Physical Layer Network Coding based on Compute-and-Forward

Abigail Elcock
Doctor of Philosophy
University of York
Electronic Engineering

February 7, 2022

Abstract

In this thesis, Compute-and-Forward is considered, where the system model consists of multiple users and a single base station. Compute-and-Forward is a type of lattice network coding which is deemed to avoid backhaul load and is therefore an important aspect of modern wireless communications networks. Initially we propose an implementation of construction D into Compute-and-Forward and investigate the implementation of multilayer lattice encoding and decoding strategies. Here we show that adopting a construction D lattice we can implement a practical lattice decoder in Compute-and-Forward. During this investigation and implementation of multilayer lattice encoding and decoding we discover an error floor due to an interaction between code layers in the multilayer decoder. We analyse and describe this interaction with mathematical expressions and give detail using lemmas and proofs. Secondly, we demonstrate the BER performance of the system model for unit valued channels, integer valued channels and complex integer valued channels. We show that using the derived expressions for interaction that the decoders on each code layer are able to indeed decode. The BER results are demonstrated for two scenarios using zero order and second order Reed-Muller codes and first and third order Reed-Muller codes. Finally, we extend our system model using construction D and existing conventional decoders to include coefficient selection algorithms. We employ an exhaustive search algorithm and analyse the throughput performance of the codes. Again, we extend this to both our models. With the throughput of the codes we see that each layer can be successfully decoded considering the interaction expressions. The purpose of the performance results is to show decodability with the extension of using differing codes.

Contents

Acknowledgements

I would like to sincerely thank my supervisor Prof. Alister Burr for his supervision, guidance and support on my research and subsequent development.

I would like to thank colleagues in the communications research group for their support and discussions throughout my project.

Finally, I would like to thank my Husband and parents for their unwavering love and encouragement in times of need, especially throughout the pandemic.

Authors Declaration

I declare that this thesis is a presentation of original work and I am the sole author. This work has not previously been presented for an award at this, or any other, University. All sources are acknowledged as References.

Glossary

Physical-layer Network Coding (PNC)
Lattice Network Coding (LNC)
Access Point (AP)
Central Processing Unit (CPU)
Network Coding (NC)
Multiple Access Channel (MAC)
Broad Cast (BC)
Log Likelihood Ratio (LLR)
Elementary Divisor Construction (EDC)
Principle Ideal Domain (PID)
Construction by Code Formula (CCF)
Reed Muller (RM)
Branch Metric (BM)
Bahl Cocke Jelinek Raviv (BCJR)
Signal to Noise Ratio (SNR)
Additive White Gaussian Noise (AWGN)
Bit Error Rate (BER)
Fast Walsh Hadamard Transform (FWHT)
Soft Decision Decoder (SDD)
Lenstra Lenstra Lovasz (LLL)
Frame Error Rate (FER)
Throughput (TP)
Quadrature Amplitude Modulation (QAM)
Multiple-Input Multiple-Output (MIMO)

List of symbols

B	Number of Base Stations
J	Number of sources
L	Number of layers
$\mathbf{w}$	Dataword
$\mathbf{x}$	Lattice codeword
$\mathbf{c}$	Codeword
$\hat{\mathbf{c}}_l$	Estimated codeword on layer l
$\mathbf{y}$	Received signal
$\mathbf{y}^{'}$	Scaled received signal
h_{bj}	Channel coefficient for source j and base station b
a_{bj}	Integer coefficient for source j and base station b
α	Scalar - moves channel coefficients close to integer coefficient
$\mathbf{s}_{l'}$	Sum component of decoded codeword
$\mathbf{k}_{l'-1}$	Carry component of decoded codeword
$\mathbf{z}$	Noise
$\mathbb{R}$	Reals
$\mathbb{Z}$	Integers
$\mathbb{F}$	Finite field
$\Re$	Real part
$\Im$	Imaginary part
I	In-phase
Q	Quadrature

G	Generator matrix
mod	Modulo operation
d_{min}	Minimum Hamming distance
$\cdot$	AND operation
$\oplus$	XOR operation

List of Figures

Chapter 1

Introduction

1.1 Overview

Managing user capacity and interference on wireless communication networks are challenges to the design of wireless communication systems. Interference can be caused by other users on the network where users have some influence on one another due to the nature of the wireless network. This means that as numbers of users increases interference on the network also increases. Noise and signals within the area also have an influence on the network and therefore tackling this issue is still a key topic. The demand for capacity has followed an exponentially increasing curve and continues in this way meaning the need for algorithms to handle this interference whilst maintaining decoding performance is prominent. Interference is generally inevitable in wireless environments and many algorithms attempt to avoid this by removing the interference signals prior to decoding. These techniques employ a significant layer of complexity as avoidance of interference affects performance rates as the network size increases. Thus, interference handling with large capacity will remain a significant challenge for wireless communication systems. Compute and Forward (C&F) has been introduced as a strategy that exploits interference in order to avoid diminishing rates between users on a network. An advantage of this strategy is that the Access Points (APs) decode linear functions of transmitted messages and not the messages themselves. Furthermore, in order to do so, C&F utilises the use of lattice codes which, due to the algebraic structure, ensures that the integer combination of codewords can be decoded. Lattice Network Coding (LNC) is a technique implemented in C&F to avoid the large backhaul load of communications systems. Commonly, methods that implement LNC employ nested lattice codes where the structure of the lattice ensures the linear combination of codewords is also a codeword itself. Each AP then transmits this information to the destination for decoding of the messages. Disadvantages of the C&F scheme become apparent from the use of Construction A lattice, as introduced in [2]. Construction A is complex to decode and requires non-binary codes where this complexity is stated in [3]. For practical use in wireless communications systems where code lengths are typically large the construction requires a large field size and therefore encoding and in particular decoding becomes extremely complex. Construction D allows the use of binary codes, with a layered decoder that can use conventional (soft decision) binary decoders. This work further investigates the use of construction D and implementation of multilayer decoders.

1.2 Objectives

The work in this thesis will consider the challenges relevant to contributing to a more practical implementation of Compute-and-Forward. Potential significant challenges considered for this objective involve:

- C&F research thus far has not particularly concentrated on the practicalities for implementation in industry. C&F was originally introduced in [2] with the implementation of lattice Construction A leading to complexities for decoding. The work of [4] highlights that a construction D lattice can be generated using Construction by Code Formula under certain conditions. Fortunately, this construction method is less complex than construction A however implementation and design of the decoder is key.

- A significant challenge will be implementation of lattice decoder dependent on the chosen codes. Implementation of a construction D lattice may be practical however multilayer decoding may require differing decoders that are non-conventional in the sense that they are not well known or established techniques in telecommunications systems.

- Another significant consideration of the C&F technique is coefficient selection. Coefficient selection has a significant impact of the computation rate therefore the challenge would be to implement an algorithm with low complexity within our practical model. Much research has identified the complexity of coefficient selection and progressed to produce algorithms with low complexity [5].

1.3 Aims

The aim of this work is to overcome challenges of implementing a practical compute-and-forward system using multilayer lattice constructions and existing multilayer decoding strategies employed within wireless communications infrastructure. The aims are summarised as follows:

- Choose practical codes, such as binary codes in order to derive a practical basis model for C&F
- Introduce existing or currently employed encoding and in particular decoding algorithms that can be readily adapted in telecommunications leading to an implementable C&F
- Solve the complexity problem of using lattice construction A by replacing this with a low complexity method that generates lattice construction D
- Investigate multilayer lattice decoding algorithms and the effect of this on decoding nested lattice codes
- Present low complexity coefficient selection algorithm within the C&F model for decoding random channels
- Extend the practical C&F system model to handle complex signals and complex valued channels
- Analyse the suited topology of the system and recovery of data at CPU

In this thesis we investigate the practical requirements for a C&F scheme, namely, the use of lattice construction A and subsequently the chosen codes and the design of the decoder. The overall objective for this work is to move closer to implementable C&F.

In order to move away from lattice construction A we further research lattice construction D and the multilayer decoder. Through our investigations into implementing lattice construction D we find that there is an interaction between layers of the multilayer decoder which causes irreducible errors (an error floor) if not properly taken into account where we show that this derives from a "carry" term between the layers. We derive mathematical expressions for the interaction and also determine additional criteria for the layer codes that avoids the irreducible errors. We evaluate performance with codes that fulfil these criteria. We extend our work to employ coefficient selection algorithm and apply the method to fading channels.

1.4 Contributions

Based on the discussions above, the main novel contributions of this thesis are listed below:

- Implementation of construction D lattice into C&F based on binary codes which are increasingly more practical for industry.
- Design of a multilayer decoder for Reed-Muller and Extended Hamming codes within our practical system model, showing good performance results
- For the unit valued channel coefficient, derivation of interaction signals between code layers have been identified during multilayer decoding. Full derivation and nature of the signals are produced along with proofs and general expressions for multiple layers.
- The conditions at which successful decoding can be achieved is also derived. This is produced in the form of criteria that should be met such that successful decoding can be achieved.
- For the real and complex valued case, we expand the derivation and highlight the significant increased complexities of decoding code layers due to this interaction. We again detail the mathematical expressions for the general case of decoding multiple layers. Theoretical analysis shows that the theoretical complexity of these signals increases significantly with increasing number of layers however the decodability of the overall system does not due to implementation of the derived expressions and operation of C&F. Numerical results show that each layer can be successfully decoded.
- We show that we can decode and recover the individual codewords and datawords at the hub from a two source system model. This is detailed for unit valued channels, integer valued channels and complex integer valued channels. We also show this for real and complex signals.

- We investigate [5] low complexity coefficient selection algorithms and embed into our model and show that random channels also achieve successful decodability in our practical C&F system model.

Within this work we use plain letter, boldface lowercase letters and boldface uppercase letters to denote scalars, vectors and matrices respectively. We denote the Gaussian integers as $\mathbb{Z}[\mathbb{i}]$, integers as $\mathbb{Z}$ and Eisenstein integers as $\mathbb{Z}[\omega]$. The fields are denoted as $\mathbb{F}_p$ where p denotes the field size.

Chapter 2

Background

2.1 Introduction

In this chapter, we cover the theory and fundamental techniques drawn upon throughout the thesis. Initially we introduce the principles of Physical Layer Network Coding, a technique which avoids interference by decoding combinations of data messages, as mentioned in chapter 1. We then introduce a summary and derivation of the physical layer network coding technique Compute-and-Forward. We discuss lattice network coding and present lattice constructions and their methods. We also describe existing decoding algorithms researched for this work and present preliminaries in abstract algebra necessary for this work. Furthermore, we cover multilayer coding schemes developed for compute-and-forward.

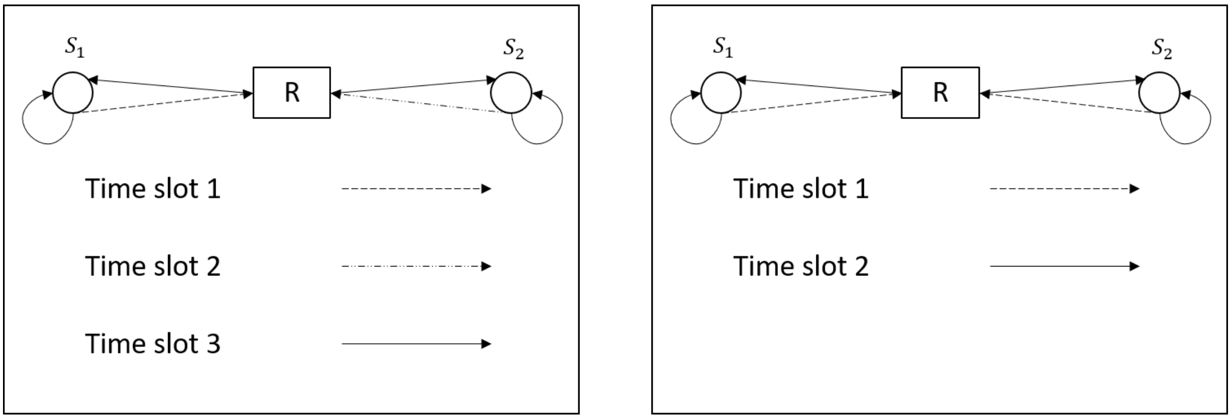
Research such as Soft Decision Decoders and Reed-Muller codes highlighted within this chapter link directly to wireless communications due to their already prominent use. Their use is only possible if a different lattice construction that allows binary codes is implemented, therefore recent research is shown in this chapter. The use of such decoders and Reed-Muller codes in this work also means that implementation into a wireless communications network is straight forward, showing that the contributions of this thesis is advantageous to moving Compute-and-Forward closer to practical implementation.

2.2 Physical Layer Network Coding

Physical Layer Network Coding (PNC) is a technique used to reduce backhaul load in order to achieve unambiguous transmission. The principle, originally introduced in [6], is to exploit the natural combinations of signals transmitted over the network.

2.2.1 Principles of PNC

In this Section we detail examples of PNC and Network Coding (NC) using the simple two-way relay channel model. We define S_1 and S_2 as source 1 and source 2 respectively and denote the base station in Figures 2.1 (a) and (b) as R. Network coding was originally introduced in [7] to increase network throughput. Within this technique base stations transmit functions of data packets instead of individual data packets therefore reducing the number of transmissions. Figure 2.1 (a) shows the operation of NC where the base station transmits the exclusive or (XOR) of the signals $\mathbf{x}_1$ and $\mathbf{x}_2$ in the third time slot. $\mathbf{x}_2$ can be recovered by source 1 by performing $\mathbf{x}_1 \oplus \mathbf{x}_1 \oplus \mathbf{x}_2$ using known information $\mathbf{x}_1$. Similarly source 2 can recover $\mathbf{x}_1$ using the same method and known information $\mathbf{x}_2$.



(a) NC.

(b) PNC.

Figure 2.1: PNC & NC transmission schemes in a two way relay channel

PNC is different to NC, shown in Figure 2.1 (b), as this technique comprises two rather than three separate stages, the multiple access channel (MAC) stage and broadcast (BC) stage. During the MAC phase all users transmit their signals simultaneously to the base station where the base station maps the signals to network coded symbols ready for the BC stage. The network coded symbols can take many forms however the XOR operation is a simple example of a linear mapping strategy.

2.3 Lattice Constructions

Compute-and-Forward is underpinned by the lattice construction, thus we introduce some essential construction methods and definitions used within this research.

N-Dimensional Lattice

The general expression to generate a lattice is:

$$\Lambda = \{\mathrm{G}_\Lambda \mathbf{d} : \mathbf{d} \in \mathbb{Z}^k\} \tag{2.1}$$

where $\mathrm{G}_\Lambda = [\mathbf{g}_1, \mathbf{g}_2, \ldots, \mathbf{g}_K]$ is the generator matrix for Λ containing basis vectors $\mathrm{g} \in \mathbb{R}^N$ of $\mathbf{G}_\Lambda$ and $\mathbf{d}$ defines the lattice vector. An N-dimensional real lattice is a finite subset of N-space $\mathbb{R}^{\mathrm{N}}$. A 1-dimensional lattice can be generated by applying a scalar α to the integers $\mathbb{Z}$, given as: $\Lambda = \{\alpha\mathbb{Z} : \alpha \in \mathbb{R}\}$ [8] [9].

Definition - Nested Lattice

A lattice $\Lambda^{'}$ is defined as being nested in another lattice Λ if $\Lambda^{'}$ is a sub-lattice of Λ. This is termed $\Lambda^{'}$ in Λ, where Λ is the fine lattice and $\Lambda^{'}$ the coarse lattice. The number of nested lattices is not limited as we can define a series of j lattices, $\Lambda_1, \Lambda_2, \ldots, \Lambda_j$. These lattices are defined as nested if $\Lambda_1 \subseteq \Lambda_2 \subseteq \ldots \subseteq \Lambda_j$. A quotient ring $\Lambda/\Lambda^{'}$ is defined as the lattice partition [8]. An example of a quotient ring is when a ring E is in the integers $\mathbb{Z}$ and an ideal of E, e, denoted as E/e is $2\mathbb{Z}$, meaning the quotient ring is $\mathbb{Z}_2 = \mathbb{Z}/2\mathbb{Z}$ [10] [11]. The ideal is defined as a special subset of the Ring's elements.

Definition - Voronoi Region

The Voronoi region or shaping region V of a lattice is the set of all points in the reals that are closest to the zero vector of the lattice. $V = \{\lambda : Q_\Lambda(\lambda) = \mathbf{0}\}$ where λ is the lattice points. Let $\mathrm{Vol}(V)$ denote the volume of V [9].

Definition - Nested Lattice Codes

A nested lattice code C is the set of all points of a fine lattice Λ that are within the fundamental Voronoi region V of a coarse lattice $\Lambda^{'}$, the points within this region are known as coset leaders [9] [8]. These set of points are expressed as:

$$C = \Lambda \ \mathrm{mod}\Lambda^{'} = \{\lambda \mathrm{mod}\Lambda^{'}, \lambda \in \Lambda\} \tag{2.2}$$

where mod denotes the modulo operation and when applied to a lattice is defined as: $\mathrm{xmod}\Lambda^{'} = \mathrm{x} - \mathcal{Q}_{\Lambda}(\mathrm{x})$ and $\mathcal{Q}$ is the quantiser defined in equation (2.13) in subsection 2.4.3. The modulo operation over $\Lambda^{'}$ represents the modulo operation taken over the lattice in the N-dimensional space. We can present the modulo operation for any scalar, say p, where the identity is $(p\mathrm{mod}\omega)\mathrm{mod}\omega = p\mathrm{mod}\omega$ [8]. Figure 2.2 shows an example of a fine and coarse lattice and thus the lattice partition. The fine lattice is represented by the blue dots and the coarse lattice represented by the red circles. The Voronoi region of each lattice can be identified and the shaping region $\Lambda/\Lambda^{'}$ shows that the fine lattice Λ is divided into 4 cosets due to the coarse lattice $\Lambda^{'}$. This is because when lattice points lie on the negative boundary in each dimension they are assigned to the region below leaving

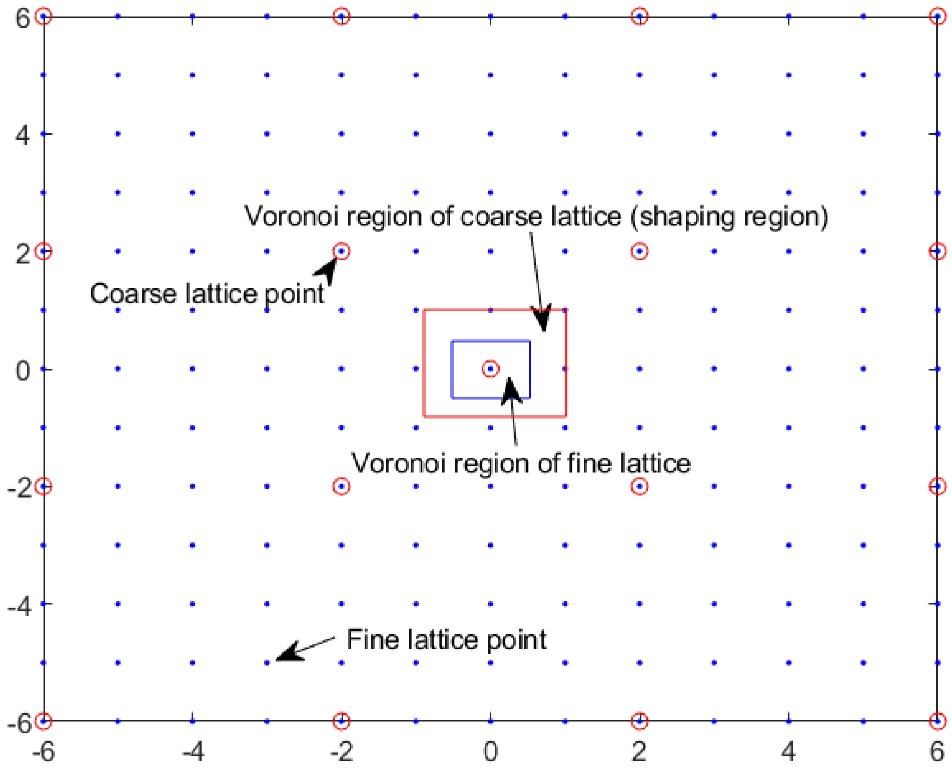

Voronoi region of coarse lattice (shaping region)
Coarse lattice point
Voronoi region of fine lattice
Fine lattice point
6
4
2
0
-2
-4
-6
-6
-4
-2
0
2
4
6

Figure 2.2: Example of fine and coarse lattice

$(0 + 0\mathbb{i}), (0 + \mathbb{i}), (1 + 0\mathbb{i})$ and $(1 + \mathbb{i})$.

2.3.1 Lattice Construction A

Let $C(k, N)$ be a linear code of length N and p be a prime number in $\mathbb{F}$ where the symbols of C are in $\mathbb{F}_p$ [12]. The codebook can be expressed as:

$$C = \{\mathrm{G}\mathbf{x} : \mathbf{x} \in \mathbb{F}_p^{\mathrm{N}}\} \tag{2.3}$$

where $\mathbf{x}$ is the encoded message vector and G the generator matrix. The lattice Λ is formed by projecting the codebook into the reals $\mathbb{R}^N$ using a map, $\mathcal{E}$. For the usual case where $\mathbb{F}_p$ consists of the integers 0 to $p - 1$, then $\mathcal{E}$ just means converting the integer value to the corresponding real value. If G is the generator matrix of C and letting I be the NxN identity matrix, then:

$$\mathrm{G}_\Lambda = \begin{bmatrix} \mathrm{G} \\ p\mathrm{I} \end{bmatrix} \tag{2.4}$$

As shown the resulting lattice is a shifted form through multiples of p. This result generates the points at the fundamental region of Λ and therefore shifted duplications of C can be generated and distributed over the whole message space as shown in Figure 2.3. This means that a copy of the points generated at the fundamental region are placed at every integer vector within the message space. Construction A consists of the lattice tessellation of the square between 0 and $p - 1$ on each axis. Figure 2.3 shows the shifted duplications at every integer of the lattice in the field $\mathbb{F}_2$, $p = 2$. The choice of p within this construction is important as the minimum euclidean distance of the lattice is p, meaning that in order to achieve a large Euclidean distance p must be large. As described, this construction method requires a large field size to be employed in order to generate code lengths deemed practical for industrial communications networks. For codes of a larger field size the complexity becomes apparent, particularly in the decoder. For example, if we choose $\mathbb{F}_{13}$, there is the probability of 13 symbol values which in turn generates at least 12 Log Likelihood Ratios (LLRs), clearly increasing complexity compared to that of a binary decoder. Clearly, for a binary decoder there are only 2 possible symbols, $\{0, 1\}$ and 1 LLR. For this reason we focus our attention to Construction D lattices which enable the use of binary codes and decoders.

