = \ln \frac{\sum_{\mathbf{x} \in \chi_{l,0}} \Pr\{\mathbf{Y}|\mathbf{X}\}}{\sum_{\mathbf{x} \in \chi_{l,1}} \Pr\{\mathbf{Y}|\mathbf{X}\}}, \quad \forall 1 \leq l \leq m. \quad (4.9)$$

We see from (4.9) that for each bit level one, there is one log operation and one divide operation for a total of m log operations and m divide operations per received symbol/block. When $l = 1$, the sets $\chi_{l,1}$ and $\chi_{l,0}$ are each half the size of $\Pr\{\mathbf{Y}|\mathbf{X}\}$ which has M elements. The $\sum$ operator adds these elements in the numerator and denominator, requiring $\frac{M}{2} - 1$ additions each. With $l = 2$, the sets $\chi_{l,k}$ ($k = \{0,1\}$), which depend on the knowledge of the decoded bits of the level $l = 1$, are half the size of the sets $\chi_{1,k}$. Thus the numerator and denominator each have $\frac{M}{4} - 1$ additions. Similarly, the sets $\chi_{3,k}$, which depend on the knowledge of the decoded bits of levels $l = 1$ and $l = 2$, are half the size of the sets $\chi_{2,k}$ and the numerator and denominator each have $\frac{M}{8} - 1$ additions each. When $l = m$, the sets $\chi_{l,k}$ have only one element each and therefore no additions. There are therefore a total of

Table 4.3: **LLR Calculation**, $b = \text{number of de-mapping iterations}$, $m = \log_2 M$.

MLC	$N'(2M - 2 + 19m)$
BICM	$N(M + 19)$
BICM-IDD	$N(M + 19) + (b - 1)N'[(22 + m)M - 2 + 19m]$

$2 \sum_{l=1}^m \left(\frac{M}{2^l} - 1 \right) = 2(M - 1 - m)$ additions per block. Taking into account the division and logarithm needed in (4.9) for each bit increases the flop count per block to $2(M - 1) + 19m$, which must be multiplied by N' to give the count per frame as shown in Table 4.3 .

When BICM is used instead of MLC, knowledge of the previously decoded bits cannot be exploited since the LLRs of all bits are calculated at the same time. Therefore the sets $\chi_{l,k}$ in (4.9) are always of size $M/2$, and the total flop count to calculate the LLRs is $2m(\frac{M}{2} - 1) + 21m = m(M + 19)$ per block.

The use of BICM-IDD, as in [5], further increases the complexity of the LLR calculations. Instead of (4.9) we use

$$\lambda_l = \ln \frac{\sum_{\mathbf{X} \in \chi_{l,0}} \Pr\{\mathbf{Y}|\mathbf{X}\} \mu_X}{\sum_{\mathbf{X} \in \chi_{l,1}} \Pr\{\mathbf{Y}|\mathbf{X}\} \mu_X}, \quad \forall 1 \leq l \leq m. \quad (4.10)$$

where μ_X is the a priori probability that $\mathbf{X}$ was transmitted, based on feedback from the decoder during the previous iteration. Calculation of μ_X for all M values of $\mathbf{X}$ can be performed efficiently using $M - 2$ additions and M exponents, or $21M - 2$ flop. A further M multiplications are needed when multiplying by $\Pr\{\mathbf{Y}|\mathbf{X}\}$, so BICM-IDD requires $(22 + m)M - 2 + 19m$ flop for each block for all but the first iteration, when only $m(M + 19)$ are needed.

4.5.3 Decoding Complexity Analysis

We now consider the total decoding complexity of the polar MLC system. Recall from Fig. 2.4 that the decoding process requires a multi-stage decoding method in which bits from lower levels are decoded, re-encoded, and used to decode bits of the higher levels. Table 4.4 compares the decoding complexity of the polar MLC decoder and that of turbo codes. First,

Table 4.4: *Decoding complexity of turbo and multi-level polar codes.* $n' = \log_2 N'$, $k = \text{constraint length}$, $N = \text{Frame length}$, $i = \text{total number of decoder iterations}$, $d = \text{number of decoders}$, $b = \text{number of de-mapping iterations}$.