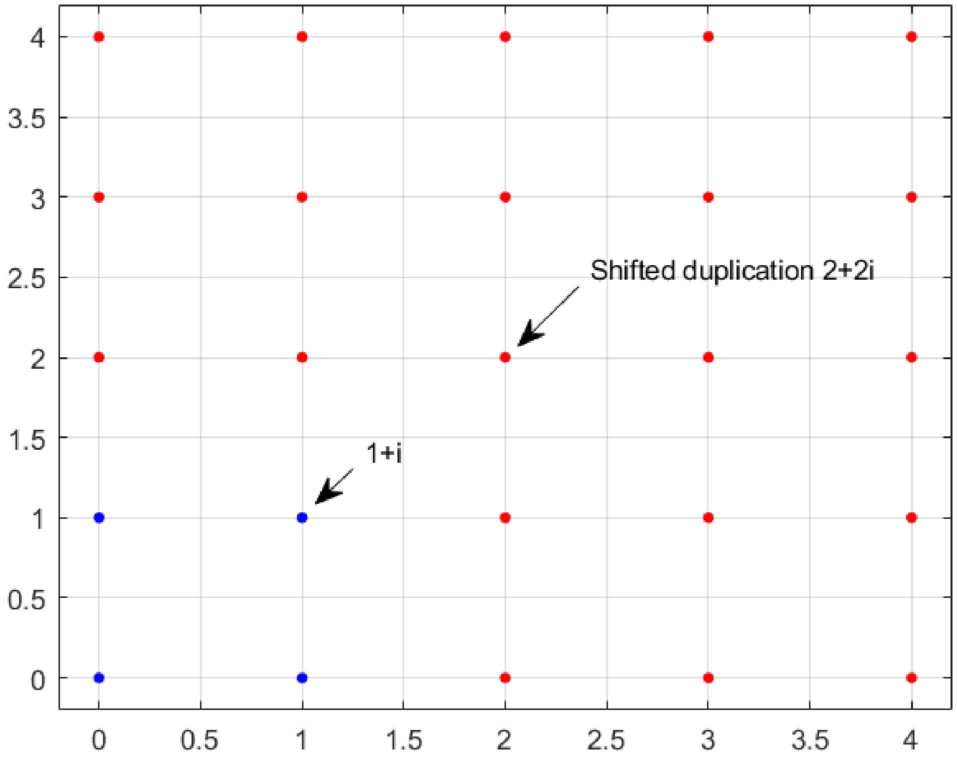


Figure 2.3: Example of tessellation in message space

2.3.2 Lattice Construction D

Lattice construction D, as described in [12], generates nested lattices. From [12], let $\mathbb{F}_2^N \supset C_{L-2} \supseteq C_{L-1} \ldots \supseteq C_0$ be a set of binary linear codes, with C_l as a $[k_l, N, d_l]$ linear block code. Basis vectors $\mathbf{g_1}, \ldots, \mathbf{g}_{k_l}$ are chosen in $\mathbb{F}_2^N$ which form the generator matrix G_l. The matrix multiplication between W_l containing message vectors and generator matrix G_l is performed to produce the binary linear codes. $\mathbf{g}_1, \ldots, \mathbf{g}_{k_l}$ are chosen such that they span C_l for $l = 0, \ldots, L-1$. A map, $\mathcal{E}_l$ then projects the finite field $\mathbb{F}_2^N$ to the reals $\mathbb{R}^N$, where $\mathcal{E}_l(x) = x \times 2^{L-l}$, $x \in C_l$. As shown in [12], the formed lattice Λ contains all vectors of the form:

$$\Lambda = \sum_{l=0}^{L-1} \sum_{j=1}^{k_l} \alpha_j^{(l)} \mathcal{E}_l(\mathbf{g}_j) + (2\mathbb{Z})^N \tag{2.5}$$

where $\alpha_j^{(l)} \in \{0, 1\}$. The resulting lattice, provided the minimum Hamming distance is $d_l \geq \frac{4^l}{\gamma}$ and $\gamma = 2$, Λ is an N-dimensional lattice with hypercubic shaping region. A group of binary codes that could adhere to these constraints would be Reed-Muller codes as they consist of varying orders and code types.

The method of multilayer decoding with construction D relies on the assumptions that the previous layer(s) has been decoded correctly and by subtracting these code estimates, previous layers can be discarded cancelling out any interference. For example at layer $l^{'}$ we subtract the previous layer estimate of the codeword $\hat{\mathbf{c}}_{l^{'}-1}$ from the previous layer scaled received signal $\mathbf{y}^{'}_{l^{'}-1}$ and divide by 2 to form $\mathbf{y}^{'}_{l^{'}}$, which we then take modulo Λ and decode to find $\hat{\mathbf{c}}_{l^{'}}$. Modulo Λ operation is implemented at each layer such that posterior layers are discarded. This means that next layers do not contribute any interference to the current layer. The operation performed by the decoder at each layer is:

$$\hat{\mathbf{c}}_l = \underset{\Lambda/\Lambda^{'}}{\text{argmin}} \sum_{l=1}^{L} \Big(\frac{1}{2^{l-1}}\mathbf{y}^{(l)} - \hat{\mathbf{c}}_{l-1}\Big)\text{mod}\Lambda \tag{2.6}$$

Lifted Construction D

The work of [1] extends construction D to the complex case, introducing ‘Lifted Construction D’. Lifted Construction D is given by $\Lambda = \Lambda_D + \mathbb{i}\Lambda_D$ where Λ_D is a construction D lattice. For a nested $\mathbb{Z}[\mathbb{i}]$ lattice, fine lattice Λ and coarse lattice $\Lambda^{'}$ lifting achieves $\Lambda_D + \mathbb{i}\Lambda_D$ and $\Lambda^{'}_D + \mathbb{i}\Lambda^{'}_D$ respectively. The encoding and decoding process shown in [1] is effectively split into two separate entities representing the complex components, In-phase and Quadrature, $\mathbf{w}^I \in W_r$ and $\mathbf{w}^Q \in W_I$, where $W = W_r + W_I$. The separate but identical encoders then use a map $\mathcal{E} : W \to \mathbb{R}^N$ to project the message to the reals producing the complete complex coded symbol output $\mathbf{x} = \mathbf{x}^I + \mathbb{i}\mathbf{x}^Q$. Furthermore, as nested codes are applied, the encoders are split into layers. Let $L = 1, 2, \ldots, l$ with nested codes $C_1 \subseteq C_2, \ldots, \subseteq C_l$ where layer l is encoded according to nested code C_l. A similar principle is employed for the decoders, using multilayer decoding with I and Q split into separate identical entities.

2.3.3 Construction π_A and Construction π_D

Construction π_A

Construction π_A extends construction A to generate multilayer codes over rings. Within this construction each code is generated using a prime sized finite field where it is necessary for each layer to use a different field size. The main difference between Construction π_A and construction A is multilayer coding and due to this the field size on each layer is smaller than that in construction A. A reduction in field size has an effect on decoding complexity as for each code layer the decoder has a smaller field size to decode. The complexity of decoding a larger sized field is briefly discussed in subsection 2.3.1. Complexity of construction A however is determined by the size of the field $\mathbb{F}_p$ which is usually very large. Construction D however is more straight forward to decode due to the use of $\mathbb{F}_2$. The authors of [13] expand their work on construction π_A to reduce this complexity and proposed a new method, Construction π_D.

Construction π_D (Elementary Divisor Construction)

Construction π_D, also known as Elementary Divisor Construction (EDC), introduced in [14] can be used to generate multilayer lattices. This construction is a generalisation of the construction π_A lattice construction which as described in [14] is a special case. The generalisation provides more scope to the choice of codes used on the code layers. The construction generates some distinct primes $p_1, p_2, \ldots, p_l$ and corresponding linear codes $C_0, C_1, \ldots, C_l$. In order to generate more code flexibility, the same field size is permitted on multiple code layers if the code layers are nested and that the modulo sum is applied to the codes. This effect solves the problem of the necessity of differing distinct primes on each code layer. However, choosing the field $\mathbb{F}_2$ on each layer results in Construction D as a special case and we employ this special case as decoding is more straight forward in $\mathbb{F}_2$.

As described in [14] Construction π_A and π_D in general is defined over any Principle Ideal Domain (PID), however for this work and for practical purposes we will only be using the Gaussian integers $\mathbb{Z}[\mathbb{i}]$ and therefore give some properties of this PID.

Definition - Properties of $\mathbb{Z}[\mathbb{i}]$

The integers $\mathbb{Z}[\mathbb{i}]$ is closed under addition, subtraction and multiplication such that the product, subtraction and sum of any two elements is also an integer. $\mathbb{Z}[\mathbb{i}]$ is not closed under division since if we divide an integer by another integer say 1 divided by 2 the result is not another integer. We can also say that $\mathbb{Z}[\mathbb{i}]$ is not a field. We detail below the main 5 properties of $\mathbb{Z}[\mathbb{i}]$ for addition and multiplication [10]. The 4 properties of table 2.4 show that the Gaussian integers form a commutative ring. The Gaussian integers also can demonstrate if x × y = 0 and x = 0 or y = 0 then there are no zero divisors. From this property it is also shown that the Gaussian integers form an integral domain.

	Addition	Multiplication
Closure	x+y	x×y
Associativity	x+(y+z) = (x+y)+z	x × (y × z) = (x × y) × z
Commutativity	x + y = y + x	x × y = y × x
Distributivity	x × (y + z) = (x × y) + (x × z)	(x + y) × z = (x × z) + (y × z)

Figure 2.4: Properties of the Gaussian integers $\mathbb{Z}[\mathbb{i}]$

2.3.4 Construction by Code Formula

Construction by Code Formula, detailed in [4], describes a general technique that can produce a construction D lattice under certain conditions. Constraints of this method are subject to the chosen codes where [4] applies rules for the codes such that construction D and Construction by Code Formula both produce construction D lattices. The definition of Construction by Code Formula is given below. As in Construction D, we define a set of nested codes:

$$\mathbb{F}_N^2 \supset C_{L-1} \supseteq C_2 \ldots \supseteq C_0 \tag{2.7}$$

with C_l as a $[k_l, N, d_l]$ linear block code. CCF is defined as the set of codewords such that:

$$C_{\text{ccf}} = \psi_0(\mathbf{c}_0) + \psi_1(\mathbf{c}_1) + \ldots + \psi_{L-1}(\mathbf{c}_{L-1}) \tag{2.8}$$

where $\psi(.)$ is a map from the finite field $\mathbb{F}_2^N$ to the reals $\mathbb{R}^N$. The resulting codes from using CCF are called linear lattice codes. The advantages of this scheme become apparent, as shown in [4], due to the scheme allowing the use of conventional decoders. The only main constraint to this scheme is the constraints regarding choice of codes on each layer in order to guarantee the construction of lattice construction D. The constraints are described in subsection 2.3.2 and use of the formula to construct the lattice, guaranteeing the correct formation of lattice points. A Construction D lattice is generated by CCF with these constraints. However, without these constraints the formula of CCF alone does not generate a Construction D lattice as shown in [4]. For a multilayer decoder we need to consider these constraints throughout the design and choice of codes on each layer which satisfy the minimum Hamming distance difference.

2.4 Compute-and-Forward

We assume that a system consists of J sources and B base stations, with each having a single antenna. We employ complex valued channels and describe the process of the Compute-and-forward technique over four stages, encoding, the channel, decoding and recovery.

2.4.1 Encoding

As in [2] each user $j = (1, \ldots, J)$ draws an original message vector over a finite field, described as $\mathbf{w}_j \in \mathbb{F}_p^{k_j}$. The encoder then uses a mapping function $\mathcal{E}$ to map the message vector to the codeword, denoted as $\mathbf{x}_j = \mathcal{E}(\mathbf{w}_j)$. For C&F we assume $\mathbf{x}_j$ lies on a lattice $\Lambda(\mathbf{x}_j = \mathcal{E}(\mathbf{w}_j), \mathbf{x}_j \in \text{mod}\Lambda^{'})$ and therefore the resulting codes are called lattice codes. The general expression for the generation of linear lattice codes is given as: $\mathbf{x}_j = [\mathbf{w}_j \times \text{G}]\text{mod}\Lambda^{'}$ where G is the generator matrix and $\text{mod}\Lambda^{'}$ represents the modulo operation taken over $\Lambda^{'}$. When this is applied to $\mathbf{x}_j$ it is defined as: $\mathbf{x}_j\text{mod}\Lambda^{'} = \mathbf{x}_j - \mathcal{Q}(\mathbf{x}_j)$ where the quantiser is defined as (2.13) in subsection 2.4.3. $\Lambda^{'}$ and Λ define the coarse and fine lattice respectively and can be obtained by employing lattice construction A as introduced in [2] and expressed in subsection 2.3.1. It is important to note that for a given lattice the generator matrix is not specific and therefore the design of the lattice can differ at each encoder. The message rate of a single user is defined as:

$$R = \frac{k_j}{N} log_2(p) \tag{2.9}$$

per complex dimension, where k_j denotes the length of the message from user j.

2.4.2 The Channel

Due to the nature of the wireless communications network system the base station receives a noisy combination of the transmitted signals from the users through the channel, expressed as: $\mathbf{y}_b = \sum_{j=1}^{J} h_{bj}\mathbf{x}_j + \mathbf{z}$, where $\mathbf{y}_b, \mathbf{z}$ and $\mathbf{x}_j \in \mathbb{C}^N$. During the process of decoding for an integer combination of lattice points the base station has to handle two sources of noise.

1. Channel noise: Gaussian noise due to the channel and the scaling factor α_b
2. Self-noise: Due to the scaled channel coefficients not being exactly equal to the integer coefficients

The channel coefficients are defined as: $\mathbf{h}_b = [h_{b1}, \ldots, h_{bJ}]$ where h_{bj} is the channel gain between user j and base station b and $\mathbf{z}$ as Gaussian noise. The received signal is initially scaled by a factor $\alpha_b \in \mathbb{C}$.

$$\mathbf{y}^{'} = (\alpha_b \mathbf{y}_b)\text{mod}\Lambda = \Big(\sum_{j=1}^{J}(\alpha_b h_{bj} \mathbf{x}_{bj}) + \alpha_b \mathbf{z}\Big)\text{mod}\Lambda \tag{2.10}$$

The base station then chooses an integer linear combination of the transmitted codewords to represent an estimate of the scaled received signal. This can be written as:

$$\mathbf{y}^{'} = \Big(\sum_{j=1}^{J} a_{bj}\mathbf{x}_j + \mathbf{z}^{'}\Big)\text{mod}\Lambda \tag{2.11}$$

where $\mathbf{z}^{'}$ is expressed as:

$$\mathbf{z}^{'} = \mathbf{z} + \sum_{j=1}^{J}(\alpha_b h_{bj} - \mathbf{a}_b)\mathbf{x}_j \tag{2.12}$$

and the integer coefficients $\mathbf{a}_b$ are chosen to minimise $E[||\mathbf{z}^{'}||^2]$. Let $\mathbf{a}_b = [a_{b1}, a_{b2}, \ldots, a_{bJ}]$ denote the coefficient vector of the linear combination. The scaling factor α_b aims to move the scaled channel $\alpha_b \mathrm{h}_b$ as close as possible to the integer coefficient vector $\mathbf{a}_b$. In order to uniformly distribute the input signals $\mathbf{x}_l$, a random dither vector $\mathrm{d}_j \in \mathbb{C}\text{mod}\Lambda$ may be employed, written as $\mathbf{x}_j = (\mathbf{x}_j + \mathrm{d}_j)\text{mod}\Lambda$. The dither vector is known by both transmitter and receiver and is removed at the base stations. As the dither vector was introduced in [2] to prove the computation rate achieved and since we are more interested in the lattice structure of C&F, the dither process is omitted in this work.

2.4.3 Decoding

The base station decodes an integer combination of the scaled received signal $\alpha_m \mathbf{y}_b$, which is a lattice point in Λ. In order to do so it applies a quantiser $\mathcal{Q}$, which quantises $(\alpha_b \mathbf{y}_b)$ to the closest lattice point and can be generally expressed as:

$$\mathcal{Q}_{\Lambda}(\mathbf{y}) \triangleq \underset{\lambda \in \Lambda}{\text{argmin}}||\mathbf{y} - \lambda|| \tag{2.13}$$

hence the integer combination $\sum_{j=1}^{J} a_{bj}\mathbf{x}_j$ is also a lattice point and $\mathbf{a}_{bj}$ are the integer coefficients.

The base station then maps this combination to a codeword $\mathbf{u}_b$ in Λ/Λ'. This can also be expressed as:

$$\mathbf{u}_b = \oplus_{j=1}^{J} q_{bj}\mathbf{w}_j \tag{2.14}$$

where this is evaluated over the finite field F_p. The channel operates over the complex and the finite field coefficients are mapped back to the reals, $\mathbb{R}$. The definition of the map is expressed as $q_{bj} = g^{-1}([a_{bj}])$ where $\mathbf{a}_b = [a_{b1}, a_{b2}, \dots, a_{bJ}]$ are the integer coefficients vectors and g^{-1} is a map that takes elements in the reals to the finite field.

2.4.4 Recovery

Finally, the Central Processing Unit (CPU), also known as the hub, recovers the individual data messages $\mathbf{w}_j$ using the B linear equations from the B base stations. The data messages from all J users can be recovered if and only if the coefficient matrix is full rank, Rank($\mathbf{Q}$) $= J$. Let $\mathbf{Q} = [\mathbf{q}_1, \mathbf{q}_2, \dots, \mathbf{q}_b]$ be the coefficient matrix where $\mathbf{q}_b = [q_{b1}, q_{b2}, \dots, q_{bJ}]$ is the coefficient vector for base station b and $q_{b1}, \dots, q_{bJ}$ are the coefficients taking values in $\mathbb{F}_2$. When $B < J$, it is equivalent to say that $\mathbf{Q}$ is singular and not all data from all sources cannot be recovered. The two-way relay channel is an example of this where we demonstrate in chapter 4 potential errors due to $\mathbf{Q}$ being singular. Decoding is carried out by multiplying the data messages by the inverse or the pseudo-inverse of $\mathbf{Q}$. The matrix will be non-singular if $B \geq J$ and all vectors $\mathbf{q}_b$ are linearly independent.

2.5 Reed-Muller codes

Reed-Muller codes are linear block codes and error-correcting codes usually used in wireless communications systems. The process of encoding these codes can be described in several ways, within this work we construct a generator matrix G. A generator matrix for a Reed-Muller RM(r,m) code of length $N = 2^m$ and order r can be generated by calculating the element-wise product, denoted as $\wedge$, of $g_0, \dots, g_n$ with permutations up to order r. For example the generator matrix of the RM(0,4) code contains the length 16 all 1's vector denoted as:

$$g_0 = (1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1) \tag{2.15}$$

If we now choose the RM(1,4) code the vectors also include g_1, g_2, g_3, g_4, which can be generated by filling a 4 x 16 array with columns of the form $(0, 0, 0, 0), (0, 0, 0, 1), (0, 0, 1, 0), \dots, (1, 1, 1, 1)$

the rows of the array then form g_1 to g_4 as shown in 2.16.

$$\begin{aligned}
g_0 &= (1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1)\\
g_1 &= (1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0)\\
g_2 &= (1,1,1,1,0,0,0,0,1,1,1,1,0,0,0,0)\\
g_3 &= (1,1,0,0,1,1,0,0,1,1,0,0,1,1,0,0)\\
g_4 &= (1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0)
\end{aligned} \tag{2.16}$$

Extending to the RM(2,4) code where $m = 4$ and $r = 2$ for $N = 16$ we use (2.16) and element-wise product to generate the rows of the matrix. The vectors are made up of $0, 1^{st}$ and 2^{nd} order expressions due to $r = 2$ therefore the rows of the generator matrix are $(g_0, g_1, g_2, g_3, g_4, g_1 \wedge g_2, g_1 \wedge g_3, g_1 \wedge g_4, g_2 \wedge g_3, g_2 \wedge g_4, g_3 \wedge g_4)$. Therefore the generator matrix for this particular code $G_{RM(2,4)}$ generating the (16,11,4) code is given as:

$$G_{RM(2,4)} = \begin{pmatrix}
1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1\\
1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0\\
1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0\\
1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0\\
1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&1&1&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&1&0&1&0&0&0&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&1&1&0&0&0&0&0&0\\
1&0&1&0&0&0&0&0&1&0&1&0&0&0&0&0\\
1&0&0&0&1&0&0&0&1&0&0&0&1&0&0&0
\end{pmatrix} \tag{2.17}$$

RM(r,m) codes are linear codes over $\mathbb{F}_2$ with code length N and minimum distance 2^{m-r}. RM(m-2,m) codes, described as $(2^m, 2^m - m - 1, 4)$ define the family of extended Hamming codes with minimum distance $d_{min} = 4$. RM(0,m) codes described as $(2^m, 1, 2^m)$ are grouped into the definition of repetition codes with a minimum distance of $d_{min} = 2^m$. The general description of RM codes is $(2^m, k, 2^{m-r})$ where $k = \sum_{i=0}^{r} \begin{pmatrix} m \\ i \end{pmatrix}$. The use of these codes within this work is ideal as the constraints of minimum distance is always a factor of 4 between different codes. This means that the requirements for the minimum distance between code layers for generating construction D lattice can be fulfilled. The family of Reed-Muller codes also creates a set of nested codes which is also a requirement of construction D.

2.6 Soft decision decoders

2.6.1 Trellis decoding

The trellis decoder is a decoding method that uses a trellis structure to examine the received code sequence of a given length and usually results in an estimate of the given codeword or dataword. Figure 2.5 shows a typical trellis diagram with states $S_0, \ldots, S_4$ and the estimated dataword corresponding to each state given as $\mathrm{w}_0, \ldots, \mathrm{w}_4$. The branches that transition from one state to another denote a transition of value 1 or 0. The dotted lines denote value 1 and solid line value 0. In order to determine the path at which the state will transition to a branch metric is calculated. The branch metric is determined usually by the chosen decoder i.e. soft or hard decision however in general this metric determines the probability of the state transition given the received sequence and the value of the next state. The solid circles are labelled to show the bit value.

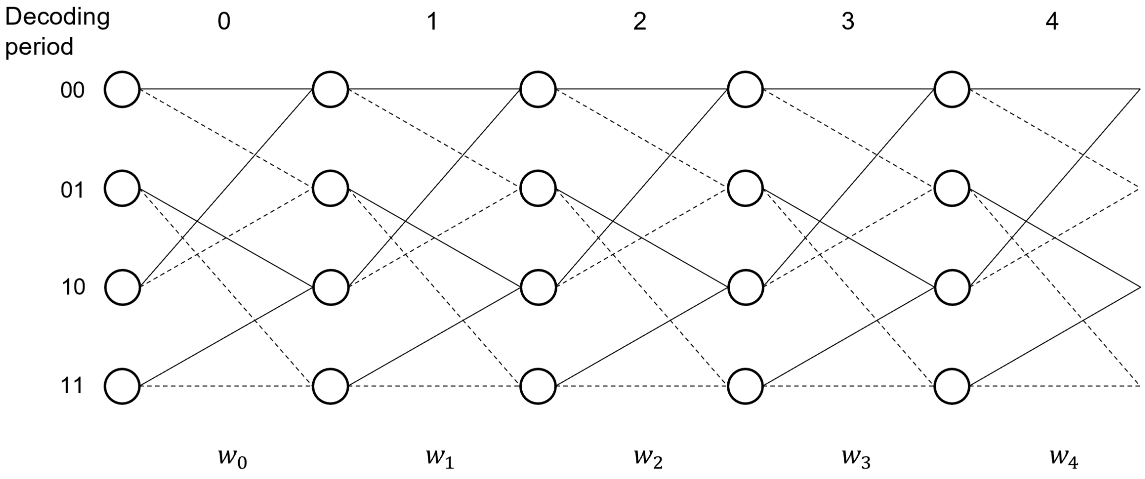


Figure 2.5: Example trellis diagram

2.6.2 Viterbi decoder

The Viterbi decoder implements the maximum likelihood Viterbi algorithm introduced by [15] for decoding codes encoded using a convolutional encoder by a trellis structure. The

Viterbi algorithm operates by calculating the maximum a posteriori probability estimate of the most likely path or sequence of states. The basic steps of the Viterbi algorithm are as follows:

- Define the trellis and calculate the branch metrics in order to determine the weight of the branches - the branch metric is calculated by using equation 2.18, the result of which increases as the probability of the branch decreases, therefore the lowest weight has the highest probability.
- At each state i calculate the minimum weight path to the current state $i - 1$ and keep a record of the minimum weight steps of the path - compare the weights of the previous branches and select lower weight branch (highest probability) and add that previous branch weight to the current branch weight.
- Identify the complete path of minimum weights (highest probabilities)
- Identify the estimate of the code

For state transitions through the trellis the branch metrics for hard decision are determined quite straight forwardly, i.e. if the received code bit is 0.55555 and another is 0.99999, both bits are deemed to be 1 even though the second bit is much closer than the first. For the soft decision Viterbi decoder, the branch metrics are calculated by the square of the difference between the received value and the expected value. For example, if the expected value are given as n_i and the received bits are given as $\mathbf{c}_i$ the branch metric BM is:

$$BM = \sum_{i=1}^{N} (n_i - \mathbf{c}_i)^2 \tag{2.18}$$

The Viterbi algorithm can be implemented for soft or hard decision decoding. Even though we do not implement this algorithm in this project we look at the description of soft decision decoding as soft decision decoding is used in industry and is also our approach for the multilayer decoder.

2.6.3 BCJR decoder

The work of [16] originally introduced the BCJR algorithm which is a maximum a posteriori algorithm that aims to minimize the probability of bit error. The algorithm is based on trellises and is sometimes known as the forward-backward algorithm due to calculating the forward probabilities α and backward probabilities β. The algorithm uses the trellis to return the log-likelihood ratios of the decoded data. In order to do so, at a given section the algorithm calculates the probabilities of the next state ss (all nodes to the right of the current section) given the received signal. This is denoted as $\alpha_{(i+1)}$:

$$\alpha_{i+1,ss} = \alpha_i(s_1)\gamma_i(s_1, ss) + \alpha_i(s_0)\gamma_i(s_0, ss) \tag{2.19}$$

where i denotes the trellis section, s denotes the starting branch of the current state and ss denotes the branch ending at the next state. The decoder then calculates the probabilities of the paths to the left of the branch nodes, given by:

$$\beta_i(s) = \beta_{i+1}(ss_1)\gamma_i(s, ss_1) + \beta_i(ss_0)\gamma_i(s, ss_0) \tag{2.20}$$

where

$$\gamma_i(s, ss_{\mathrm{w}}) = P(br_i(s, ss_{\mathrm{w}})|\{s, \mathbf{y}_i\}) \tag{2.21}$$

is the a posteriori probability of the branch $br_i(s, ss)$ given the received signal $\mathbf{y}_i$ for the trellis section i. The likelihood ratio of the data corresponding to the particular trellis section is given by:

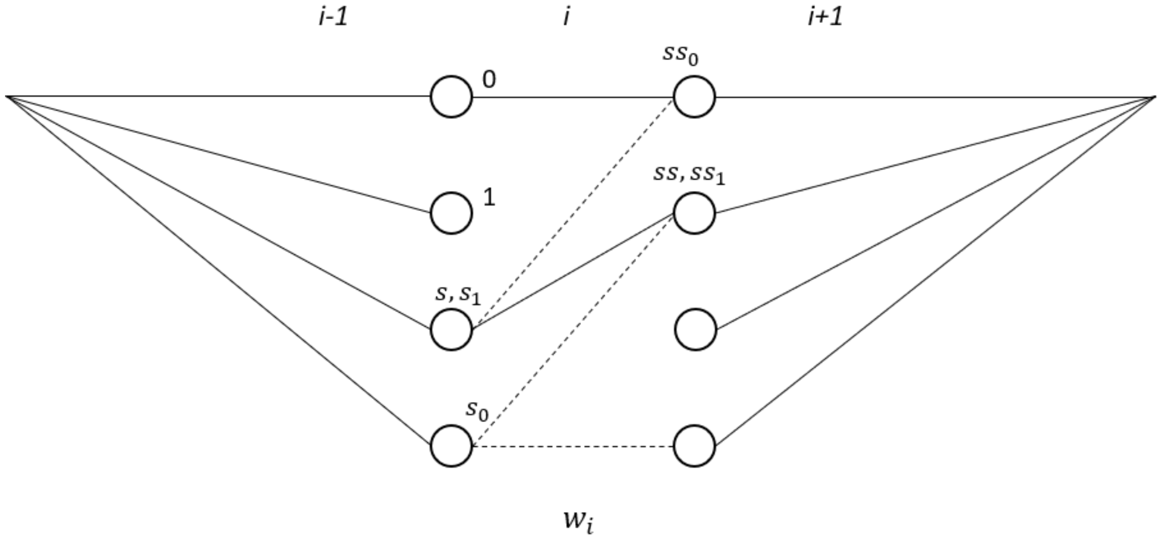


Figure 2.6: Example section of trellis highlighting branches starting with current state s and ending in next state ss

$$LR(\mathrm{w}_i) = \frac{\sum_{k=0}^{n-1} \alpha_i(s)\gamma_i(s, ss_1)\beta_i(ss_1)}{\sum_{k=0}^{n-1} \alpha_i(s)\gamma_i(s, ss_0)\beta_i(ss_0)} \tag{2.22}$$

Where (2.22) denotes the sum of the probabilities of all paths through the trellis.
In conventional decoding there is no a priori information as both the probability of the bit values $\{0, 1\}$ are equally likely. Our work in Section 3.4 employs the BCJR algorithm for a particular scenario and specific encoding pattern where we do indeed have extrinsic information at particular code bits which is passed on from the previous decoder. The use of this information helps to reduce the error probability of decoding those codes bits on that layer, where this will be further described in Section 3.4. Usually the a priori information is calculated by a likelihood ratio which for a Gaussian channel can be calculated by:

$$P(\mathbf{y}_i | br_i(s, ss_{\mathrm{w}}) = e^{(-\delta^2(s,ss_{\mathrm{w}})/2\sigma^2)} \tag{2.23}$$

where $\delta_i(s, ss_{\mathrm{w}})$ denotes the Euclidean distance between the received signal and the transmitted signal. The complexity of this algorithm is much greater than that of Viterbi and requires in-depth knowledge of Signal to Noise Ratio (SNR) however the Viterbi algorithm does not. The use of convolutional codes and subsequently the use of the BCJR decoder shown in our work in Section 3.4 detailed the error floor and the potential performance of the codes. Subsequently, in later sections the implementation of codes with higher Hamming distances have been introduced into the system model in order to demonstrate a higher performance.

2.6.4 Reed-Muller decoder

As with encoding, there are several approaches to decoding Reed-Muller codes. The work of [17] originally introduces a general trellis decoder for Reed-Muller codes, the diagrams shown in Figure 10 of [17] show code lengths up to and including 32. The method uses the iterated squaring construction in order to generate the codes and in turn uses trellis decoding. The squaring construction is represented by the trellis diagram in Figure 2.7, taken from [17], where the trellis is used in its traditional sense to encode the dataword and consists of $\tilde{m}$ sections joined at $\tilde{m}$ states. The branches correspond to particular codes that are joined at a singular state however the states now represent a Cartesian product of the codeword. This produces a chain of nested codes and as shown in [17].

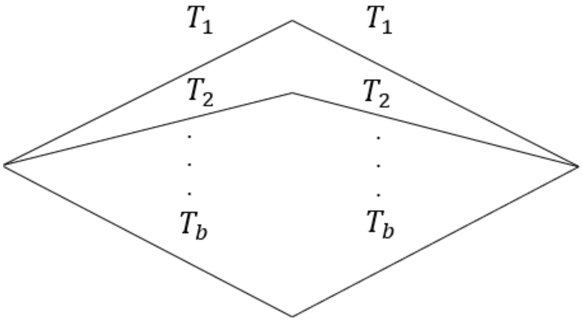


Figure 2.7: Squaring construction trellis representation

The author of [17] highlights that the binary RM codes, with length $N = 2^m$, can be represented as trellises and hence can be soft decision decoded using the Viterbi or MAP algorithm. The use of conventional and well established decoding algorithms such as these is advantageous as their use in industry is prominent. Figure 2.8, taken from [17], shows a trellis for the RM(8,1) code. The horizontal lines of the trellis contain two trellis sections as the branches are pairs of branches representing 2 bits. The number of bits on each path is 8 representing the code length 8. There are only two paths through the trellis in this case and therefore this only sends one bit per codeword, this is a repetition code.

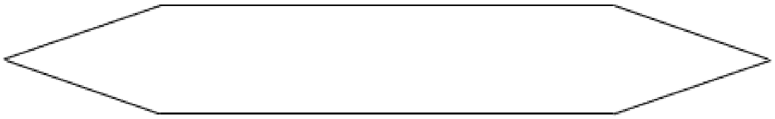


Figure 2.8: Four section trellis diagram for Reed-Muller code RM(8,1) of length 8

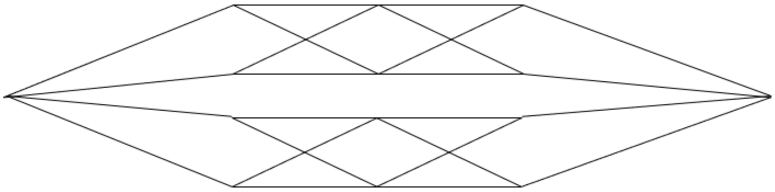


Figure 2.9: Four section trellis diagram of Reed-Muller (8,4) code

Figure 2.9, taken from [17], demonstrates the trellis of the RM(8,4) code. Again, each branch represents two bits. There are 16 paths through the trellis where this corresponds to 4 information bits.

2.6.5 Soft Decision Hamming Codes

The work of [18] introduces a low complexity soft decision Hamming decoder that decodes by detecting error patterns belonging to the same syndrome. The error patterns are given in matrix $\mathbf{E}$ which is tabulated in advance for each possible syndrome. To calculate a syndrome we examine the received signal $\mathbf{y}$ and the codeword $\mathbf{c}$. For the syndrome-based soft decision decoding algorithm a 7,4 Hamming code is generated and encoded, in order to produce $\mathbf{c}$, given by:

$$\mathbf{c} = \mathbf{wG} \tag{2.24}$$

The signal is then modulated and subject to additive white Gaussian noise (AWGN), denoted as $\mathbf{z}$. The received signal is given as:

$$\mathbf{y} = \mathbf{x} + \mathbf{z} \tag{2.25}$$

where $\mathbf{x} = \mathcal{E}(\mathbf{c})$ and $\mathcal{E}$ is an encoding map taking the codeword from the finite field $\mathbb{F}_2^N$ to the reals. The difference between these two vectors is the error pattern vector $\mathbf{e}$ which we can write the received signal as: $\mathbf{y} = \mathbf{c} + \mathbf{e}$ where $\mathbf{e}$ denotes the error caused by noise. To obtain the syndrome we need to hard decision decode the received signal to give a logical result. We denote this as $\mathbf{y}_{hd} = \mathbf{c}_{hd} + \mathbf{e}_{hd}$ where $+$ denotes modulo two sum and $\mathbf{c}_{hd}$ and $\mathbf{e}_{hd}$ denote the hard decision components of $\mathbf{y}_{hd}$. If the parity check matrix is $\mathbf{H}$ then $\mathbf{Hy}_{hd}^T = \mathbf{H}(\mathbf{c}_{hd} + \mathbf{e}_{hd})^T = \mathbf{Hc}_{hd}^T + \mathbf{He}_{hd}^T = \mathbf{He}_{hd}^T$ as $\mathbf{Hc}_{hd}^T = 0$ for the codeword and $\mathbf{Hy}_{hd}^T$ is called the syndrome of $\mathbf{y}_{hd}$. If the syndrome is 0 then no errors have occurred. To determine the correct error pattern we calculate this by multiplying the absolute values of LLR's of the received signal with each row of the matrix $\mathbf{E}$. The resulting row vector is then added together, where the vector with the lowest sum of L-values indicates the error pattern with highest probability of correct decoding. For a syndrome-based technique, Log-likelihood ratios (LLR) of the received signal $\mathbf{y}$ are calculated.

$$\mathbf{L}(\mathbf{x}|\mathbf{y}) = \ln \frac{P(\mathbf{x} = +1|\mathbf{y})}{P(\mathbf{x} = -1|\mathbf{y})} \tag{2.26}$$

which subsequently leads to

$$\mathbf{L}(x|y) = \ln \frac{exp(-\frac{E_b}{N_o}(\mathbf{y} - 1)^2)}{exp(-\frac{E_b}{N_o}(\mathbf{y} + 1)^2)} = \frac{4E_b}{N_o}\mathbf{y} \tag{2.27}$$

where the assumption is that logical zero and one have equal probability. For this work an adaptation of this decoder has been developed in order to implement unipolar codes for a single frame. The complexity of this algorithm is significantly lower compared to existing decoding algorithms such as exhaustive maximum likelihood and Trellis-based due to its approximation of maximum-likelihood.

2.7 Summary

As briefly described in the introduction chapter Compute-and-Forward was originally introduced using lattice construction A. In this chapter the complexity of this construction has been explained along with an alternative lattice construction, namely lattice construction D which allows the use of binary codes. Generation of a construction D lattice has previously been researched in [4] and described as CCF, hence this thesis demonstrates the implementation of the construction in C&F. Through the use of CCF binary codes are now able to be employed, which aligns with standard practice in wireless communications networks. Furthermore by implementing a construction D lattice the choice of codes employed becomes more flexible and in turn the choice of decoder. This is in contrast to lattice construction A. Codes such as convolutional and Reed-Muller codes have been researched and therefore implemented into this work to highlight this aspect.

Chapter 3

Lattice Decoding of C&F using Construction D

3.1 Introduction

As highlighted in the introduction, original research introducing C&F uses Construction A lattices for encoding and decoding [2] [12]. The impracticality of employing this construction is described in subsection 2.3.1 and highlights the reasoning of the approach taken to this work. In addition to this the use of binary codes is also deemed practical due to their use within telecommunication standards. As mentioned previously, in order to implement binary codes we generate a construction D lattice using CCF as described in subsection 2.3.4. The advantages for decoding and multilayer decoding are also highlighted in this subsection.

Within this chapter we detail the design of the successive layered decoder and provide general expressions for decoding L layers. During the course of this work we have uncovered an issue that complicates the multilayer decoding process, in that inter-layer interaction occurs between layers, where a carry signal from the codes on one layer affects the decoding of the next layer. The effects of this have not previously been documented within this research field. We detail and derive mathematical expressions which describe the output of each layer decoder and that each layer can indeed be successfully decoded following these expressions for the inter-layer interaction. The main contributions for this chapter are as follows:

- We describe and implement Construction D multilayer encoder and decoder in a C&F system model.
- We derive mathematical expressions which represent the inter-layer interaction, and evaluate the output of each layer decoder. We provide expressions based on the general case meaning that they can be employed for multiple users and any number of layers.
- We determine the effects the inter-layer interaction has on the codes and detail additional requirements on the codes of each layer.
- We evaluate the C&F system and demonstrate that we can successfully decode multiple layers using multiple users with the derived expressions.

We begin the chapter by providing a description of the system model and describing the operation of the encoder(s) and multilayer decoder. Examples demonstrating the interaction signals occurring between layers are provided detailing this for a two and three user system.

3.2 C&F System Model

We consider a multi-user $j = (1, \ldots, J)$, C&F system model with single AP. Each terminal is transmitting Construction D nested lattice codeword(s), $\mathbf{x}_l \in \mathbb{R}^N$. Description of construction D can be found in subsection 2.3.2. Figure 3.1 details the CCF encoding process for a multi-layer construction from a single source for complex message(s). The user draws a binary message vector over a finite field $\mathbf{w} \in F_2^k$ where $k = 2\sum_{i=l}^{L-1} k_i$ and k_i is the message vector length for the i^{th} layer. The encoder is split into separate and identical encoders for In-phase (I) and Quadrature (Q), $\mathbf{w}^I$ and $\mathbf{w}^Q$, which in turn are broken down into layer encoders $(\mathcal{E}_0^{\{I,Q\}}, \ldots, \mathcal{E}_{L-1}^{\{I,Q\}})$. As shown in the diagram, the output of each component is then summed producing the resultant lattice codes $\mathbf{x}^{\{I,Q\}} = 2^{L-1}\mathbf{c}_L^{\{I,Q\}} + \ldots + 2\mathbf{c}_1^{\{I,Q\}} + \mathbf{c}_0^{\{I,Q\}}$ using equation (2.8). Subsequently, the final output of the encoding process for j terminal is:

$$\mathbf{x}_j = \sum_{l=0}^{L-1} 2^l \mathbf{c}_{lj}^{\{I,Q\}} \tag{3.1}$$

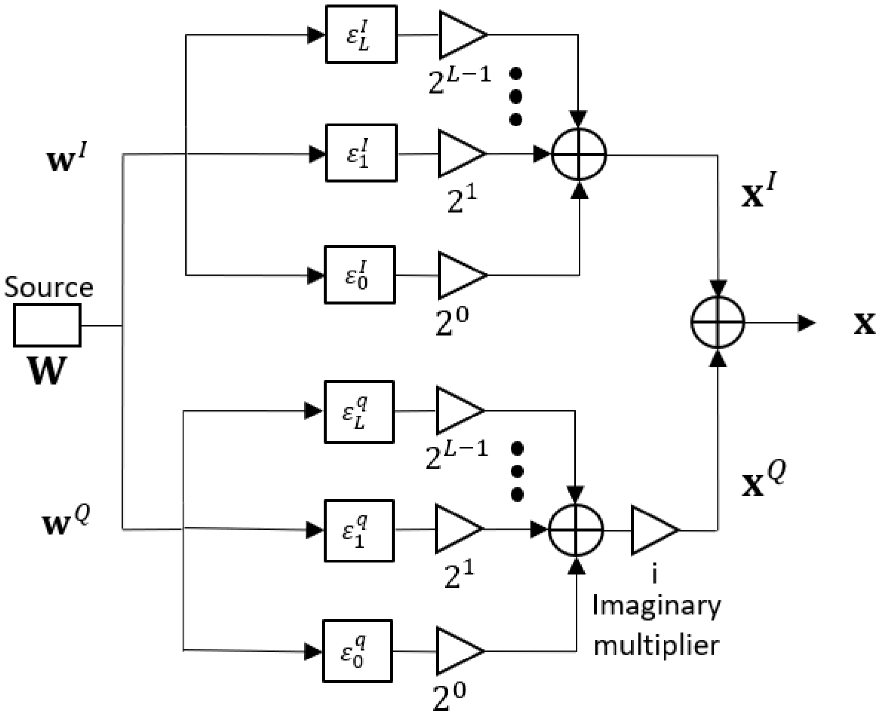


Figure 3.1: Multilayer encoding process for complex signals

where $\mathbf{c}_{lj}^{\{I,Q\}}$ is the codeword from code C_l for layer l from the j^{th} terminal. The received signal:

$$\mathbf{y} = (h_1\mathbf{x}_1 +, \ldots, +h_J\mathbf{x}_J) + \mathbf{z} \tag{3.2}$$

The noise vector $\mathbf{z} \backsim \mathcal{N}(0, \mathbf{I}_J\sigma^2)$ and the channel coefficients $h_1, \ldots, h_J$. The received signal is then multiplied by a factor α and then modulo-2^L operation applied to the result, where the modulo operation is denoted as mod:

$$\begin{aligned} \mathbf{y}' &= (\alpha\mathbf{y})\text{mod}2^L = (\alpha h_1\mathbf{x}_1 +, \ldots, +\alpha h_J\mathbf{x}_J + \alpha\mathbf{z})\text{mod}2^L \\ &= (a_1\mathbf{x}_1 +, \ldots, +a_J\mathbf{x}_J + \mathbf{z}')\text{mod}2^L \end{aligned} \tag{3.3}$$

where $\mathbf{z}' = \mathbf{z} + (\alpha h_1 - (a_1)\text{mod}2^L)\mathbf{x}_1 +, \ldots, +(\alpha h_J - a_J)\mathbf{x}_J$. We can show $(a_1\mathbf{x}_1 +, \ldots, + a_J\mathbf{x}_J)\text{mod}2^L = ((a_1)\text{mod}2^L(\mathbf{x}_1)\text{mod}2^L +, \ldots, +(a_J)\text{mod}2^L(\mathbf{x}_J)\text{mod}2^L)\text{mod}2^L$, which means that we can replace a_j by $a_j\text{mod}2^L$. Hence the integer channel coefficients may be treated as positive and also between 0 and 2^{l-1}. a_j are integers chosen to minimise $E[\|\mathbf{z}'\|^2]$. In this chapter we assume that the channel coefficients h_J are Gaussian integers due to implementation of Gaussian integer lattice and hence α can be set to 1 and a_J are equal to

h_J, so $\mathbf{z} = \mathbf{z}^{'}$.

3.3 Multilayer Lattice Decoding of C&F

Due to the multi-layer construction of CCF and Construction D the approach to decoding nested lattice codes is successive layer decoding, meaning each layer is decoded in turn from layer 0 (inner most layer) to layer L-1 (outer most layer). As in [4] [14], this approach produces good performance for lattice decoding using lattice construction D.

As shown in Figure 3.1 the operation of the decoder shows that at layer 0 (the innermost layer) we take $\mathbf{y}^{'}$ modulo 2, and decode the result to generate $\hat{\mathbf{c}}_0$. At layer 1 we subtract $\hat{\mathbf{c}}_0$ from $\mathbf{y}^{'}$ then divide by 2 to generate $\mathbf{y}_1^{'}$. We then take this modulo 2 and decode to generate $\hat{\mathbf{c}}_1$. For the general case, at layer $l^{'}$ we subtract the previous layer estimate of the codeword $\hat{\mathbf{c}}_{l'-1}$ from the previous layer scaled received signal $\mathbf{y}_{l'-1}^{'}$ and divide by 2 to form $\mathbf{y}_{l'}^{'}$, which we then take modulo 2 and decode to find $\hat{\mathbf{c}}_{l'}$. Note that this decoder is preceded by a modulo 2^L operation. By subtracting the estimate of the code on the previous layer we are assuming it has been decoded correctly, cancelling out any previous layer interference. The modulo operation provides a cancellation of codes on outer layers and due to the use of binary codes we apply modulo 2. While this is effective for a single user decoding

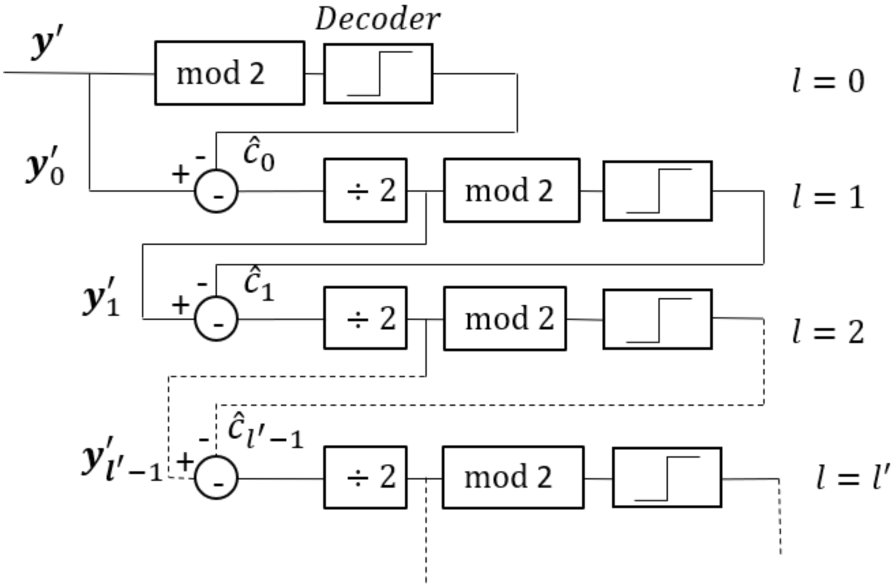


Figure 3.2: Multilayer decoding process

of a construction D lattice, it results in an interaction between layers when applied to C&F.