Component	Number of Flop
Polar Decoder	$32.5Nn'$
Turbo decoder	$diN(11(2^{k+1}) + 2^{k+3} + 2^{2k+1} + 2^{k+2} + 19)$

we consider the complexity of a single polar decoder. As described in Chapter 2, the polar decoder makes use of two basic node calculations in order to decode received signals. These are

$$\lambda_{i,j} = \begin{cases} \ln \left[\frac{\exp\{\lambda_{i,j+1}\} \exp\{\lambda_{i+2j,j+1}\} + 1}{\exp\{\lambda_{i,j+1}\} + \exp\{\lambda_{i+2j,j+1}\}} \right], & \text{if } \left\lfloor \frac{i-1}{2^j} \right\rfloor \equiv 0 \pmod{2} \\ \lambda_{i,j+1} + (1 - 2\hat{v}_{i-2^j,j})\lambda_{i-2^j,j+1}, & \text{otherwise} \end{cases} \quad (4.11)$$

for $1 \leq i \leq N'$ and $0 \leq j \leq n'$. The upper node calculation for $\lambda_{i,j}$ requires two exponents, two additions, one multiplication, one division, and one logarithm, or 64 flop. The lower node calculation needs either one addition or one subtraction, depending on $\hat{v}_{i-2^j,j}$. For each polar decoder in the MLC, the upper node calculations occur in each column $j < n'$ and in half the rows of i where $\left\lfloor \frac{i}{j-1} \right\rfloor \equiv 0 \pmod{2}$, while the lower node calculations also occur in each column and in the other half of the rows. Therefore, the lower and upper node values of $\lambda_{i,j}$ are each calculated a total of $\frac{N'}{2}n'$ times per decoder. Recall that there are m decoders in the MLC system.

In [23], the complexity of a turbo decoder per bit was determined. For each code bit, each decoder requires 2^k exponents, one log, $2^{k+3} + 2^{2k}$ multiplications, $2^{k+2} + 2^{2k} - 2$ additions, and $2^{k+1} + 1$ divisions for a total of $11(2^{k+1}) + 2^{k+3} + 2^{2k+1} + 2^{k+2} + 19$ flop where k is the constraint length of the turbo encoders. To obtain the number of flop for the entire decoding procedure, this must be multiplied by the frame length N , the number of decoders used d and the total number of decoder iterations performed a .

As an example, with a sub-code length of $N' = 512$ and a Grassmannian constellation with $M = 4096$ points, $m = \log_2 4096 = 12$ and $N = 12 \times 512 = 6144$ bits, Table 4.5 shows the number of operations needed for decoding polar and turbo codes per received frame. The

Table 4.5: *Decoding complexity example of turbo and multi-level polar codes.* $n' = 9$, $k = 5$, $N = 6144$, $i = 32$, $d = 2$, $b = 4$.

Component	Number of Flop
Polar Decoder	1,797,120
Turbo decoder	1,240,596,480

Table 4.6: *Comparison of the total number of receiver operations for Grassmannian and Golden codes.* $n' = \log_2 N'$.

Method	Number of flop
Grassmannian	$N'M(8N_tN_rT + 2N_tN_r + 20) + 32.5Nn' + N'(2M - 2 + 19m)$
Golden Code	$N'M(8N_t^2N_r + 4N_tN_r + 20) + N'(N_tN_r + 2M - 2 + 19m) + 34Nn'$

turbo decoder implemented in [5] uses $d = 2$ MAP decoders with $b = 4$ de-mapping/decoding iterations (the LLRs are calculated 4 times) and $a = 8$ BCJR based turbo iterations within each de-mapping/decoding iteration for a total of $i = 32$ iterations each, and $k = 5$. We see from Table 4.5 that the turbo decoder is orders of magnitude more complex than the polar MLC decoders.