Initially, within this chapter we demonstrate the error floor presented when the interaction signals are omitted from consideration at the decoder and present an example of a three layer scheme for a single user using a specific codeword and detail the condition at which the decoder can perfectly decode layer l'. As will be described the inner layer does not have any preceding layers and therefore the decoder only considers the codewords of its own l'^{th} layer. Outer layers are affected by this 'interaction' occurring between layers which we address through examination of the codewords of each layer. The second part of this chapter gives a description regarding the interaction signal effect on each layer where we consider the unit and complex integer valued channels. Detailed derivations are given on how the decoded output may be predicted. Furthermore, it is shown that the effect imposes additional requirements on the codes used on each layer in order to ensure decodability. These additional requirements are given in the form of theorems followed by corresponding proofs. An example of a three layer scheme is given under each channel condition and performance results discussed.

3.4 Demonstration of error floor for inner and outer layers

As will be demonstrated in Sections (3.6 - 3.9) the interaction signals are generated during multilayer decoding of C&F. The investigation of these interaction signals led to many simulations and testing to determine the exact nature of the signal, why the inner layer is not effected and the remaining outer layers are. For this investigation a two user two layer nested binary code is implemented, as in [4], where we employ convolutional codes for both layers. User 1 and 2 generate a length 150 message vector $\mathbf{w}$ which is split into a length 10 data vector $\mathbf{w}_{0\{1,2\}}$ and a length 140 data vector $\mathbf{w}_{1\{1,2\}}$. Each are convolutionally encoded using a $\frac{1}{2}$ rate convolutional encoder. In order to ensure that each dataword is encoded to the same length we add padding to each. For $\mathbf{w}_{1\{1,2\}}$ we pad with 0 in positions (1,16,31,46,61,76,91,106,121,136). For layer 0 we use repetition vectors $\mathbf{r}_1 = [1,0,0,1,0,0,1,0,0,1,0,0,1,0,0]$ and $\mathbf{r}_2 = [0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]$ to replace each data bit 1 and 0 in $\mathbf{w}_{0\{1,2\}}$ respectively. The padded datawords are then encoded using a half rate convolutional encoder, where equation (2.8) is employed in order to construct the lattice codes $\mathbf{x}_1$ and $\mathbf{x}_0$. The received signal is generated using equation (2.8) where $h_1 = h_2 = 1$. Decoding is carried out successively as described in Section 3.3 using a BCJR decoder for each layer. We incorporate the work of [19] for the BCJR decoder however adapting slightly for a priori information. We incorporate a priori information due to the nature of the repetition codes used which will be advantageous for those particular code bits when entering the decoder. The estimated decoded output on each layer is the exclusive OR (XOR) combination of the data. We present the results of this in the form of BER performance in Figure 3.3. Figure 3.3 details the result when decoding two layers, layer 0 and layer 1 without the consideration of the interaction signal. The inner layer (layer 0) has been successfully decoded however the outer layer presents an error floor which for the moment cannot be overcome. The error floor starts to show at fairly low SNR which, when we compare this to layer 0, is approximately 9dB. This provides the evidence that there is an interaction happening between code layers which has not been accounted for.

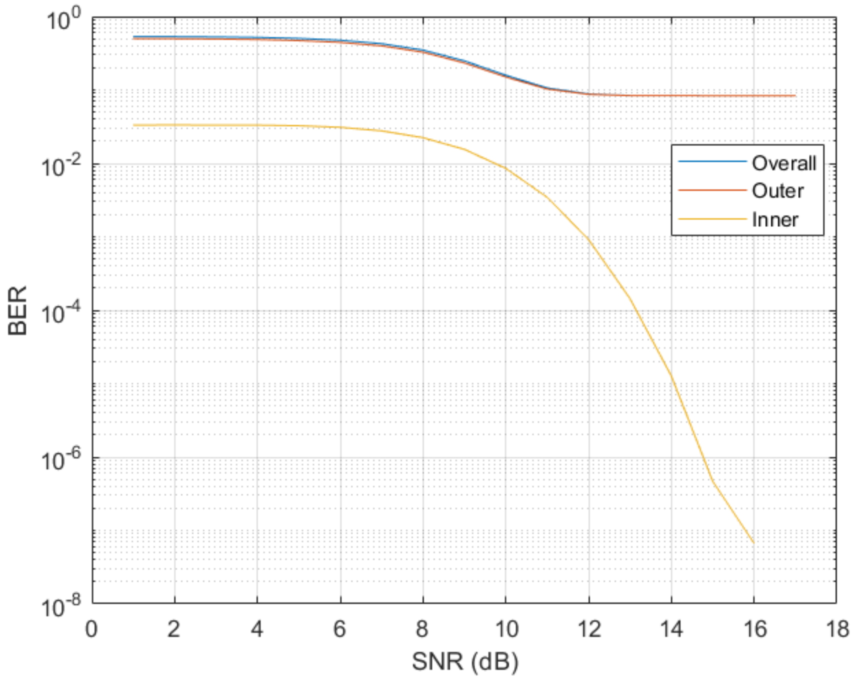


Figure 3.3: Example of error floor on outer layer code (layer 1)

3.5 Interaction between code layers

For this section we consider a two user and a three layer $l = \{0, 1, 2\}$ C&F scheme. We assume that the channel coefficients are unit valued, $h_1 = h_2 = 1$, and the codewords $\mathbf{c}_{l1}$ and $\mathbf{c}_{l2}$ are real. Due to the choice of channel coefficients we are able to omit them from the received signal. The simplest form of channel coefficients is employed as it is sufficient for our initial demonstration of this interaction, we also omit the presence of noise as this is not the focus of our demonstration. The codes used for this scenario will be Reed-Muller codes, 1st order RM(1,5) code for the inner layer, 3rd order RM(3,5) for the middle layer and the outer layer is left uncoded.

In order to demonstrate the interaction discovered to occur between code layers we introduce a particular codeword for each layer. The data generated from each source S_1 and S_2 for all layers is given below, along with the associated codeword.
For layer 0:

$$\mathbf{w}_{01} = [1, 1, 1, 1, 0, 1] \tag{3.4}$$

$$\mathbf{w}_{02} = [0, 1, 0, 1, 0, 1] \tag{3.5}$$

$$\mathbf{c}_{01} = [1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 1] \tag{3.6}$$

$$\mathbf{c}_{02} = [1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0] \tag{3.7}$$

For layer 1:

$$\mathbf{w}_{11} = [0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1] \tag{3.8}$$

$$\mathbf{w}_{12} = [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1] \tag{3.9}$$

$$\mathbf{c}_{11} = [1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 1, 0, 0, 0] \tag{3.10}$$

$$\mathbf{c}_{12} = [1, 0, 0, 1, 1, 1, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1] \tag{3.11}$$

Layer 2 is uncoded and therefore the codewords are also the datawords.

$$\mathbf{w}_{21} = [0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 1] \tag{3.12}$$

$$\mathbf{w}_{22} = [0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1, 1, 1] \tag{3.13}$$

$$\mathbf{c}_{21} = [0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 1] \tag{3.14}$$

$$\mathbf{c}_{22} = [0,0,0,0,0,1,1,1,1,0,0,0,0,1,1,1,0,0,0,1,1,1,0,0,0,0,0,1,0,1,1,1] \tag{3.15}$$

Using (2.8) the construction D lattice is generated, for user 1 S_1 and user 2 S_2:

$$\begin{aligned} S_1 &= 4\mathbf{c}_{21} + 2\mathbf{c}_{11} + \mathbf{c}_{01} \\ &= [3,2,7,2,6,1,4,1,6,1,0,1,5,2,3,4,4,1,0,3,5,6,3,4,1,2,7,0,6,1,4,5] \end{aligned} \tag{3.16}$$

$$\begin{aligned} S_2 &= 4\mathbf{c}_{22} + 2\mathbf{c}_{12} + \mathbf{c}_{02} \\ &= [3,0,1,2,2,7,6,5,7,2,3,0,0,5,4,5,2,1,0,5,5,4,1,0,0,1,0,5,1,4,5,6] \end{aligned} \tag{3.17}$$

Thus, the received signal, ignoring noise and interference is:

$$\begin{aligned} \mathbf{y} &= S_1 + S_2 \\ &= [6,2,8,4,8,8,10,6,13,3,3,1,5,7,7,9,6,2,0,8,10,10,4,4,1,3,7,5,7,5,9,11] \end{aligned} \tag{3.18}$$

We now apply a modulo 8 operation to the received signal.

$$\begin{aligned} \mathbf{y}^{'} &= (\mathbf{y})\text{mod}8 \\ &= [6,2,0,4,0,0,2,6,5,3,3,1,5,7,7,1,6,2,0,0,2,2,4,4,1,3,7,5,7,5,1,3] \end{aligned} \tag{3.19}$$

Finally, the modulo 2 operation is applied generating the scaled received signal $\mathbf{y}^{'}_{\text{mod}2}$.

$$\begin{aligned} \mathbf{y}^{'}_{\text{mod}2} &= (\mathbf{y}^{'})\text{mod}2 \\ &= [0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1] \end{aligned} \tag{3.20}$$

This result is passed to the inner layer decoder which in this case without the presence of noise and the channel coefficients being unit valued, the codeword is the same as the scaled received signal for the inner layer. As we will see this applies to all layers.

$$\hat{\mathbf{c}}_0 = [0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1] \tag{3.21}$$

where $\hat{\mathbf{c}}_0$ denotes the estimate of the codeword on that layer. We can see that $\hat{\mathbf{c}}_0$ is the XOR combination of $\mathbf{c}_{01}$ and $\mathbf{c}_{02}$. It also follows that the result is therefore a codeword of the inner code and that the resulting data is also the XOR of the input data on layer 0. This is what was expected as the decoded output on this layer.

As detailed in Section 3.3 and shown in Figure 3.2 the estimate of the decoded codeword $\hat{\mathbf{c}}_0$ is subtracted from the scaled received signal $\mathbf{y}^{'}$ to produce $\mathbf{y}^{'}_{1'}$ such that we can begin decoding the middle layer.

$$\mathbf{y}^{'}_{1'} = \frac{\mathbf{y}^{'} - \hat{\mathbf{c}}_0}{2} = [3, 1, 4, 2, 0, 4, 1, 3, 2, 1, 1, 0, 2, 3, 3, 0, 3, 1, 0, 0, 1, 1, 2, 2, 0, 1, 3, 2, 3, 2, 0, 1] \tag{3.22}$$

$$\mathbf{y}^{'}_{1'}\mathrm{mod}2 = [1, 1, 0, 0, 0, 0, 1, 1, 0, 1, 1, 0, 0, 1, 1, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1] \tag{3.23}$$

Again, (3.23) is subject to the middle layer decoder where the decoded result produces:

$$\hat{\mathbf{c}}_1 = [1, 1, 0, 0, 0, 0, 1, 1, 0, 1, 1, 0, 0, 1, 1, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1] \tag{3.24}$$

By looking at $\mathbf{c}_{11}$ and $\mathbf{c}_{12}$ it can be shown that $\hat{\mathbf{c}}_1$ is not the XOR combination of the codewords on this layer. This also proves that again, under these circumstances that we again demonstrate the interaction occurring between the inner and middle layer. In order to establish the nature of the interaction on the middle layer we look at the scaled received signal and the codewords on the middle layer as well as the inner layer. We know through our analysis so far that there is evidence for an interaction between the inner and middle layers. We could also predict that the interaction is in addition to the XOR combination of the middle layer codewords. Using this prediction we focus our attention on the inner layer codewords establish if a combination of the inner layer codes $\mathbf{c}_{01}$ and $\mathbf{c}_{02}$ appear on the middle layer. In order to do so we examine equation (3.18) and see that at positions $(1, 2)$ and $(7, 8)$ when we subtract $\hat{\mathbf{c}}_0$ and divide by two the resulting bits are 1. If this was just the XOR of the middle layer codewords this would not be true. If we analyse the inner layer codewords we see that if we take the AND combinations of the inner layer codewords with the XOR of the middle layer codewords the resulting bits for positions $(1, 2)$ and $(7, 8)$ are 1 and indeed the result is equal to $\hat{\mathbf{c}}_1$.

From this observation and our analysis of this particular scenario we can definitively say that the decoded output (3.24) is the XOR combination of the codewords from both sources on the middle layer, (3.10) and (3.11), XOR'ed with the AND combinations on the inner layer, (3.6) and (3.7). This means that we have now shown the presence of a carry or interaction term generated between layers during multilayer decoding.

For the outer layer we subtract the decoded codeword of the middle layer (3.24) from $\mathbf{y}^{'}_{1'}$ and apply the modulo 2 to produce the decoded codeword of the outer layer. As this layer is not encoded then the remaining codeword should be the codeword of the outer layer. If we apply the knowledge of the interaction term from the inner to the middle layer we can analyse what is happening on the outer layer. The decoded output is:

$$\hat{\mathbf{c}}_2 = [1,0,0,1,0,0,0,1,1,0,0,0,1,1,1,0,1,0,0,0,0,0,1,1,0,0,1,1,1,1,0,0] \tag{3.25}$$

Again we see the XOR combination of the outer and middle layer as well as the AND combinations of the middle and inner layers. The XOR operation is applied to these terms and results in (3.25). We expand on this analysis in the next Section, where the following chapter details the equations on each layer in terms of logical expressions.

Prior to this analysis the estimated expected result on each layer would be the XOR combination of the codewords on the same layer and previous and outer layers would not cause any such interaction. As demonstrated previous layers do indeed have an impact on the outer layers and this needs to be taken into account for decoding. As we will see, the following sections describe and derive expressions for this interaction for multiple layers and users.

3.6 Real only signals using unit valued channels

For this section we assume a multi-user system, unit valued real only channels with $\mathbf{x}_J \in \mathbb{R}$ and $\mathbf{c}_{LJ} \in \mathbb{R}$ and $\mathbb{F}_2$. We ignore $\mathbf{z}^{'}$ and self-interference as self-interference is zero and here we focus on decodability not error rate. Then:

$$\mathbf{y}^{'} = (a_1\mathbf{x}_1+,\dots,+a_J\mathbf{x}_J)\text{mod}2^L = (a_1\text{mod}2^L\mathbf{x}_1+,\dots,+a_J\text{mod}2^L\mathbf{x}_J)\text{mod}2^L \tag{3.26}$$

Note that the mod2^L operation means that $\mathbf{y}^{'}$ is positive. As we assume that the channel coefficients $h_1,\dots,h_J$ are unit valued the integer coefficients $a_1,\dots,a_J$ can be omitted from the expressions.

$$\mathbf{y}^{'} = (\mathbf{x}_1+,\dots,+\mathbf{x}_J)\text{mod}2^L \tag{3.27}$$

Following the operation of the multilayer decoder, shown in Figure 3.2 the received signal can be expressed as:

$$\begin{aligned} \mathbf{y}^{'}_{l'} = \frac{\mathbf{y}^{'}_{l'-1} - \hat{\mathbf{c}}_{l'-1}}{2} = \frac{\mathbf{y}^{'}_{l'-2} - \hat{\mathbf{c}}_{l'-2}}{4} - \frac{\hat{\mathbf{c}}_{l'-1}}{2} = \\ \frac{1}{2^{l'}}\left(\mathbf{y}^{'} - \sum_{l=0}^{l'-1} 2^{l}\mathbf{c}_{l}\right) = \frac{1}{2^{l'}}\left(\sum_{l=0}^{L-1} 2^{l}\mathbf{y}^{'}[l] - \sum_{l=0}^{l'-1} 2^{l}\hat{\mathbf{c}}_{l}\right) \end{aligned} \tag{3.28}$$

where $[l]$ denotes the l^{th} bit of the binary representation of $\mathbf{y}^{'}$, with [0] being the least significant bit. In Section 3.8 we provide an analysis of this statement by introducing Lemma 1 and subsequently theorem 1 for integer valued channels. We then take $\mathbf{y}^{'}_{l'}$ modulo-2 and decode to find $\hat{\mathbf{c}}_{l'}$.

$$\hat{\mathbf{c}}_{l'} = \mathbf{y}^{'}_{l'}\text{mod}2 = \left\lfloor \left(\frac{1}{2^{l'}}\left(\sum_{l=0}^{L-1} 2^{l}\mathbf{y}^{'}[l] - \sum_{l=0}^{l'-1} 2^{l}\hat{\mathbf{c}}_{l}\right)\right)\right\rfloor \text{mod}2 \tag{3.29}$$

The decoded codeword for layer $l^{'}$ is therefore given by the $l^{'}th$ bit of the vector $\mathbf{y}^{'}$. Hence:

$$\hat{\mathbf{c}}_{l'} = \mathbf{y}^{'}[l^{'}] \tag{3.30}$$

for all positive integer $l^{'}$. With exception of layer 0 which has no preceding layers:

$$\hat{\mathbf{c}}_{0} = \mathbf{y}^{'}\text{mod}2 = \left\lfloor \left(\sum_{l=0}^{L-1} 2^{l}\mathbf{y}^{'}[l]\right)\right\rfloor \text{mod}2 = \mathbf{y}^{'}[0] \tag{3.31}$$

Using (3.29), this then leads to the result:

$$\hat{\mathbf{c}}_{l'} = \left\lfloor \sum_{l=0}^{l'} 2^{l-l'}(\mathbf{c}_{l1}+,\ldots,+\mathbf{c}_{lJ})\right\rfloor \text{mod}2 \tag{3.32}$$

which defines the output codeword in terms of the input codewords, where the floor function arises from the general result that $\mathbf{y}[l] = \left\lfloor \frac{\mathbf{y}}{2^{l}} \right\rfloor \text{mod}2$. This can also be written as:

$$\hat{\mathbf{c}}_{l'} = (\mathbf{c}_{l'1}+,\ldots,+\mathbf{c}_{l'J})\text{mod}2 + \left\lfloor \sum_{l=0}^{l'-1} 2^{l-l'}(\mathbf{c}_{l1}+,\ldots,+\mathbf{c}_{lJ})\right\rfloor \text{mod}2 \tag{3.33}$$

which we can then write as:

$$\hat{\mathbf{c}}_{l'} = (\mathbf{s}_{l'} + \mathbf{k}_{l'-1})\text{mod}2 \tag{3.34}$$

The term $\mathbf{s}_{l'} = (\mathbf{c}_{l1}+,\dots,+\mathbf{c}_{lJ})\text{mod}2$ is the term one may expect to find in the estimated codeword on this layer (the modulo-2 sum of the codewords from that layer). The $\mathbf{k}_{l'-1}$ is an interaction term occurring between the layers, where this is essentially the 'carry' term from the sum on the previous layer. This term must be taken into account when decoding $\hat{\mathbf{c}}_{l'}$ and when considering the expected output at the decoder. Now we can further expand:

$$\begin{aligned}\mathbf{k}_{l'-1} &= \left\lfloor \sum_{l'=0}^{l'-1} 2^{l-l'}(\mathbf{c}_{l1}+,\dots,+\mathbf{c}_{lJ}) \right\rfloor = \left\lfloor \frac{1}{2}(\sum_{l=0}^{l'-1}(\mathbf{c}_{l1}+,\dots,+\mathbf{c}_{lJ}) + \sum_{l'=0}^{l'-2} 2^{l-l'-1}(\mathbf{c}_{l1}+,\dots,+\mathbf{c}_{lJ}) \right\rfloor \\ &= \left\lfloor \frac{1}{2}(\mathbf{s}_{l'-1} + \mathbf{k}_{l'-2}) \right\rfloor \text{mod}2\end{aligned} \tag{3.35}$$

This section highlights the special case of unit valued channels and real only signals. For the general case, this is investigated in Section 3.8 and 3.9 for real valued channels and complex valued channels respectively.

3.7 Complex signals using unit valued channels

We can now straightforwardly expand to the complex case where $\mathbf{x}_l$ is complex, expressed as: $\mathbf{x}_l = \mathbf{x}^{\mathfrak{R}} + \mathbf{x}^{\mathfrak{I}}$, and the channel coefficients $h_1, \dots, h_J$ remain unit valued. In order to generate complex lattice codes $\mathbf{c}_l$ we apply 'Lifted Construction D' from [1], introduced in subsection 2.3.2. For the coarse lattice Λ' lifting achieves $\Lambda' = \Lambda'_C + \mathbb{i}\Lambda'_C$ and the fine lattice $\Lambda = \Lambda_F + \mathbb{i}\Lambda_F$. This results in a nested $\mathbf{Z}[i]$ lattice. This means we can express the real and imaginary parts as codewords C: $\mathbf{c}^I$,$\mathbf{c}^Q \in C$. Then, again ignoring noise, interference and omitting unit valued integer coefficients, the received signal:

$$\mathbf{y}' = (\mathbf{x}_1+,\dots,+\mathbf{x}_J)\text{mod}2^L = \Big(\Big((\mathbf{x}_1^{\mathfrak{R}} + \mathbb{i}\mathbf{x}_1^{\mathfrak{I}})\text{mod}2^L+,\dots,+(\mathbf{x}_J^{\mathfrak{R}} + \mathbb{i}\mathbf{x}_J^{\mathfrak{I}})\Big)\text{mod}2^L\Big)\text{mod}2^L \tag{3.36}$$

We therefore may rewrite:

$$\begin{aligned}\mathbf{y}^{'}_{l'} &= \frac{\mathbf{y}^{'}_{l'-1}-\hat{\mathbf{c}}_{l'-1}}{2} = \frac{1}{2^{l'}}\left(\mathbf{y}^{'} - \sum_{l=0}^{l^{'}-1} 2^{l}\hat{\mathbf{c}}_{l}\right)\\ &= \frac{1}{2^{l'}}\left(\left(\mathbf{y}^{'\Re}+\mathbb{i}\mathbf{y}^{'\Im}\right) - \sum_{l=0}^{l^{'}-1} 2^{l}\left(\hat{\mathbf{c}}^{I}_{l}+\mathbb{i}\hat{\mathbf{c}}^{Q}_{l}\right)\right)\\ &= \frac{1}{2^{l'}}\left(\sum_{l=0}^{L-1} 2^{l}\left(\mathbf{y}^{'\Re}[l]+\mathbb{i}\mathbf{y}^{'\Im}[l]\right) - \sum_{l=0}^{l^{'}-1} 2^{l}\left(\hat{\mathbf{c}}^{I}_{l}+\mathbb{i}\hat{\mathbf{c}}^{Q}_{l}\right)\right)\end{aligned} \tag{3.37}$$

writing $\mathbf{y}^{'\{\Re,\Im\}} = \sum_{l=0}^{L-1}\left(2^{l}\mathbf{y}^{'\{\Re,\Im\}}[l]\right)$. Then:

$$\hat{\mathbf{c}}^{\{\Re,\Im\}}_{l'} = \mathbf{y}^{'\{\Re,\Im\}}[l^{'}] \tag{3.38}$$

With the exception of layer 0, as demonstrated in the previous subsection and expanded to:

$$\hat{\mathbf{c}}_{0} = (\mathbf{y}^{'\Re}+\mathbb{i}\mathbf{y}^{'\Im})\mathrm{mod}2 = \left(\sum_{l=0}^{L-1} 2^{l}(\mathbf{y}^{'\Re}[l]+\mathbb{i}\mathbf{y}^{'\Im}[l]\right) = \mathbf{y}^{'\Re}[0]+\mathbb{i}\mathbf{y}^{'\Im}[0] \tag{3.39}$$

As in the real only case neglecting noise and interference:

$$\hat{\mathbf{c}}_{l'} = \hat{\mathbf{c}}^{I}_{l'}+\mathbb{i}\hat{\mathbf{c}}^{Q}_{l'} = \left\lfloor \sum_{l=0}^{l^{'}} 2^{l-l^{'}}\left((\mathbf{c}^{I}_{l1}+\mathbb{i}\mathbf{c}^{Q}_{l1})+,\ldots,+(\mathbf{c}^{I}_{lJ}+\mathbb{i}\mathbf{c}_{lJ})\right)\right\rfloor \mathrm{mod}2 \tag{3.40}$$

which can also be written as:

$$\hat{\mathbf{c}}_{l'} = \left((\mathbf{c}^{I}_{l'1}+\mathbb{i}\mathbf{c}^{Q}_{l'1})+,\ldots,+(\mathbf{c}^{I}_{l'J}+\mathbb{i}\mathbf{c}^{Q}_{l'J})\right)\mathrm{mod}2 + \left\lfloor \sum_{l=0}^{l^{'}-1} 2^{l-l^{'}}(\mathbf{c}^{I}_{l1}+\mathbb{i}\mathbf{c}^{Q}_{l1})+,\ldots,+(\mathbf{c}^{I}_{lJ}+\mathbb{i}\mathbf{c}^{Q}_{lJ})\right\rfloor \mathrm{mod}2 \tag{3.41}$$

as in (3.34), then:

$$\hat{\mathbf{c}}_{l'} = (\mathbf{s}_{l'}+\mathbf{k}_{l'-1})\mathrm{mod}2 \tag{3.42}$$

where the ‘carry’ term can be given as:

$$\mathbf{k}_{l'-1} = \left\lfloor \sum_{l'=0}^{l'-1} 2^{l-l'} \Big((\mathbf{c}_{l1}^{I} + \mathbb{i}\mathbf{c}_{l1}^{Q}) +, \dots, + (\mathbf{c}_{lJ}^{I} + \mathbb{i}\mathbf{c}_{lJ}^{Q}) \Big) \right\rfloor = \Bigg\lfloor \frac{1}{2} \Bigg(\sum_{l=0}^{l'-1} \Big((\mathbf{c}_{l1}^{I} + \mathbb{i}\mathbf{c}_{l1}^{Q}) +, \dots, + (\mathbf{c}_{lJ}^{I} + \mathbb{i}\mathbf{c}_{lJ}^{Q}) \Big)$$

$$+ \sum_{l'=0}^{l'-2} 2^{l-l'-1} \Big((\mathbf{c}_{l1}^{I} + \mathbb{i}\mathbf{c}_{l1}^{Q}) +, \dots, + (\mathbf{c}_{lJ}^{I} + \mathbb{i}\mathbf{c}_{lJ}^{Q}) \Big) \Bigg) \Bigg\rfloor = \left\lfloor \frac{1}{2} (\mathbf{s}_{l'-1} + \mathbf{k}_{l'-2}) \right\rfloor \text{mod} 2 \tag{3.43}$$

As in the previous section we demonstrate a special case of the expected decoded code on each layer, deriving the mathematical expression of the interaction term. In the next section we move to the general case.

3.8 Integer valued channels

We now progress to describe layered decoding for the general case using integer valued channels and binary codes. Then the received signal is:

$$\mathbf{y}' = (a_1 \mathbf{x}_1 +, \dots, + a_J \mathbf{x}_J) \text{mod} 2^L = (a_1 \text{mod} 2^L \mathbf{x}_1 +, \dots, + a_J \text{mod} 2^L \mathbf{x}_J) \text{mod} 2^L \tag{3.44}$$

We now seek to relate the estimated codewords on each layer of the decoder to the transmitted codewords on each layer. We consider the effects of integer channels within this and therefore introduce our first main result by means of Lemma 1. This is in a sense a statement of the obvious: the code word output on the l'th layer is bit l' of the scaled version of the received signal.

Lemma 1. *Neglecting the effect of noise and interference, the decoded codeword for layer l' is given by the l'^{th} bit of the vector $\mathbf{y}'$, that is:*

$$\hat{\mathbf{c}}_l' = \mathbf{y}'[l'] \tag{3.45}$$

Proof. By induction, if we assume that $\hat{\mathbf{c}}_{l'} = \mathbf{y}'[l], \forall l < l'$, then:

$$\hat{\mathbf{c}}_{l'} = \left(\frac{1}{2^{l'}} \left(\sum_{l=0}^{L-1} 2^{l} \mathbf{y}'[l] \right) \right) \text{mod} 2 = \left(\sum_{l=l'}^{L-1} 2^{l-l'} \mathbf{y}'[l] \right) \text{mod} 2 = \mathbf{y}'[l'] \tag{3.46}$$

however, as stated previously, with exception to layer 0 which has no preceding layers. $\hat{\mathbf{c}}_0$

is therefore again expressed as (3.31) and hence by induction, $\hat{\mathbf{c}}_{l'} = \mathbf{y}^{'}[l^{'}]$ for all positive integer $l^{'}$. This then leads to the first main result, which defines the output codeword in terms of the input codewords based on the integer valued channel coefficients. □

Theorem 1. *Again neglecting the effect of noise and interference:*

$$\hat{\mathbf{c}}_{l'} = \Bigg\lfloor \sum_{m^{'}=0}^{l^{'}} 2^{m^{'}-l^{'}} \sum_{l=0}^{m^{'}} \Big((a_1[m^{'}-l]\mathbf{c}_{l1})+,\ldots,+(a_J[m^{'}-l]\mathbf{c}_{lJ})\Big)\Bigg\rfloor \text{mod}2 \tag{3.47}$$

Proof.

$$\begin{aligned}\mathbf{y}^{'} &= \sum_{l=0}^{L-1}(a_1 2^l \mathbf{c}_{l1})+,\ldots,+(a_J 2^l \mathbf{c}_{lJ}) = \sum_{l=0}^{L-1}(\sum_{m=0}^{L-1} 2^m a_1[m]2^l\mathbf{c}_{l1}+,\ldots,+\sum_{m=0}^{L-1} 2^m a_J[m]2^l\mathbf{c}_{lJ})\\ &= \sum_{l=0}^{L-1}(\sum_{m=0}^{L-1} a_1[m]2^{l+m}\mathbf{c}_{l1}+,\ldots,+\sum_{m=0}^{L-1} a_J[m]2^{l+m}\mathbf{c}_{lJ})\\ &= \sum_{m^{'}=0}^{L-1} 2^{m^{'}} \sum_{l=0}^{m^{'}}(a_1[m^{'}-l]\mathbf{c}_{l1}+,\ldots,+a_J[m^{'}-l]\mathbf{c}_{lJ})\end{aligned} \tag{3.48}$$

□

To find the $l^{'}$th bit $p[l^{'}]$ of a positive integer p take $\left\lfloor \frac{p}{2^{l'}} \right\rfloor$mod2. Then:

$$\mathbf{y}^{'}[l^{'}] = \hat{\mathbf{c}}_{l'} = \left\lfloor \frac{\mathbf{y}^{'}}{2^{l'}} \right\rfloor \text{mod}2 = \Bigg\lfloor \sum_{m^{'}=0}^{l^{'}} 2^{m^{'}-l^{'}} \sum_{l=0}^{m^{'}}(a_1[m^{'}-l]\mathbf{c}_{l1}+,\ldots,+a_J[m^{'}-l]\mathbf{c}_{lJ})\Bigg\rfloor \text{mod}2 \tag{3.49}$$

As with the unit valued case we can also restate this result as:

$$\begin{aligned}\hat{\mathbf{c}}_{l'} = \Bigg(&\sum_{l=0}^{l^{'}}(a_1[l^{'}-l]\mathbf{c}_{l1}+,\ldots,+a_J[l^{'}-l]\mathbf{c}_{lJ})\\ &+\Bigg\lfloor \sum_{m^{'}=0}^{l^{'}-1} 2^{m^{'}-l^{'}} \sum_{l=0}^{m^{'}}(a_1[m^{'}-l]\mathbf{c}_{l1}+,\ldots,+a_J[m^{'}-l]\mathbf{c}_{lJ})\Bigg\rfloor\Bigg)\text{mod}2\end{aligned} \tag{3.50}$$

which we can then re-write as:

$$\hat{\mathbf{c}}_{l'} = (\mathbf{s}_{l'} + \mathbf{k}_{l'-1})\text{mod}2 \tag{3.51}$$

The term $\mathbf{s}_{l'}$ is now the modulo-2 sum of the codewords from that layer weighted by the corresponding bits from the binary representation of the integer channel coefficients. As in (3.34) $\mathbf{k}_{l'-1}$ remains as a 'carry' term occurring between the layers. This must be taken into account both when decoding $\mathbf{c}_{l'}$ and when considering the expected output of the decoder. Now we can further expand the 'carry' term:

$$\begin{aligned}\mathbf{k}_{l'-1} &= \left\lfloor \sum_{m'=0}^{l'-1} 2^{m'-l'} \sum_{l=0}^{m'} (a_1[m'-l]\mathbf{c}_{l1}+,\ldots,+a_J[m'-l]\mathbf{c}_{lJ}) \right\rfloor \\ &= \left\lfloor \frac{1}{2}(\sum_{l=0}^{l'-1}(a_1[l'-1-l]\mathbf{c}_{l1}+,\ldots,+a_J[l'-1-l]\mathbf{c}_{lJ}) \right. \\ &\left. + \sum_{m'=0}^{l'-2} 2^{m'-l'-1} \sum_{l=0}^{m'} (a_1[m'-l]\mathbf{c}_{l1}+,\ldots,+a_J[m'-l]\mathbf{c}_{lJ})) \right\rfloor = \left\lfloor \frac{1}{2}(\mathbf{s}_{l'-1}+\mathbf{k}_{l'-2}) \right\rfloor\end{aligned} \tag{3.52}$$

To understand the implications of this we first define Lemma 2:

Lemma 2. *Suppose that* $q_1, q_2, \ldots, q_J$ *are binary variables, i.e.* $q_1, q_2, \ldots, q_J \in 0, 1$ *Then:*

$$\left\lfloor \frac{1}{2} \sum_{i=1}^{j} q_i \right\rfloor \mathrm{mod} 2 = \left(\sum_{i=2}^{j} \sum_{f=1}^{i-1} q_i \cdot q_f \right) \mathrm{mod} 2 \tag{3.53}$$

where $\cdot$ *denotes product, which for binary variables is the same as the logical AND operation on Boolean variables. In other words, the floor taken modulo 2 of half the sum of the variables is the modulo 2 sum of all distinct pairwise products of the variables, that is it is the XOR combination of all AND combinations of the variables.*

Proof. Let the number of 1's among the q_i's be p. The result on the left hand side (lhs) of the equation is the integer quotient of $p/2$ taken modulo 2: $quot(p/2)\mathrm{mod}2$ The result on the right hand side (rhs) is the number of combinations containing 2 1's out of the p taken modulo 2 i.e. $\overset{p}{\underset{2}{C}}\mathrm{mod}2 = \left(\frac{p(p-1)}{2}\right)\mathrm{mod}2$. Now, $\frac{p(p-1)}{2}$ is even provided $p/2$ or $(p-1)/2$ is even, i.e if p is a multiple of 4 or one more than a multiple of 4, and odd otherwise. The same applies to $quot(p/2)$ which verifies the Lemma. □

This leads to our second, and most important main result which arises because the operation of the layered decoder requires that $\hat{\mathbf{c}}_{l'}$ is a codeword of C_l so that the decoder for that layer can decode it.

Theorem 2. *The layered decoder can decode a multilayer lattice using Construction by Code Formula based on binary codes if and only if:*

1. *The codes used on all layers $C_l, l = 0 \dots L-1$ are linear;*

2. *The codes are nested: that is, codes on inner layers are subcodes of those on outer layers: $C_l \subseteq C_{l'}, l \leqslant l'$;*

3. *The pairwise AND combinations of all codewords of C_l must be codewords of the code on the next layer, C_{l+1}.*

Proof. To establish (1) as a necessary condition for decodability consider equation (3.50). The expression for $\hat{\mathbf{c}}_{l'}$ is a modulo 2 sum which contains the term $\sum_{l=0}^{l'}(a_1[l'-l]\mathbf{c}_{l1}+, \dots, +a_J[l'-l]\mathbf{c}_{lJ})\text{mod}2$. This in in turn contains the modulo-2 sum of $a_J[0]\mathbf{c}_{l'J}$, in which $a_J[0] \in [0,1]$. Since it is possible that all the other terms in (3.50) are zero, for decodability in all cases this term must be a codeword. By definition it is a codeword of $C_{l'}$ if and only if $C_{l'}$ is linear.

To establish (2) as a necessary condition, note that the expression for $\hat{\mathbf{c}}_{l'}$ also contains terms $a_1[m]\mathbf{c}_{(l'-m)1}+, \dots, +a_J[m]\mathbf{c}_{(l'-m)J}, m = 1 \dots l$ and again $a_J[m] \in [0,1], m = 1 \dots l$. Again these are necessarily codewords of $C_{l'}$ if and only if all codewords of $C_{l'-m}, m = 1 \dots l'$ are also codewords of C_l and these codes also are linear.

To establish (3) as a necessary condition, consider (3.51). This contains $\mathbf{k}_{l'-1}$, which therefore must be a codeword of $C_{l'}$. Then from (3.52) $\mathbf{k}_{l'-1} = \frac{1}{2}(\mathbf{s}_{l'-1} + \mathbf{k}_{l'-2})$ it is clear that for decodability on layer $l' - 1$, $\mathbf{k}_{l'-2}$ must also be a codeword of $C_{l'-1}$. Thus from Lemma 2 the AND combination of two codewords of $C_{l'-1}$ must be a codeword of $C_{l'}$, and indeed this applies to any two codewords of $C_{l'-1}$. To establish these as sufficient conditions, start on layer 0, where the conditions certainly ensure decodability. On layer 1 we have that $\hat{\mathbf{c}}_1 = (\mathbf{s}_1 + \mathbf{k}_0)\text{mod}2$, and $\mathbf{k}_0 = \lfloor \frac{\mathbf{s}_0}{2} \rfloor$. Then condition (3) ensures that $\mathbf{k}_0$ is a codeword of C_1, and conditions (1) and (2) ensure that $\hat{\mathbf{c}}_1$ also is. This process can be continued up to layer l', hence showing that the conditions ensure decodability at any layer. □

Note that conditions (1) and (2) are already required for the codes used on each layer in Construction D. Theorem 2 shows that if the layered decoder is to be used, condition (3) of the theorem must be added to these. It is not in fact a particularly onerous condition, since in the example scheme in Section 3.9 the outer layer is left uncoded, and of course all possible words are thus codewords. Also a repetition code is employed on layer 0, and the condition requires only that the all 1s codeword is included in the layer 1 code which is also required as a result of condition (2). We demonstrate an example scheme in Section 4.4 with three layers as in practice it would be rare to use more than 3 layers in a Construction D scheme.

3.9 Complex Integer valued channels

Finally, we extend the results of the previous subsection to complex integer coefficient channels and assume that $\mathbf{x}_J$ and $\mathbf{c}_J$ are complex and binary. As described in Section 3.2, the received signal $\mathbf{y}$ is multiplied by α and then taken modulo-2^L to achieve (3.3). In this Section we consider $h_1, \ldots, h_J$ and α to be complex therefore: $a_J = a_J^{\mathfrak{R}} + \mathbb{i} a_J^{\mathfrak{I}}$ and we can rewrite $\mathbf{y}^{'}$ as:

$$\begin{aligned}
\mathbf{y}^{'} =& (a_1\mathbf{x}_1+, \ldots, +a_J\mathbf{x}_J)\text{mod}2^L = \Bigg(\Big((a_1^{\mathfrak{R}} + \mathbb{i}a_1^{\mathfrak{I}})(\mathbf{x}_1^{\mathfrak{R}} + \mathbb{i}\mathbf{x}_1^{\mathfrak{I}})\Big)\text{mod}2^L \\
&+, \ldots, +\Big((a_J^{\mathfrak{R}} + \mathbb{i}a_J^{\mathfrak{I}})(\mathbf{x}_J^{\mathfrak{R}} + \mathbb{i}\mathbf{x}_J^{\mathfrak{I}})\Big)\text{mod}2^L\Bigg)\text{mod}2^L \\
&= \Bigg(\Big((a_1^{\mathfrak{R}}\mathbf{x}_1^{\mathfrak{R}} - a_1^{\mathfrak{I}}\mathbf{x}_1^{\mathfrak{I}}) + \mathbb{i}(a_1^{\mathfrak{I}}\mathbf{x}_1^{\mathfrak{R}} + a_1^{\mathfrak{R}}\mathbf{x}_1^{\mathfrak{I}})\Big)\text{mod}2^L \\
&+, \ldots, +\Big((a_J^{\mathfrak{R}}\mathbf{x}_J^{\mathfrak{R}} - a_J^{\mathfrak{I}}\mathbf{x}_J^{\mathfrak{I}}) + \mathbb{i}(a_J^{\mathfrak{I}}\mathbf{x}_J^{\mathfrak{R}} + a_J^{\mathfrak{R}}\mathbf{x}_J^{\mathfrak{I}})\Big)\text{mod}2^L\Bigg)\text{mod}2^L \\
&= \Bigg((a_1^{\mathfrak{R}}\text{mod}2^L)\mathbf{x}_1^{\mathfrak{R}} + \Big((-a_1^{\mathfrak{I}})\text{mod}2^L\Big)\mathbf{x}_1^{\mathfrak{I}} \\
&+, \ldots, +(a_J^{\mathfrak{R}}\text{mod}2^L)\mathbf{x}_J^{\mathfrak{R}} + \Big((-a_J^{\mathfrak{I}})\text{mod}2^L\Big)\mathbf{x}_J^{\mathfrak{I}}\Bigg)\text{mod}2^L + \mathbb{i}\Bigg((a_1^{\mathfrak{I}}\text{mod}2^L)\mathbf{x}_1^{\mathfrak{R}} \\
&+ (a_1^{\mathfrak{R}}\text{mod}2^L)\mathbf{x}_1^{\mathfrak{I}}+, \ldots, +(a_J^{\mathfrak{I}}\text{mod}2^L)\mathbf{x}_J^{\mathfrak{R}} + (a_J^{\mathfrak{R}}\text{mod}2^L)\mathbf{x}_J^{\mathfrak{I}}\Bigg)\text{mod}2^L
\end{aligned} \tag{3.54}$$

For notational convenience we will write $(-a_J^{\mathfrak{I}})$ as $a_J^{-\mathfrak{I}}$ from this point forward. we may re-write:

$$\begin{aligned}
\mathbf{y}_{l'}^{'} = \frac{\mathbf{y}_{l'-1}^{'} - \mathbf{c}_{l'-1}\hat{}}{2} &= \frac{1}{2^{l'}}\Bigg(\mathbf{y}^{'} - \sum_{l=0}^{l'-1} 2^l\hat{\mathbf{c}}_l\Bigg) = \frac{1}{2^{l'}}\Bigg((\mathbf{y}^{'\mathfrak{R}} + \mathbb{i}\mathbf{y}^{'\mathfrak{I}}) - \sum_{l=0}^{l'-1} 2^l(\hat{\mathbf{c}}_l^I + \mathbb{i}\hat{\mathbf{c}}_l^Q)\Bigg) \\
&= \frac{1}{2^{l'}}\Bigg(\sum_{l=0}^{L-1} 2^l(\mathbf{y}^{'\mathfrak{R}}[l] + \mathbb{i}\mathbf{y}^{'\mathfrak{I}}[l]) - \sum_{l=0}^{l'-1} 2^l(\hat{\mathbf{c}}_l^I + \mathbb{i}\hat{\mathbf{c}}_l^Q)\Bigg)
\end{aligned} \tag{3.55}$$

where we can write $\mathbf{y}^{'\{\mathfrak{R},\mathfrak{I}\}} = \sum_{l=0}^{L-1}\left(2^l \mathbf{y}^{'\{\mathfrak{R},\mathfrak{I}\}}[l]\right)$

Lemma 3. *As in the real case, neglecting noise and interference, the decoded codeword for layer $l^{'}$ is given by the $l^{'th}$ bit of the vector $\mathbf{y}^{'}$. In the complex case the decoded codeword for layer $l^{'}$ is given by the $l^{'th}$ bit of the real and imaginary parts of $\mathbf{y}^{'}$. Denoted as:*

$$\hat{\mathbf{c}}_{l^{'}}^{\{\mathfrak{R},\mathfrak{I}\}} = \mathbf{y}^{'\{\mathfrak{R},\mathfrak{I}\}}[l^{'}] \tag{3.56}$$

Proof. Using the above equation:

$$\mathbf{y}_{l^{'}}^{'\mathfrak{R}} + \mathbb{i}\mathbf{y}_{l^{'}}^{'\mathfrak{I}} = \frac{1}{2^{l^{'}}}\left(\sum_{l=0}^{L-1} 2^l(\mathbf{y}^{'\mathfrak{R}}[l] + \mathbb{i}\mathbf{y}^{'\mathfrak{I}}[l]) - \sum_{l=0}^{l^{'}-1} 2^l(\hat{\mathbf{c}}_l^I + \mathbb{i}\hat{\mathbf{c}}_l^Q)\right) \tag{3.57}$$

Extracting real and imaginary parts leads to:

$$\mathbf{y}_{l^{'}}^{'\{\mathfrak{R},\mathfrak{I}\}} = \frac{1}{2^{l^{'}}}\left(\sum_{l=0}^{L-1} 2^l \mathbf{y}^{'\{\mathfrak{R},\mathfrak{I}\}}[l] - \sum_{l=0}^{l^{'}-1} 2^l \hat{\mathbf{c}}_l^{\{I,Q\}}\right) \tag{3.58}$$

Then the result follows by following Lemma 1, treating the real and imaginary parts separately. □

Theorem 3. *As in theorem 1 neglecting noise and interference:*

$$\begin{aligned}\hat{\mathbf{c}}_{l'} = \hat{\mathbf{c}}_{l'}^I + \mathbb{i}\hat{\mathbf{c}}_{l'}^Q = \Bigg\lfloor \sum_{m^{'}=0}^{l^{'}} 2^{m^{'}-l^{'}} \sum_{l=0}^{m^{'}} \Big(& (a_1^{\mathfrak{R}}[m^{'}-l]\mathbf{c}_{l1}^I + a_1^{-\mathfrak{I}}[m^{'}-l]\mathbf{c}_{l1}^Q +, \dots, + a_J^{\mathfrak{R}}[m^{'}-l]\mathbf{c}_{lJ}^I \\ & + a_J^{-\mathfrak{I}}[m^{'}-l]\mathbf{c}_{lJ}^Q) + \mathbb{i}(a_1^{\mathfrak{R}}[m^{'}-l]\mathbf{c}_{l1}^Q + a_1^{\mathfrak{I}}[m^{'}-l]\mathbf{c}_{l1}^I \\ & +, \dots, + a_J^{\mathfrak{R}}[m^{'}-l]\mathbf{c}_{lJ}^Q + a_J^{\mathfrak{I}}[m^{'}-l]\mathbf{c}_{lJ}^I)\Big)\Bigg\rfloor \text{mod} 2\end{aligned} \tag{3.59}$$

where $a_J^{\{\mathfrak{R},\mathfrak{I}\}}[m]$ denotes the m^{th} bit binary representation of the corresponding integer channel coefficient.