As for the LDPC decoder, it is difficult to estimate the number of iterations needed for decoding a frame. However, because the LDPC system simulated in this thesis uses BICM, from Table 4.3, the LLR calculations require 25,282,560 flop using our example. In comparison, the polar MLC requires 6,107,136 flop for decoding and LLR calculations. The LLR calculations for the BICM LDPC system alone require about 4 times more flop. It is of note however, that the SC polar decoder is a serial process and cannot be parallelized like the decoding process for LDPC codes and this affects the latency. However, even with this taken into account, the overall latency for the polar code system will still be less than that of the LDPC codes. Therefore, we see that the polar decoder system is significantly less complex than both the LDPC and the turbo systems while still providing superior error rate performance.

Putting all these results together, we can obtain the expressions for the total receiver complexity of the polar MLC system for both the Grassmannian and the Golden code constellations shown in Table 4.6. Continuing with the previous example, with a constellation

Table 4.7: Example of the total number of receiver operations for Grassmannian and Golden codes.

Method	Number of flop
Grassmannian	333,207,552
Golden Code	215,907,328

size of 4096 and a sub-code length of 512, the total number of flop required at the receiver is shown in Table 4.7. This implies that the Grassmannian requires about 1.5 times the number of flop as the Golden code. We therefore see that even though there is a complexity-performance trade-off between the two constellations, the polar MLC receiver is still overall less complex than the turbo decoder while providing better performance.

Chapter 5

Conclusion

A novel methodology for designing polar codes that work effectively with Grassmannian constellations was proposed in this thesis. This combination, based on multi-level polar coding, was effectuated by the help of our novel set partitioning algorithm. This algorithm is a generalised one that works both for regular and irregular constellations. It was shown that this algorithm was able to partition both non-coherent (Grassmannian) and coherent (Golden codes) multidimensional constellations. In addition, a different way of designing polar codes which involves finding the codes that achieve a target FER at the lowest possible design SNR was proposed. This is in contrast to the previous design methodologies which seek to minimize the FER at one design SNR, which tends to produce codes that perform poorly at other SNRs. By combining these two propositions, the error rate curves showed that this novel system came the closest to the non-coherent MIMO capacity compared with the other state-of-the-art coding techniques. In addition, this novel method is significantly less computationally intensive than the other BICM based coding methods.

There are several ways in which this work may be expanded upon in the future, and these include:

- The polar decoder used in this system was the suboptimal successive cancellation decoder. Other more powerful, albeit more complex, algorithms such as the successive

cancellation list (SCL) and the adaptive cyclic redundancy check-aided SCL (aCA-SCL) may be used to further improve the performance of this system [8].

- In order to reduce the complexity of the Grassmannian mapper, a list-based de-mapper can be used with little effect on the performance.
- Larger Grassmannian constellations with different values of T can be explored.

References

- [1] L. Zheng and D. N. C. Tse, “Communication on the Grassmann manifold: A geometric approach to the noncoherent multiple-antenna channel,” *IEEE Trans. Inf. Theory*, vol. 48, no. 2, pp. 359–383, February 2002.
- [2] R. Gohary and T. Davidson, “Non-coherent MIMO communication: Grassmannian constellations and efficient detection,” *IEEE Trans. Inf. Theory*, vol. 55, no. 3, pp. 1176–1205, March 2009.
- [3] S. Roger, J. Cabrejas, D. Calabuig, J. F. Monserrat, Y. Fouad, R. H. Gohary, and H. Yanikomeroglu, “Non-coherent MIMO communication for the 5th generation mobile: Overview and practical aspects,” *Waves Journal*, no. 6, pp. 5–13, 2014.
- [4] R. W. Hamming, “Error detecting and error correcting codes,” *Bell System Technical Journal*, vol. 29, no. 2, pp. 147–160, April 1950.
- [5] M. El-Azizy, R. Gohary, and T. Davidson, “A BICM-IDD scheme for non-coherent MIMO communication,” *IEEE Trans. Wireless Commun*, vol. 8, no. 2, pp. 541–546, Feb 2009.
- [6] G. Colman, R. Gohary, M. El-Azizy, T. Willink, and T. Davidson, “Quasi-Gray labelling for Grassmannian constellations,” *IEEE Trans. Wireless Commun*, vol. 10, no. 2, pp. 626–636, Feb 2011.