Proof. Generalising the proof of Theorem 1.

$$\begin{aligned}\mathbf{y}^{'}[l^{'}] &= \hat{\mathbf{c}}_{l'} = \hat{\mathbf{c}}^{I}_{l'} + \mathbb{i}\hat{\mathbf{c}}^{Q}_{l'} = \left\lfloor \frac{\mathbf{y}^{'}}{2^{l'}} \right\rfloor \mathrm{mod}2 \\ &= \Bigg\lfloor \frac{1}{2^{l'}} \Bigg((a^{\mathfrak{R}}_{1}\mathrm{mod}2^{L})\mathbf{x}^{\mathfrak{R}}_{1} + (a^{-\mathfrak{I}}_{1}\mathrm{mod}2^{L})\mathbf{x}^{\mathfrak{I}}_{1}+, \ldots, +(a^{\mathfrak{R}}_{J}\mathrm{mod}2^{L})\mathbf{x}^{\mathfrak{R}}_{J} \\ &+ \Big((a^{-\mathfrak{I}}_{J})\mathrm{mod}2^{L} \Big) \mathbf{x}^{\mathfrak{I}}_{J} \Bigg) \mathrm{mod}2^{L} + \mathbb{i}\Big((a^{\mathfrak{I}}_{1}\mathrm{mod}2^{L})\mathbf{x}^{\mathfrak{R}}_{1} \\ &+ (a^{\mathfrak{R}}_{1}\mathrm{mod}2^{L})\mathbf{x}^{\mathfrak{I}}_{1}+, \ldots, +(a^{\mathfrak{I}}_{J}\mathrm{mod}2^{L})\mathbf{x}^{\mathfrak{R}}_{J} + (a^{\mathfrak{R}}_{J}\mathrm{mod}2^{L})\mathbf{x}^{\mathfrak{I}}_{J} \Big) \mathrm{mod}2^{L} \Bigg\rfloor \mathrm{mod}2\end{aligned} \tag{3.60}$$

Now writing $a^{\{\mathfrak{R},\mathfrak{I}\}}_{\{1,\ldots,J\}} = \sum_{m=0}^{L-1} 2^{m} a^{\{\mathfrak{R},\mathfrak{I}\}}_{\{1,\ldots,J\}}[m]$ terms of the form $(a^{\{\mathfrak{R},\mathfrak{I}\}}_{\{1,\ldots,J\}}\mathrm{mod}2^{L})\mathbf{x}^{\{\mathfrak{R},\mathfrak{I}\}}_{\{1,\ldots,J\}}$ can be written:

$$\begin{aligned}&\Big(a^{\{\mathfrak{R},\mathfrak{I}\}}_{\{1,\ldots,J\}}\mathrm{mod}2^{L} \Big) \mathbf{x}^{\{\mathfrak{R},\mathfrak{I}\}}_{\{1,\ldots,J\}} = \sum_{m=0}^{L-1} 2^{m} a^{\{\mathfrak{R},\mathfrak{I}\}}_{\{1,\ldots,J\}}[m] \sum_{l=0}^{L-1} 2^{l} \mathbf{c}^{\{I,Q\}}_{l\{1,\ldots,J\}} \\ &= \sum_{m=0}^{L-1} \sum_{l=0}^{L-1} 2^{l+m} a^{\{\mathfrak{R},\mathfrak{I}\}}_{\{1,\ldots,J\}}[m] \mathbf{c}^{\{I,Q\}}_{l\{1,\ldots,J\}} = \sum_{m^{'}=0}^{L-1} 2^{m^{'}} \sum_{l=0}^{m^{'}} a^{\{\mathfrak{R},\mathfrak{I}\}}_{\{1,\ldots,J\}}[m^{'} - l] \mathbf{c}^{\{I,Q\}}_{l\{1,\ldots,J\}}\end{aligned} \tag{3.61}$$

and similarly:

$$\Big((a^{-\mathfrak{I}}_{J})\mathrm{mod}2^{L} \Big) \mathbf{x}^{\mathfrak{I}}_{\{1,\ldots,J\}} = \sum_{m^{'}=0}^{L-1} 2^{m^{'}} \sum_{l=0}^{m^{'}} a^{\mathfrak{I}}_{J}[m^{'} - l] \mathbf{c}^{Q}_{lJ} \tag{3.62}$$

substituting these terms into the above expression gives the result. □

$$
\begin{aligned}
\hat{\mathbf{c}}_{l'} = \hat{\mathbf{c}}^{I}_{l'} + \mathbb{i}\hat{\mathbf{c}}^{Q}_{l'} = \Bigg(\bigg(&\sum_{l=0}^{l'}(a_1^{\Re}[l'-l]\mathbf{c}^{I}_{l1} + a_1^{-\Im}[l'-l]\mathbf{c}^{Q}_{l1}+,\ldots,+a_J^{\Re}[l'-l]\mathbf{c}^{I}_{lJ} + a_J^{-\Im}[l'-l]\mathbf{c}^{Q}_{lJ})\bigg)\mathrm{mod}2\\
&+\bigg\lfloor\frac{1}{2}\sum_{l=0}^{l'-1}(a_1^{\Re}[l'-1-l]\mathbf{c}^{I}_{l1} + a_1^{-\Im}[l'-1-l]\mathbf{c}^{Q}_{l1}+,\ldots,+a_J^{\Re}[l'-1-l]\mathbf{c}^{I}_{lJ} + a_J^{-\Im}[l'-1-l]\mathbf{c}^{Q}_{lJ})\\
&+\frac{1}{4}\sum_{l=0}^{l'-2}(a_1^{\Re}[l'-2-l]\mathbf{c}^{I}_{l1} + a_1^{-\Im}[l'-2-l]\mathbf{c}^{Q}_{l1}+,\ldots,+a_J^{\Re}[l'-2-l]\mathbf{c}^{I}_{lJ} + a_J^{-\Im}[l'-2-l]\mathbf{c}^{Q}_{lJ}) + \ldots\\
&\frac{1}{2^{l'}}(a_1^{\Re}[0]\mathbf{c}^{I}_{l1} + a_1^{-\Im}[0]\mathbf{c}^{Q}_{l1}+,\ldots,+a_J^{\Re}[0]\mathbf{c}^{I}_{lJ} + a_J^{-\Im}[0]\mathbf{c}^{Q}_{lJ})\bigg\rfloor\mathrm{mod}2\Bigg)\mathrm{mod}2\\
&+\mathbb{i}\Bigg(\bigg(\sum_{l=0}^{l'}(a_1^{\Re}[l'-l]\mathbf{c}^{Q}_{l1} + a_1^{\Im}[l'-l]\mathbf{c}^{I}_{l1}+,\ldots,+a_J^{\Re}[l'-l]\mathbf{c}^{Q}_{lJ} + a_J^{\Im}[l'-l]\mathbf{c}^{I}_{lJ})\bigg)\mathrm{mod}2\\
&+\bigg\lfloor\frac{1}{2}\sum_{l=0}^{l'-1}(a_1^{\Re}[l'-1-l]\mathbf{c}^{Q}_{l1} + a_1^{\Im}[l'-1-l]\mathbf{c}^{I}_{l1}+,\ldots,+a_J^{\Re}[l'-1-l]\mathbf{c}^{Q}_{lJ} + a_J^{\Im}[l'-1-l]\mathbf{c}^{I}_{lJ})\\
&+\frac{1}{4}\sum_{l=0}^{l'-2}(a_1^{\Re}[l'-2-l]\mathbf{c}^{Q}_{l1} + a_1^{\Im}[l'-2-l]\mathbf{c}^{I}_{l1}+,\ldots,+a_J^{\Re}[l'-2-l]\mathbf{c}^{Q}_{lJ} + a_J^{\Im}[l'-2-l]\mathbf{c}^{I}_{lJ}) + \ldots\\
&\frac{1}{2^{l'}}(a_1^{\Re}[0]\mathbf{c}^{Q}_{l1} + a_1^{\Im}[0]\mathbf{c}^{I}_{l1}+,\ldots,+a_J^{\Re}[0]\mathbf{c}^{Q}_{lJ} + a_J^{\Im}[0]\mathbf{c}^{I}_{lJ})\bigg\rfloor\mathrm{mod}2\Bigg)\mathrm{mod}2
\end{aligned}
\tag{3.63}
$$

It is straightforward to show that Theorem 2 must also apply to the codes used for the in-phase and quadrature components.

The diagrams shown in Figures 3.4 and 3.5 visualise the process of determining the occurrence of the 'carry' term for each code layer, shown for two users and three code layers. This is detailed for the In-phase and Quadrature component where $\bar{a}_j[\{0,1,2\}]\mathbb{i}$ denotes the two's complement of the integer coefficient of each significant bit. As can be observed from our analysis in this chapter, for each layer the number of 'carry' term(s) increase meaning that the complexity of this signal increases with the number of code layers. In practice we would not see the number of code layers be higher than three and therefore evaluate the expression up to three layers.

3.10 Summary

This chapter has derived the mathematical expressions which describe the scaled received signal $\mathbf{y}'_{l'}$ at each layer, the expected decoded output $\hat{\mathbf{c}}_{l'}$ at each code layer including the interaction term found to occur between layers. In Section 3.1 the chapter was introduced and in Section 3.2 an outline of the C&F system model was provided. The structure of multilayer decoding was described in Section 3.3 followed by a demonstration of error floor between layers when the interaction signal is not considered in Section 3.4. The interaction between code layers is given in Section 3.5. The special case implementing unit valued channels and real signals is given in Section 3.6 before expanding this work to complex signals in Section 3.7. In Section 3.8 derived expressions using integer channel coefficients are given where this is expanded to using complex signals in Section 3.9. This work progresses to describe the general case and can be implemented for any integer valued channel using complex or real signals for multiple layers and users.

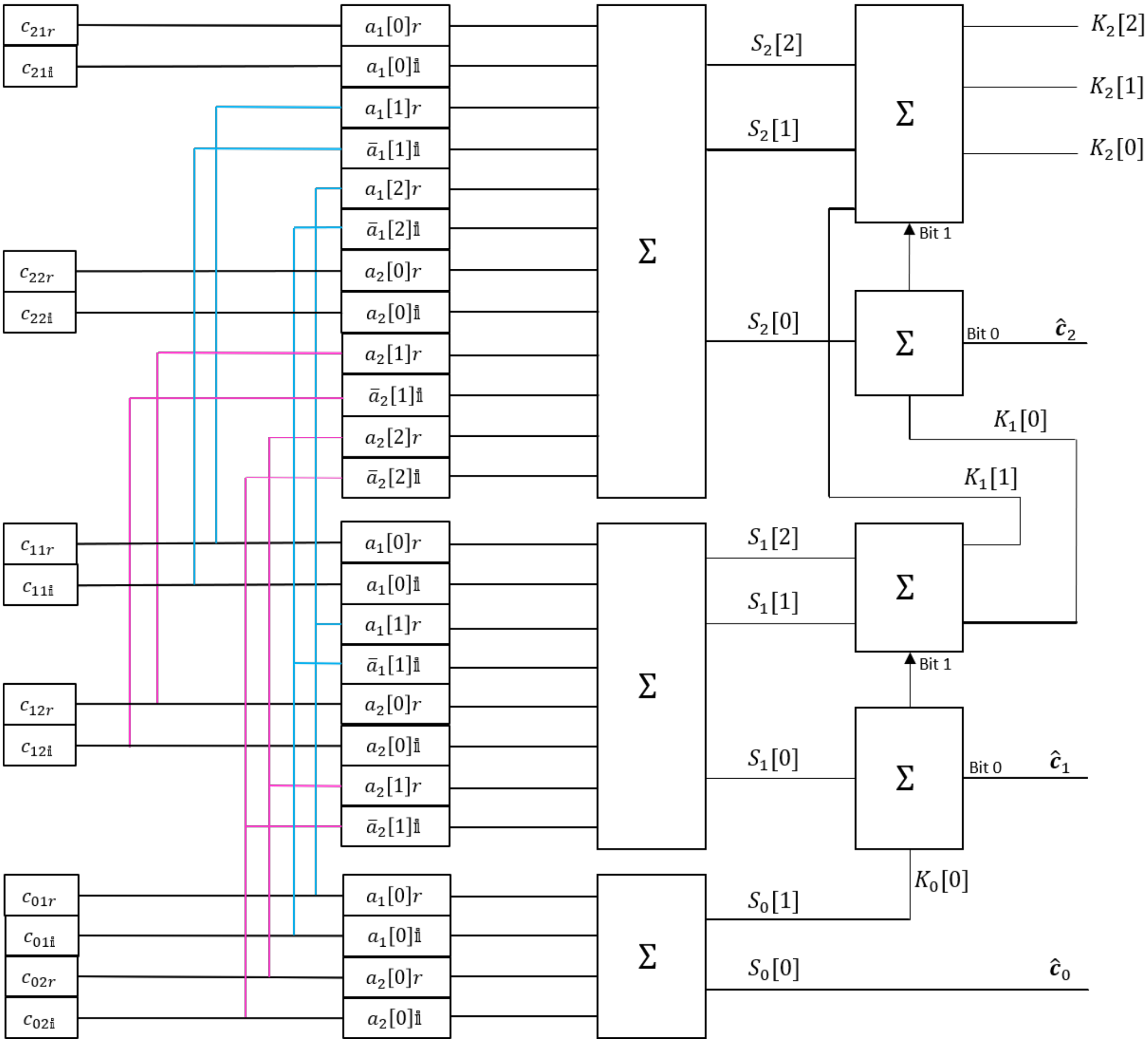


Figure 3.4: Full adder - In-phase component for complex signals and complex integer coefficients

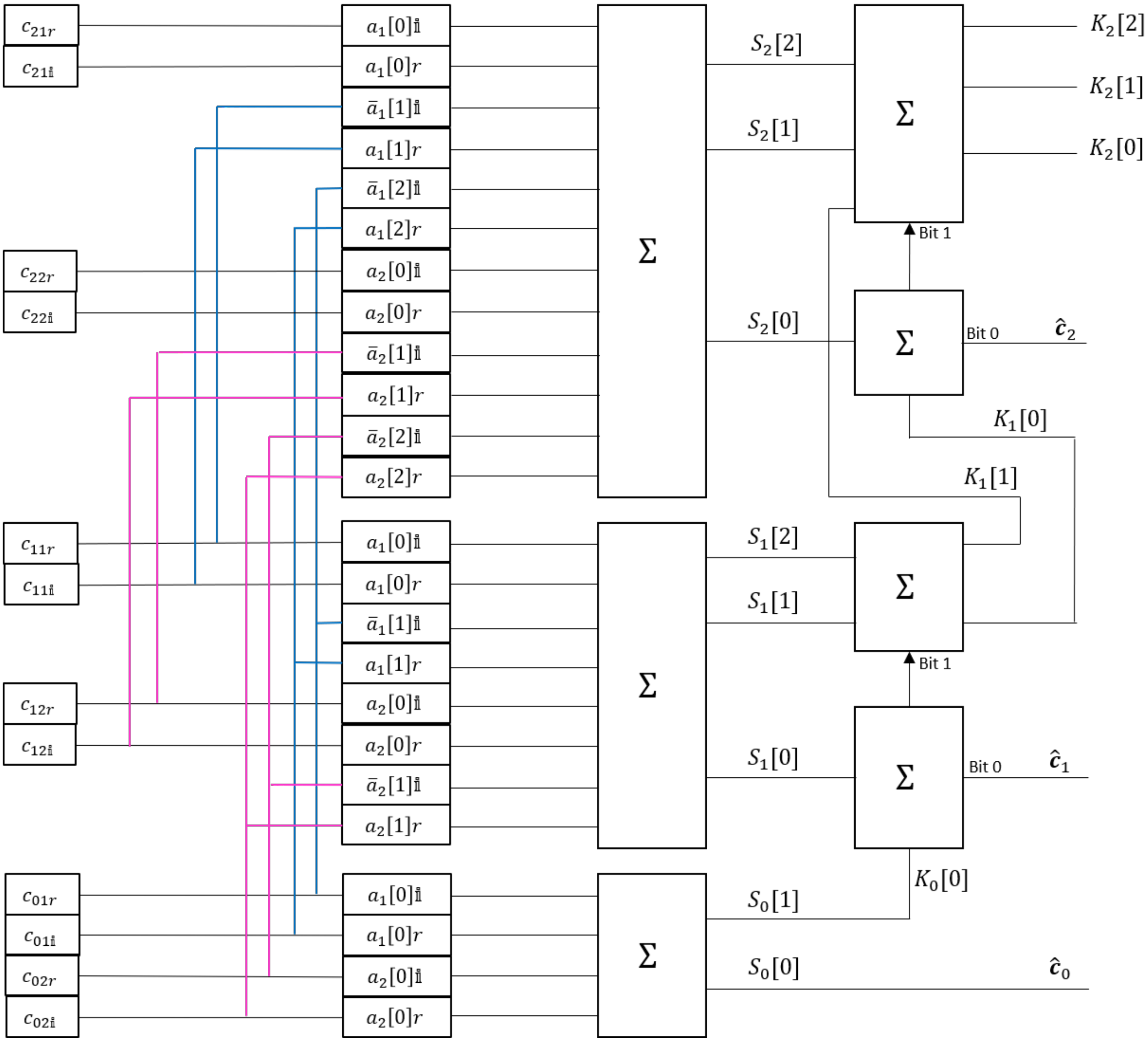


Figure 3.5: Full adder - Quadrature component for complex signals and complex integer coefficients

Chapter 4

BER performance of C&F system model using Construction D

We now present the expected output results of the layered decoder based on particular channel values using our derived expressions in chapter 3 and subsequently demonstrate the BER performance results for the corresponding system model. We initially choose a two user system with three layers, detailing the decoder expected output results for each layer for unit valued channels and complex integer valued channels before extending the system model to incorporate three users. For the evaluation of BER performance we assume the signals transmitted from all users are complex binary. For the three user case we only employ complex integer valued channels as the two user case using unit valued channels can be straight-forwardly extended. BER results are evaluated for two scenarios. Initially we demonstrate results using a repetition code on the inner layer, extended Hamming code on the middle layer and leave the outer layer uncoded. For the second scenario we extend the length of the code and employ a Hadamard code for the inner layer, extended Hamming code for the middle layer and again leave the outer layer uncoded. We then analyse how to recover the individual codes and data messages at the hub for unit valued channels, integer value channels and complex integers valued channels whilst also demonstrating how this can evaluated for real and complex signals.

We begin by describing the operation of the specific encoders and decoders for each code layer, in each scenario, and detail how the chosen codes fulfil the construction D and theorem 2 requirements for the multilayer decoder.

4.1 0 and 2^{nd} order Reed Muller codes

Initially we implement 0 and 2^{nd} order Reed-Muller codes for the inner layer and middle layers respectively. Specifically, we implement the (16,1,16) RM(0,4) code for the inner layer, (16,11,4) RM(2,4) code for the middle layer and leave the outer layer uncoded. As stated by the construction D constraints we separate the minimum euclidean distance between the code layers by a multiple of 4, giving the inner layer $d_{min} = 16$ and middle layer $d_{min} = 4$. The outer layer also fulfils the constraints needed for decoding even though it doesn't need decoding as the AND combinations of the middle and inner layers are indeed a codeword. The structure and design of the inner and middle layer encoder(s) and decoder(s) within the multilayer decoder is given in the next subsections.

4.1.1 Encoding

Inner layer As highlighted in Section 2.5, the inner layer RM(0,4) code sits within a group of repetition codes. Each user generates a length $k = 1$ binary message, where a repetition map, given as $\text{rep}_{16}(\cdot)$, is used to repeat the message to a length $N = 16$ code vector.

Middle layer For the middle layer we encode using a generator matrix of standard form $G = [I, P]$. We ensure that the AND combinations of codewords of the inner layer are codewords of the middle layer. Due to the inner layer being a repetition code we know that the all zeros codeword and all ones codeword are codewords of that layer. This means that we know that the chosen codes fulfil the theorem 2 constraints.

4.1.2 Decoding

Inner layer Decoding the inner layer can be approached in several ways. The approach taken in this scheme is to evaluate the average value of the scaled received signal $\mathbf{y}_0'$ and use this soft value to determine the output of the decoder. Once the average value has been calculated the map $\text{rep}_{16}(\cdot)$ is applied in order reconstruct the codeword.

Middle layer Decoding of the middle layer for the (16,11,4) code allows the application of the work in [18] described in subsection 2.6.5. The work of [18] applies the near-maximum-likelihood Hamming decoder algorithm for multiple frames to decode, in this work we are implementing a single frame and therefore $Nframes = 1$. During encoding we construct the generator matrix in standard form. The work of [18] applies a soft decision decoder (SDD) by calculating the LLRs and syndrome of the errors, therefore the form $G = [I, P]$ is important for this decoding method. Once the SDD has been applied the decoder returns the decoded codeword. The decoder can return the perfectly decoded codeword given that the AND combinations of the inner layer are present in the middle layer. This also means that the form of the generator matrix has an effect on the decodability through the construction of the codeword itself on the middle layer.

4.2 1st and 3rd order Reed-Muller codes

In order to demonstrate successful decoding and improved performance of the overall system we implement 1st and 3rd order Reed-Muller codes on the inner and middle layer respectively. We choose Reed-Muller codes again that satisfy the construction D constraints and those constraints identified with the multilayer decoder, introduced in chapter 3. For the inner layer we implement a (32,6,16) RM(1,5) code generating a length 32 code using 6 data bits of minimum distance $d_{min} = 16$. According to theorem 2, for decodability the AND combination of two codewords of $C_{l'-1}$ must be a codeword of $C_{l'}$. To adhere to this condition and construction D constraints of the minimum euclidean distance between layers being a minimum difference of a multiple of 4, we employ a (32,26,4) RM(3,5) code, code length 32, data length 26 and minimum distance $d_{min} = 4$. The operation of the individual decoders within the multilayer decoder is shown in Figure 4.1.

4.2.1 Encoding inner and middle layers

Inner layer

In order to encode the inner layer data we generate the Hadamard matrix in polar form and select the $1^{st}, 2^{nd}, 4^{th}, 8^{th}, 16^{th}$ and 32^{nd} rows of the matrix and switch back to unipolar form, this results in the generator matrix for layer 0, G_0. G_0 is then matrix multiplied by the inner layer data in order to produce the codewords of C_0.

Middle layer

As the middle layer ($l = 1$) has the inner layer as a preceding layer we ensure that the AND combinations of codewords of the inner layer are also codewords of the middle layer. In order for this to be true we employ the non-standard form generator matrix G_1. This is achieved by applying the wedge product operations between basis vectors g up to order r for g_a where $1 \leq a \leq m$ $\Big(g_0, g_1, \ldots, g_m, \ldots, (g_{a1} \wedge g_{a2}), \ldots, (g_{a1} \wedge g_{a2}, \ldots, \wedge g_{ar})\Big)$. For this particular case we have permutations of $(g_0, g_1, g_2, g_3, g_4, g_5)$ resulting in a 26x32 generator matrix G_1. The standard form generator matrix $G_{1sf} = [I, P]$ is not employed in this case as it does not contain all possible AND combinations of codewords of the inner layer. For this reason we adapt the decoder of [18] for successful decoding, where this is described in the next subsection.

4.2.2 Decoding

Inner layer

As shown in Figure 4.1, the inner layer decoder incorporates the Fast Walsh-Hadamard Transform (FWHT) as the RM(1,5) code is a Hadamard code. The definition given to codes that identify within this group of codes is $RM(1, m)$ where $(2^m, m + 1, 2^{m-1})$.

The process of decoding this layer begins with applying the inverse Fast Walsh-Hadamard Transform and selecting the largest absolute value of this output as an index of the codeword. The inverse transform is then able to be recreated and finally the FWHT applied to the recreated inverse transform of the codeword to obtain the resulting codeword.

Middle layer

Due to the implementation of the non-standard form of the Generator matrix on this layer and the algorithm of [18] requiring a standard form generator matrix, it is necessary to adapt the work of [18]. Therefore, we apply permutations on the vectors of the Generator matrix, which we name G_1 for this layer, such that it is transformed to standard form, $G_1 sf = [I, P]$. This is then input into the algorithm of [18]. The decoding process for the middle layer, as shown in Figure 4.1, is to apply the same permutations on $\mathbf{y}_1^{'}$ prior to inputting the result through the soft-decision decoder. This means that the code can be decoded. The output of this transform is given as $\mathbf{y}_{1sf}^{'}$. Next, the Log-Likelihood ratios are calculated and input into Soft-decision decoder (SDD). Finally, the decoded output codeword has the inverse permutations applied such that the codeword is returned to the original order.

We obviously omit a description of the outer layer decoder due to the data remaining uncoded.

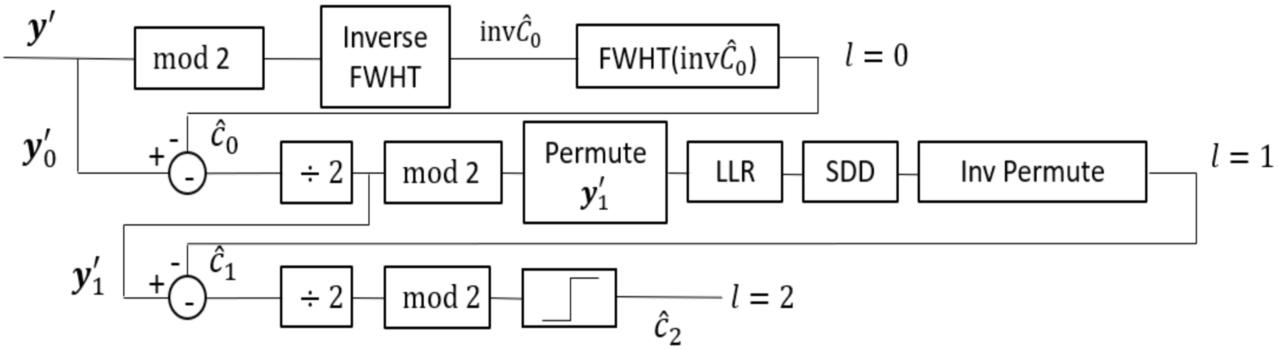


Figure 4.1: Multilayer decoder structure for RM(1,5) and RM(3,5) codes on inner layer and middle layer respectively - outer layer remains uncoded

4.3 Two User - Unit valued channels

We demonstrate two examples of decoding using the same channels but different codes. As shown, both scenarios use RM codes and adhere to construction D constraints and that of theorem 2. We begin by demonstrating a simple example using a two user system model and expressing the expected decoded output using the expressions calculated in chapter 3. We choose a three layer scheme and assume the real only unit valued case for the channels $h_{\{1,2\}} = 1$, and subsequently $\mathbf{a}_{\{1,2\}} = 1$. For complex codes, the decoder is split into two separate entities for the real and imaginary parts. Using (3.39), for layer 0 we may write:

$$\hat{\mathbf{c}}_0 = \hat{\mathbf{c}}_0^I + \mathbb{i}\hat{\mathbf{c}}_0^Q = \mathbf{c}_{01}^I \oplus \mathbf{c}_{02}^I + \mathbb{i}(\mathbf{c}_{01}^Q \oplus \mathbf{c}_{02}^Q) \tag{4.1}$$

which details the expected result of the inner layer decoder which is a modulo two sum of the codewords on layer 0. Given that $\mathbf{c}_{01}$ and $\mathbf{c}_{02}$ are binary (4.1) is in fact the logical XOR of the codewords and indeed the XOR of the data words on that layer. We denote $\oplus$ as the logical XOR operation.

Using (3.41) and (3.43) for layer 1 we may write:

$$\hat{\mathbf{c}}_1 = \left(\mathbf{c}_{11}^I \oplus \mathbf{c}_{12}^I \oplus \left(\mathbf{c}_{01}^I \cdot \mathbf{c}_{02}^I\right)\right) + \mathbb{i}\left(\mathbf{c}_{11}^Q \oplus \mathbf{c}_{12}^Q \oplus \left(\mathbf{c}_{01}^Q \cdot \mathbf{c}_{02}^Q\right)\right) \tag{4.2}$$

During decoding and as expected for the middle layer we can see that the expression (4.2) includes codewords from the middle and inner layers from both sources. The expression can be shown to result in the logical XOR of the codewords on the middle layer plus the logical AND of the codewords on the previous layer, $l = 0$. We denote the logical AND operation as $(\cdot)$.

Again using equations (3.41) and (3.43) for layer 2 we may write:

$$\begin{aligned} \hat{\mathbf{c}}_2 = & \left(\mathbf{c}_{21}^{I} \oplus \mathbf{c}_{22}^{I}\right) \oplus \left(\left(\mathbf{c}_{11}^{I} \cdot \mathbf{c}_{12}^{I}\right) \oplus \left(\mathbf{c}_{11}^{I} \oplus \mathbf{c}_{12}^{I}\right) \cdot \left(\mathbf{c}_{01}^{I} \cdot \mathbf{c}_{02}^{I}\right)\right) \\ & + \mathbb{i}\left(\mathbf{c}_{21}^{Q} \oplus \mathbf{c}_{22}^{Q}\right) \oplus \left(\left(\mathbf{c}_{11}^{Q} \cdot \mathbf{c}_{12}^{Q}\right) \oplus \left(\mathbf{c}_{11}^{Q} \oplus \mathbf{c}_{12}^{Q}\right) \cdot \left(\mathbf{c}_{01}^{Q} \cdot \mathbf{c}_{02}^{Q}\right)\right) \end{aligned} \tag{4.3}$$

As can be expected from our analysis in chapter 3, the number of terms in the expressions of expected output of the decoder at each layer increases with higher number of layers. We see that (4.3) shows that the expression for the expected output on layer 2 is the logical XOR of the layer 2 codewords and the logical XOR of ANDs of the codewords on all previous layers. In this case this is the logical AND combinations of layers 0 and 1 and XOR of layer 1.

4.3.1 RM(0,4), RM(2,4) & uncoded layer BER results

The evaluated BER performance of this scheme is given in Figure 4.2. The results show that the scheme can be successfully decoded when considering the interaction term. The overall BER is given along with the individual performance of the layers. As shown, there is a mismatch in performance between the layers even though the effective minimum Euclidean distance on each layer is the same. The focus of the results is the successful decoding of each layer however, the mismatch of performance occurs due to differing effective numbers of neighbours between the layers. Notice that the middle layer is a sub-optimum maximum likelihood algorithm therefore the difference in performance between the inner layer and middle layer is large.

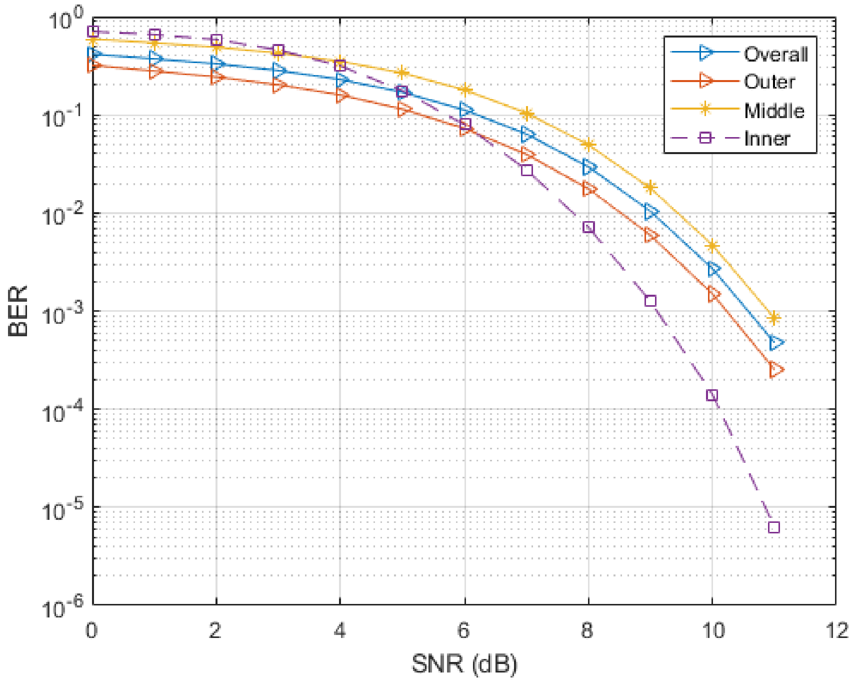


Figure 4.2: Two-user, three layer system model, $h_1 = h_2 = 1$ Inner layer (16,1,16) repetition code, middle layer (16,11,4) Extended Hamming code, outer layer uncoded

4.3.2 RM(1,3), RM(3,5) & uncoded layer BER results

We now implement RM(1,3) code for layer 0 (inner layer), RM(3,5) for layer 1 (middle layer) and again layer 2 (outer layer) remains uncoded.

The BER performance of this proposed scheme, shown in Figure 4.3, shows that the overall performance of the system is consistent for all three layers however the inner layer, as expected, lags behind the middle and outer layer in terms of performance. The performance between the layers is now closer together in contrast to Figure 4.2 due to the decoders implemented. The decodability again is demonstrated and due to the derived expressions of Section 3.7 successful decoding can be carried out.

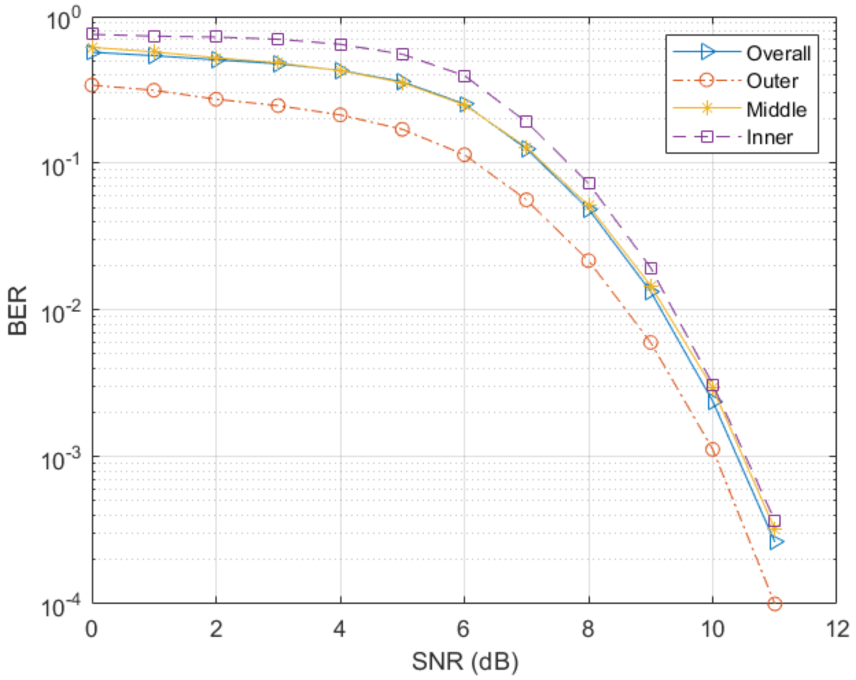


Figure 4.3: Two-user, three layer system model, $h_1 = h_2 = 1$, Inner layer (32,6,16) Hadamard code, middle layer (32,26,4) Extended Hamming code, outer layer uncoded

4.4 Two user - Complex integer valued channels

We extend the previous subsection to complex integer valued channels using the three layer scheme described and detail the expressions for the codewords for each layer. We choose $h_1 = -3 + 5\mathbb{i}$ and $h_2 = 2 + 7\mathbb{i}$. It is easy to see that α is 1 as the channel coefficients are integers and therefore do not need to be moved towards an integer, consequently $a_1 = -3 + 5\mathbb{i}, a_2 = 2 + 7\mathbb{i}$. As we are assuming $\mathbf{x}_{\{1,2\}}$ and $a_{\{1,2\}}$ are complex we can treat the real and imaginary parts separately.

Real part of received signal, neglecting noise and interference: $a^{\mathfrak{R}}_{\{1,2\}}[\{0,1,2\}]\mathbf{c}^{\mathfrak{I}}_{l\{1,2\}}$ and $a^{-\mathfrak{I}}_{\{1,2\}}[\{0,1,2\}]\mathbf{c}^{Q}_{l\{1,2\}}$; imaginary part: $a^{\mathfrak{R}}_{\{1,2\}}[\{0,1,2\}]\mathbf{c}^{Q}_{l\{1,2\}}$ and $a^{\mathfrak{I}}_{\{1,2\}}[\{0,1,2\}]\mathbf{c}^{I}_{l\{1,2\}}$.

For this particular case we evaluate $a^{-\mathfrak{I}}_{j}$ for each user integer coefficient value in order to take into consideration the minus sign due to the imaginary components. For $a_1 = -3 + 5\mathbb{i}$ and $l' = 0, 1, 2$, $a^{\{\mathfrak{R},\mathfrak{I}\}}_{1}[\{0,1,2\}] = \{1,0,1\}$ and $a^{-\mathfrak{I}}_{1} = \{1,1,0\}$: this is -5mod8 = 3 in binary. For $a_2 = 2 + 7\mathbb{i}$ and $l' = 0, 1, 2$, $a^{\{\mathfrak{R},\mathfrak{I}\}}_{2}[\{0,1,2\}] = \{1,1,1\}$ and $a^{-\mathfrak{I}}_{2} = \{1,0,0\}$: this is -7mod8 = 1 in binary. As we begin to accumulate more terms due to complex codes and complex integer channels we introduce a function that describes the combinations of terms within the expression for the logical AND and XOR operation. For a term q this is given as:

$$\text{andxor2}(q_1, \ldots, q_j, \ldots, q_J) \triangleq \bigoplus_{j,j' \in 1,\ldots,J, j \in j'} q_j \cdot q_{j'} \tag{4.4}$$

Using equation (3.63), for layer 0 we may write:

$$\begin{aligned}\hat{\mathbf{c}}_0 = \hat{\mathbf{c}}_0^I + \mathbb{i}\hat{\mathbf{c}}_0^Q = \Big(& a_1^{\Re}[0] \cdot \mathbf{c}_{01}^I \oplus a_1^{-\Im}[0] \cdot \mathbf{c}_{01}^Q \oplus a_2^{\Re}[0] \cdot \mathbf{c}_{02}^I \oplus a_2^{-\Im}[0] \cdot \mathbf{c}_{02}^Q\Big) \\ &+ \mathbb{i}\Big(a_1^{\Re}[0] \cdot \mathbf{c}_{01}^Q \oplus a_1^{\Im}[0] \cdot \mathbf{c}_{01}^I \oplus a_2^{\Re}[0] \cdot \mathbf{c}_{02}^Q \oplus a_2^{\Im}[0] \cdot \mathbf{c}_{02}^I\Big)\end{aligned} \tag{4.5}$$

For layer 1 we split the real and imaginary parts of the expression:

$$\begin{aligned}\hat{\mathbf{c}}_1^I = \Big(& a_1^{\Re}[1] \cdot \mathbf{c}_{01}^I \oplus a_1^{-\Im}[1] \cdot \mathbf{c}_{01}^Q \oplus a_2^{\Re}[1] \cdot \mathbf{c}_{02}^I \oplus a_2^{-\Im}[1] \cdot \mathbf{c}_{02}^Q \\ &\oplus a_1^{\Re}[0] \cdot \mathbf{c}_{11}^I \oplus a_1^{-\Im}[0] \cdot \mathbf{c}_{11}^Q \oplus a_2^{\Re}[0] \cdot \mathbf{c}_{12}^I \oplus a_2^{-\Im}[0] \cdot \mathbf{c}_{12}^Q\Big) \\ &\oplus \text{andxor2}\Big(a_1^{\Re}[0] \cdot \mathbf{c}_{01}^I,\ a_1^{-\Im}[0] \cdot \mathbf{c}_{01}^Q,\ a_2^{\Re}[0] \cdot \mathbf{c}_{02}^I,\ a_2^{-\Im}[0] \cdot \mathbf{c}_{02}^Q\Big)\end{aligned} \tag{4.6}$$

$$\begin{aligned}\hat{\mathbf{c}}_1^Q = \Big(& a_1^{\Re}[1] \cdot \mathbf{c}_{01}^Q \oplus a_1^{\Im}[1] \cdot \mathbf{c}_{01}^I \oplus a_2^{\Re}[1] \cdot \mathbf{c}_{02}^Q \oplus a_2^{\Im}[1] \cdot \mathbf{c}_{02}^I \\ &\oplus a_1^{\Re}[0] \cdot \mathbf{c}_{11}^Q \oplus a_1^{\Im}[0] \cdot \mathbf{c}_{11}^I \oplus a_2^{\Re}[0] \cdot \mathbf{c}_{12}^Q \oplus a_2^{\Im}[0] \cdot \mathbf{c}_{12}^I\Big) \\ &\oplus \text{andxor2}\Big(a_1^{\Re}[0] \cdot \mathbf{c}_{01}^Q,\ a_1^{\Im}[0] \cdot \mathbf{c}_{01}^I,\ a_2^{\Re}[0] \cdot \mathbf{c}_{02}^Q,\ a_2^{\Im}[0] \cdot \mathbf{c}_{02}^I\Big)\end{aligned} \tag{4.7}$$

As in layer 0 and layer 1 we use (3.63) and generate and expression for layer 2:

$$\begin{aligned}\hat{\mathbf{c}}_2^I = \Big(& a_1^{\Re}[2] \cdot \mathbf{c}_{01}^I \oplus a_1^{-\Im}[2] \cdot \mathbf{c}_{01}^Q \oplus a_2^{\Re}[2] \cdot \mathbf{c}_{02}^I \oplus a_2^{-\Im}[2] \cdot \mathbf{c}_{02}^Q \\ &\oplus a_1^{\Re}[1] \cdot \mathbf{c}_{11}^I \oplus a_1^{-\Im}[1] \cdot \mathbf{c}_{11}^Q \oplus a_2^{\Re}[1] \cdot \mathbf{c}_{12}^I \oplus a_2^{-\Im}[1] \cdot \mathbf{c}_{12}^Q \\ &\oplus a_1^{\Re}[0] \cdot \mathbf{c}_{21}^I \oplus a_1^{-\Im}[0] \cdot \mathbf{c}_{21}^Q \oplus a_2^{\Re}[0] \cdot \mathbf{c}_{22}^I \oplus a_2^{-\Im}[0] \cdot \mathbf{c}_{22}^Q\Big) \\ &\oplus \text{andxor2}\Big(a_1^{\Re}[0] \cdot \mathbf{c}_{11}^I,\ a_1^{-\Im}[0] \cdot \mathbf{c}_{11}^Q,\ a_2^{\Re}[0] \cdot \mathbf{c}_{12}^I,\ a_2^{-\Im}[0] \cdot \mathbf{c}_{12}^Q, \\ &a_1^{\Re}[1] \cdot \mathbf{c}_{01}^I,\ a_1^{-\Im}[1] \cdot \mathbf{c}_{01}^Q,\ a_2^{\Re}[1] \cdot \mathbf{c}_{02}^I,\ a_2^{-\Im}[1] \cdot \mathbf{c}_{02}^Q\Big) \\ &\oplus \text{andxor2}\Big(a_1^{\Re}[0] \cdot \mathbf{c}_{01}^I,\ a_1^{-\Im}[0] \cdot \mathbf{c}_{01}^Q,\ a_2^{\Re}[0] \cdot \mathbf{c}_{02}^I,\ a_2^{-\Im}[0] \cdot \mathbf{c}_{02}^Q\Big)\end{aligned} \tag{4.8}$$

$$\begin{aligned}\hat{\mathbf{c}}_2^Q = \Big(&a_1^{\Re}[2]\cdot\mathbf{c}_{01}^Q \oplus a_1^{\Im}[2]\cdot\mathbf{c}_{01}^I \oplus a_2^{\Re}[2]\cdot\mathbf{c}_{02}^Q \oplus a_2^{\Im}[2]\cdot\mathbf{c}_{02}^I\\ &\oplus a_1^{\Re}[1]\cdot\mathbf{c}_{11}^Q \oplus a_1^{\Im}[1]\cdot\mathbf{c}_{11}^I \oplus a_2^{\Re}[1]\cdot\mathbf{c}_{12}^Q \oplus a_2^{\Im}[1]\cdot\mathbf{c}_{12}^I\\ &\oplus a_1^{\Re}[0]\cdot\mathbf{c}_{21}^Q \oplus a_1^{\Im}[0]\cdot\mathbf{c}_{21}^I \oplus a_2^{\Re}[0]\cdot\mathbf{c}_{22}^Q \oplus a_2^{\Im}[0]\cdot\mathbf{c}_{22}^I\Big)\\ &\oplus \text{andxor2}\Big(a_1^{\Re}[0]\cdot\mathbf{c}_{11}^Q,\ a_1^{\Im}[0]\cdot\mathbf{c}_{11}^I,\ a_2^{\Re}[0]\cdot\mathbf{c}_{12}^Q,\ a_2^{\Im}[0]\cdot\mathbf{c}_{12}^I,\\ &a_1^{\Re}[1]\cdot\mathbf{c}_{01}^Q,\ a_1^{\Im}[1]\cdot\mathbf{c}_{01}^I,\ a_2^{\Re}[1]\cdot\mathbf{c}_{02}^Q,\ a_2^{\Im}[1]\cdot\mathbf{c}_{02}^I\Big)\\ &\oplus \text{andxor2}\Big(a_1^{\Re}[0]\cdot\mathbf{c}_{01}^Q,\ a_1^{\Im}[0]\cdot\mathbf{c}_{01}^I,\ a_2^{\Re}[0]\cdot\mathbf{c}_{02}^Q,\ a_2^{\Im}[0]\cdot\mathbf{c}_{02}^I\Big)\end{aligned} \tag{4.9}$$

Equations (4.5-4.9) are the derived mathematical expressions implemented for successful decoding, where the complexity increases from inner to outermost layers. For this example we highlight the ‘carry’ term $\mathbf{k}_{l'}$ for layer 1 and layer 2, where layer 0 does not have one due to no previous layers.