- [7] E. Arikan, “Channel polarization: A method for constructing capacity-achieving codes,” in *Proc. IEEE International Symposium on Information Theory*, Toronto, Canada, July 2008, pp. 1173–1177.
- [8] K. Nui, K. Chen, J. Lin, and Q. Zhang, “Polar codes: Primary concepts and practical decoding algorithms,” *IEEE Communications Magazine*, vol. 52, pp. 192–203, July 2014.
- [9] M. Seidl, A. Schenk, C. Stierstorfer, and J. B. Huber, “Aspects of polar-coded modulation,” in *Proc. International ITG Conference on Systems, Communication and Coding*, Munich, Germany, January 2013, pp. 1–6.
- [10] H. Imai and S. Hirakawa, “A new multilevel coding method using error-correcting codes,” *IEEE Transactions on Information Theory*, vol. 23, no. 3, pp. 371–377, May 1977.
- [11] E. Zehavi, “8-PSK trellis codes for a Rayleigh channel,” *IEEE Transactions on Communication*, vol. 40, no. 5, pp. 873–884, May 1992.
- [12] U. Wachsmann, R. F. H. Fischer, and J. B. Huber, “Multilevel codes: Theoretical concepts and practical design rules,” *IEEE Transactions on Information Theory*, vol. 45, no. 5, pp. 1361–1391, July 1999.
- [13] E. Sasoglu, E. Telatar, and E. Arikan, “Polarization for arbitrary discrete memoryless channels,” in *Proc. Information Theory Workshop*, Taormina, Italy, October 2009, pp. 144–148.
- [14] H. Saber and I. Marsland, “An incremental redundancy hybrid ARQ scheme via puncturing and extending of polar codes,” *IEEE Transactions on Communications*, vol. 63, no. 11, pp. 3964–3973, November 2015.
- [15] L. Szczecinski and A. Alvarado, *Bit-Interleaved Coded Modulation: Fundamentals, Analysis and Design*. Wiley, February 2015.

- [16] M. Seidl, A. Schenk, C. Stierstorfer, and J. Huber, “Multilevel polar-coded modulation,” in *Proc. IEEE International Symposium on Information Theory*, Istanbul, Turkey, July 2013, pp. 1302–1306.
- [17] G. Ungerboeck, “Trellis-coded modulation with redundant signal sets part I: Introduction,” *IEEE Communications Magazine*, vol. 25, pp. 5–11, February 1987.
- [18] G. D. Forney, “Coset codes – Part I: Introduction and geometrical classification,” *IEEE Transactions on Information Theory*, vol. 34, no. 5, pp. 1123–1151, September 1988.
- [19] G. T. Toussaint and B. K. Bhattacharya, “Optimal algorithms for computing the minimum distance between two finite planar sets,” *Pattern Recognition Letters*, vol. 2, no. 2, pp. 79–82, 1981.
- [20] H. Vangala, E. Viterbo, and Y. Hong, “A comparative study of polar code constructions for the awgn channel,” January 2015, paper under review.
- [21] J.-C. Belfiore, G. Rekaya, and E. Viterbo, “Golden code: A 2×2 full-rate space-time code with non-vanishing determinants,” *IEEE Trans. Inf. Theory*, vol. 51, pp. 1432–1436, April 2005.
- [22] A. Fog. (2016, January) Instruction tables. Technical University of Denmark. [Online]. Accessed: 14-January-2016. http://www.agner.org/optimize/instruction_tables.pdf.
- [23] L. Hebbes, R. R. Malyan, and P. Bali, “Computational complexities and the relative performance of turbo codes,” in *Proc. EUROCON*, vol. 1, Bratislava, Slovakia, July 2001, pp. 37–40.