Layer 1 - In-phase:

$$\mathbf{k}_1^I = \text{andxor2}\Big(a_1^{\Re}[0]\cdot\mathbf{c}_{01}^I,\ a_1^{-\Im}[0]\cdot\mathbf{c}_{01}^Q,\ a_2^{\Re}[0]\cdot\mathbf{c}_{02}^I,\ a_2^{-\Im}[0]\cdot\mathbf{c}_{02}^Q\Big) \tag{4.10}$$

Quadrature:

$$\mathbf{k}_1^Q = \text{andxor2}\Big(a_1^{\Re}[0]\cdot\mathbf{c}_{01}^Q,\ a_1^{\Im}[0]\cdot\mathbf{c}_{01}^I,\ a_2^{\Re}[0]\cdot\mathbf{c}_{02}^Q,\ a_2^{\Im}[0]\cdot\mathbf{c}_{02}^I\Big) \tag{4.11}$$

As per the unit valued case in Section 4.2, $\mathbf{k}_1^{\{I,Q\}}$ is shown mathematically to be the logical AND combinations of codewords on layer 0 and layer 1 but now extended to include In-phase and Quadrature components.

Layer 2 - In-phase:

$$\begin{aligned}\mathbf{k}_2^I = \text{andxor2}\Big(&a_1^{\Re}[0]\cdot\mathbf{c}_{11}^I,\ a_1^{-\Im}[0]\cdot\mathbf{c}_{11}^Q,\ a_2^{\Re}[0]\cdot\mathbf{c}_{12}^I,\ a_2^{-\Im}[0]\cdot\mathbf{c}_{12}^Q,\\ &a_1^{\Re}[1]\cdot\mathbf{c}_{01}^I,\ a_1^{-\Im}[1]\cdot\mathbf{c}_{01}^Q,\ a_2^{\Re}[1]\cdot\mathbf{c}_{02}^I,\ a_2^{-\Im}[1]\cdot\mathbf{c}_{02}^Q\Big)\\ \oplus \text{andxor2}\Big(&a_1^{\Re}[0]\cdot\mathbf{c}_{01}^I,\ a_1^{-\Im}[0]\cdot\mathbf{c}_{01}^Q,\ a_2^{\Re}[0]\cdot\mathbf{c}_{02}^I,\ a_2^{-\Im}[0]\cdot\mathbf{c}_{02}^Q\Big)\end{aligned} \tag{4.12}$$

Quadrature:

$$
\begin{aligned}
\mathbf{k}_2^Q = \operatorname{andxor2}\Big(& a_1^{\Re}[0]\cdot\mathbf{c}_{11}^Q,\ a_1^{\Im}[0]\cdot\mathbf{c}_{11}^I,\ a_2^{\Re}[0]\cdot\mathbf{c}_{12}^Q,\ a_2^{\Im}[0]\cdot\mathbf{c}_{12}^I,\\
& a_1^{\Re}[1]\cdot\mathbf{c}_{01}^Q,\ a_1^{\Im}[1]\cdot\mathbf{c}_{01}^I,\ a_2^{\Re}[1]\cdot\mathbf{c}_{02}^Q,\ a_2^{\Im}[1]\cdot\mathbf{c}_{02}^I\Big)\\
& \oplus \operatorname{andxor2}\Big(a_1^{\Re}[0]\cdot\mathbf{c}_{01}^Q,\ a_1^{\Im}[0]\cdot\mathbf{c}_{01}^I,\ a_2^{\Re}[0]\cdot\mathbf{c}_{02}^Q,\ a_2^{\Im}[0]\cdot\mathbf{c}_{02}^I\Big)
\end{aligned}
\tag{4.13}
$$

Finally, (4.12) and (4.13) demonstrate that the first term is the logical AND combinations of layer 0 and layer 1 components and the second term is the logical AND combinations of layer 0 combinations where these combinations are combined using the XOR operation.

4.4.1 RM(0,4), RM(2,4) & uncoded layer BER results

The BER performance of this system using RM(0,4), RM(2,4) and uncoded for layers 0,1,2 respectively demonstrates successful decoding of the three layer system using the derived mathematical expressions (4.5 - 4.9). Figure 4.4 shows that the same trend of each layer follows that shown in the unit valued channel case which is what we expect.

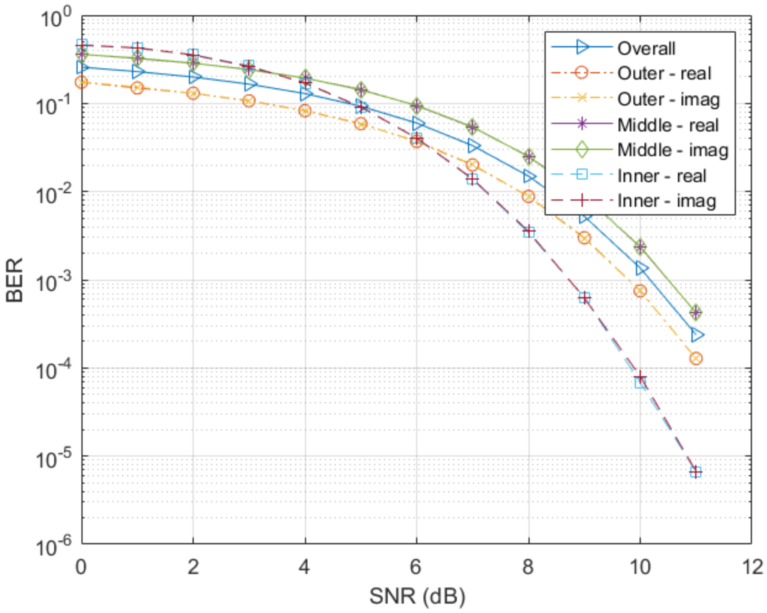


Figure 4.4: Two user, three layer system model - $h_1 = -3 + 5\mathbb{i}, h_2 = 2 + 7\mathbb{i}$, Inner layer (16,1,16) repetition code, middle layer (16,11,4) Extended Hamming code, outer layer uncoded

4.4.2 RM(1,5), RM(3,5) & uncoded layer BER results

As in the unit valued channel case the codes chosen to evaluate these derived expressions for the expected output are the (32,6,16) Hadamard code for layer 0 (inner layer), (32,26,4) Extended Hamming code for layer 1 (middle layer) and again layer 2 (outer layer) remains uncoded. Clearly, as with the previous subsection, Figure 4.5 demonstrates successful decoding of all three layers for both In-phase and Quadrature components using the derived expressions for this particular complex channel case. By observing Figure 4.5 we see that each layer and indeed the overall BER performance of the system when comparing to Figure 4.3 follows the same trend for all three layers.

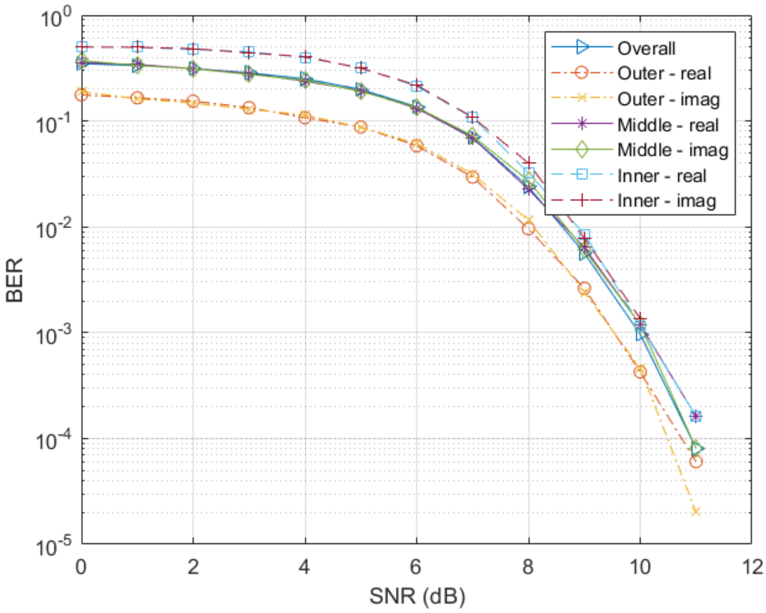


Figure 4.5: Two user, three layer system model - $h_1 = -3 + 5\mathrm{i}, h_2 = 2 + 7\mathrm{i}$, Inner layer (32,6,16) Hadamard code, middle layer (32,26,4) Extended Hamming code, outer layer uncoded

4.5 Three users - Complex integer valued channels

We extend the two user case to consider three users within our complex signal, complex integer channel, system model. We apply the same equations as in the previous Section and show that the derived expressions can be used to show the successful decoding of each layer.

We begin by extending the results of the previous subsection and assume that again $\mathbf{x}_{\{1,2,3\}}$ and $\mathbf{c}_{\{1,2,3\}}$ are complex and binary. We now consider the channels h_1, h_2, h_3, where these channels are also deemed to be complex integers and therefore: $a_{\{1,2,3\}} = a^{\Re}_{\{1,2,3\}} + \mathbb{i}a^{\Im}_{\{1,2,3\}}$. We choose the complex integer channels $h_1 = 1 + 2\mathbb{i}, h_2 = -3 + \mathbb{i}$ and $h_3 = 2 + 2\mathbb{i}$ at random. Again, we take into consideration the minus sign before the integer value for each channel value and determine the appropriate bit value for each layer. For $h_1 = 1 + 2\mathbb{i}$ and $l' = 0, 1, 2$, $a_1^{\{\Re,\Im\}}[\{0, 1, 2\}] = \{0, 1, 0\}$ and $a_1^{-\Im} = \{0, 1, 1\}$: this is -2mod8 = 6 in binary. For $a_2 = -3 + \mathbb{i}$ and $l' = 0, 1, 2$, $a_2^{\{\Re,\Im\}}[\{0, 1, 2\}] = \{1, 0, 0\}$ and $a_2^{-\Im} = \{1, 1, 1\}$: this is -1mod8 = 7 in binary. For $a_3 = 2 + 2\mathbb{i}$ and $l' = 0, 1, 2$, again, $a_3^{\{\Re,\Im\}}[\{0, 1, 2\}] = \{0, 1, 0\}$ and $a_3^{-\Im} = \{0, 1, 1\}$.

As in the two user case we use (3.63) to determine an expression for the expected decoded output at each layer. For layer 0 we may write:

$$\begin{aligned}\hat{\mathbf{c}}_0 = \hat{\mathbf{c}}_0^I + \mathbb{i}\hat{\mathbf{c}}_0^Q = &\Big(a_1^{\Re}[0] \cdot \mathbf{c}_{01}^I \oplus a_1^{-\Im}[0] \cdot \mathbf{c}_{01}^Q \oplus a_2^{\Re}[0] \cdot \mathbf{c}_{02}^I \oplus a_2^{-\Im}[0] \cdot \mathbf{c}_{02}^Q \oplus a_3^{\Re}[0] \cdot \mathbf{c}_{03}^I \oplus a_3^{-\Im}[0] \cdot \mathbf{c}_{03}^Q\Big) \\ &+ \mathbb{i}\Big(a_1^{\Re}[0] \cdot \mathbf{c}_{01}^Q \oplus a_1^{\Im}[0] \cdot \mathbf{c}_{01}^I \oplus a_2^{\Re}[0] \cdot \mathbf{c}_{02}^Q \oplus a_2^{\Im}[0] \cdot \mathbf{c}_{02}^I \oplus a_3^{\Re}[0] \cdot \mathbf{c}_{03}^Q \oplus a_3^{\Im}[0] \cdot \mathbf{c}_{03}^I\Big)\end{aligned} \tag{4.14}$$

For layer 1 we split the real and imaginary parts of the expression:

$$\begin{aligned}\hat{\mathbf{c}}_1^I = &\Big(a_1^{\Re}[1] \cdot \mathbf{c}_{01}^I \oplus a_1^{-\Im}[1] \cdot \mathbf{c}_{01}^Q \oplus a_2^{\Re}[1] \cdot \mathbf{c}_{02}^I \oplus a_2^{-\Im}[1] \cdot \mathbf{c}_{02}^Q \oplus a_3^{\Re}[1] \cdot \mathbf{c}_{03}^I \oplus a_3^{-\Im}[1]\mathbf{c}_{03}^Q \\ &\oplus a_1^{\Re}[0] \cdot \mathbf{c}_{11}^I \oplus a_1^{-\Im}[0] \cdot \mathbf{c}_{11}^Q \oplus a_2^{\Re}[0] \cdot \mathbf{c}_{12}^I \oplus a_2^{-\Im}[0] \cdot \mathbf{c}_{12}^Q \oplus a_3^{\Re}[0] \cdot \mathbf{c}_{13}^I \oplus a_3^{-\Im}[0] \cdot \mathbf{c}_{13}^Q\Big) \\ &\oplus \text{andxor2}\Big(a_1^{\Re}[0] \cdot \mathbf{c}_{01}^I,\ a_1^{-\Im}[0] \cdot \mathbf{c}_{01}^Q,\ a_2^{\Re}[0] \cdot \mathbf{c}_{02}^I,\ a_2^{-\Im}[0] \cdot \mathbf{c}_{02}^Q,\ a_3^{\Re}[0] \cdot \mathbf{c}_{03}^I,\ a_3^{-\Im}[0] \cdot \mathbf{c}_{03}^Q\Big)\end{aligned} \tag{4.15}$$

$$\begin{aligned}\hat{\mathbf{c}}_1^Q = \Big(& a_1^{\Re}[1]\cdot\mathbf{c}_{01}^Q \oplus a_1^{\Im}[1]\cdot\mathbf{c}_{01}^I \oplus a_2^{\Re}[1]\cdot\mathbf{c}_{02}^Q \oplus a_2^{\Im}[1]\cdot\mathbf{c}_{02}^I \oplus a_3^{\Re}[1]\cdot\mathbf{c}_{03}^Q \oplus a_3^{\Im}[1]\cdot\mathbf{c}_{03}^I \\ &\oplus a_1^{\Re}[0]\cdot\mathbf{c}_{11}^Q \oplus a_1^{\Im}[0]\cdot\mathbf{c}_{11}^I \oplus a_2^{\Re}[0]\cdot\mathbf{c}_{12}^Q \oplus a_2^{\Im}[0]\cdot\mathbf{c}_{12}^I \oplus a_3^{\Re}[0]\cdot\mathbf{c}_{13}^Q \oplus a_3^{\Im}[0]\cdot\mathbf{c}_{13}^I\Big) \\ &\oplus \mathrm{andxor2}\Big(a_1^{\Re}[0]\cdot\mathbf{c}_{01}^Q,\ a_1^{\Im}[0]\cdot\mathbf{c}_{01}^I,\ a_2^{\Re}[0]\cdot\mathbf{c}_{02}^Q,\ a_2^{\Im}[0]\cdot\mathbf{c}_{02}^I,\ a_3^{\Re}[0]\cdot\mathbf{c}_{03}^Q,\ a_3^{\Im}[0]\cdot\mathbf{c}_{03}^I\Big)\end{aligned} \tag{4.16}$$

For layer 2:

$$\begin{aligned}\hat{\mathbf{c}}_2^I = \Big(& a_1^{\Re}[2]\cdot\mathbf{c}_{01}^I \oplus a_1^{-\Im}[2]\cdot\mathbf{c}_{01}^Q \oplus a_2^{\Re}[2]\cdot\mathbf{c}_{02}^I \oplus a_2^{-\Im}[2]\cdot\mathbf{c}_{02}^Q \oplus a_3^{\Re}[2]\cdot\mathbf{c}_{03}^I \oplus a_3^{-\Im}[2]\cdot\mathbf{c}_{03}^Q \\ &\oplus a_1^{\Re}[1]\cdot\mathbf{c}_{11}^I \oplus a_1^{-\Im}[1]\cdot\mathbf{c}_{11}^Q \oplus a_2^{\Re}[1]\cdot\mathbf{c}_{12}^I \oplus a_2^{-\Im}[1]\cdot\mathbf{c}_{12}^Q \oplus a_3^{\Re}[1]\cdot\mathbf{c}_{13}^I \oplus a_3^{-\Im}[1]\cdot\mathbf{c}_{13}^Q \\ &\oplus a_1^{\Re}[0]\cdot\mathbf{c}_{21}^I \oplus a_1^{-\Im}[0]\cdot\mathbf{c}_{21}^Q \oplus a_2^{\Re}[0]\cdot\mathbf{c}_{22}^I \oplus a_2^{-\Im}[0]\cdot\mathbf{c}_{22}^Q \oplus a_3^{\Re}[0]\cdot\mathbf{c}_{23}^I \oplus a_3^{-\Im}[0]\cdot\mathbf{c}_{23}^Q\Big)\end{aligned} \tag{4.17}$$

$$\begin{aligned}&\oplus \mathrm{andxor2}\Big(a_1^{\Re}[0]\cdot\mathbf{c}_{11}^I,\ a_1^{-\Im}[0]\cdot\mathbf{c}_{11}^Q,\ a_2^{\Re}[0]\cdot\mathbf{c}_{12}^I,\ a_2^{-\Im}[0]\cdot\mathbf{c}_{12}^Q,\ a_3^{\Re}[0]\cdot\mathbf{c}_{13}^I,\ a_3^{-\Im}[0]\cdot\mathbf{c}_{13}^Q, \\ & a_1^{\Re}[1]\cdot\mathbf{c}_{01}^I,\ a_1^{-\Im}[1]\cdot\mathbf{c}_{01}^Q,\ a_2^{\Re}[1]\cdot\mathbf{c}_{02}^I,\ a_2^{-\Im}[1]\cdot\mathbf{c}_{02}^Q,\ a_3^{\Re}[1]\cdot\mathbf{c}_{03}^I,\ a_3^{-\Im}[1]\mathbf{c}_{03}^Q\Big) \\ &\oplus \mathrm{andxor2}\Big(a_1^{\Re}[0]\cdot\mathbf{c}_{01}^I,\ a_1^{-\Im}[0]\cdot\mathbf{c}_{01}^Q,\ a_2^{\Re}[0]\cdot\mathbf{c}_{02}^I,\ a_2^{-\Im}[0]\cdot\mathbf{c}_{02}^Q,\ a_3^{\Re}[0]\cdot\mathbf{c}_{03}^I,\ a_3^{-\Im}[0]\cdot\mathbf{c}_{03}^Q\Big)\end{aligned}$$

$$\begin{aligned}\hat{\mathbf{c}}_2^Q = \Big(& a_1^{\Re}[2]\cdot\mathbf{c}_{01}^Q \oplus a_1^{\Im}[2]\cdot\mathbf{c}_{01}^I \oplus a_2^{\Re}[2]\cdot\mathbf{c}_{02}^Q \oplus a_2^{\Im}[2]\cdot\mathbf{c}_{02}^I \oplus a_3^{\Re}[2]\cdot\mathbf{c}_{03}^Q \oplus a_3^{\Im}[2]\cdot\mathbf{c}_{03}^I \\ &\oplus a_1^{\Re}[1]\cdot\mathbf{c}_{11}^Q \oplus a_1^{\Im}[1]\cdot\mathbf{c}_{11}^I \oplus a_2^{\Re}[1]\cdot\mathbf{c}_{12}^Q \oplus a_2^{\Im}[1]\cdot\mathbf{c}_{12}^I \oplus a_3^{\Re}[1]\cdot\mathbf{c}_{13}^Q \oplus a_3^{\Im}[1]\cdot\mathbf{c}_{13}^I \\ &\oplus a_1^{\Re}[0]\cdot\mathbf{c}_{21}^Q \oplus a_1^{\Im}[0]\cdot\mathbf{c}_{21}^I \oplus a_2^{\Re}[0]\cdot\mathbf{c}_{22}^Q \oplus a_2^{\Im}[0]\cdot\mathbf{c}_{22}^I \oplus a_3^{\Re}[0]\cdot\mathbf{c}_{23}^Q \oplus a_3^{\Im}[0]\cdot\mathbf{c}_{23}^I\Big) \\ &\oplus \mathrm{andxor2}\Big(a_1^{\Re}[0]\cdot\mathbf{c}_{11}^Q,\ a_1^{\Im}[0]\cdot\mathbf{c}_{11}^I,\ a_2^{\Re}[0]\cdot\mathbf{c}_{12}^Q,\ a_2^{\Im}[0]\cdot\mathbf{c}_{12}^I,\ a_3^{\Re}[0]\cdot\mathbf{c}_{13}^Q,\ a_3^{\Im}[0]\cdot\mathbf{c}_{13}^I, \\ & a_1^{\Re}[1]\cdot\mathbf{c}_{01}^Q,\ a_1^{\Im}[1]\cdot\mathbf{c}_{01}^I,\ a_2^{\Re}[1]\cdot\mathbf{c}_{02}^Q,\ a_2^{\Im}[1]\cdot\mathbf{c}_{02}^I,\ a_3^{\Re}[1]\cdot\mathbf{c}_{03}^Q,\ a_3^{\Im}[1]\cdot\mathbf{c}_{03}^I\Big) \\ &\oplus \mathrm{andxor2}\Big(a_1^{\Re}[0]\cdot\mathbf{c}_{01}^Q,\ a_1^{\Im}[0]\cdot\mathbf{c}_{01}^I,\ a_2^{\Re}[0]\cdot\mathbf{c}_{02}^Q,\ a_2^{\Im}[0]\cdot\mathbf{c}_{02}^I,\ a_3^{\Re}[0]\cdot\mathbf{c}_{03}^Q,\ a_3^{\Im}[0]\cdot\mathbf{c}_{03}^I\Big)\end{aligned} \tag{4.18}$$

As in the previous subsection these equations are used to evaluate decoding in our simulation and calculate BER performance.

4.5.1 RM(0,4), RM(2,4) & uncoded layer BER results

The BER performance of the system demonstrates that successful decoding can straightforwardly be expanded to the three user case and that our derived expressions and implementation for complex signals and channels in this particular case, are versatile and can be used for multiple users and nested lattice layers. If we compare Figure 4.4 (two users) and Figure 4.6 (three users) we see that the performance is extremely similar for each layer. We can therefore demonstrate that for our example scheme we can successfully decode with increasing numbers of users and the performance not be degraded. We chose random complex integer channel values for the two-user and three-user case such that we can demonstrate that the derived expressions are suitable for varying complex integer values. As expected the performance of the inner layer, both In-phase and Quadrature, is greater than that of the middle and outer layers.

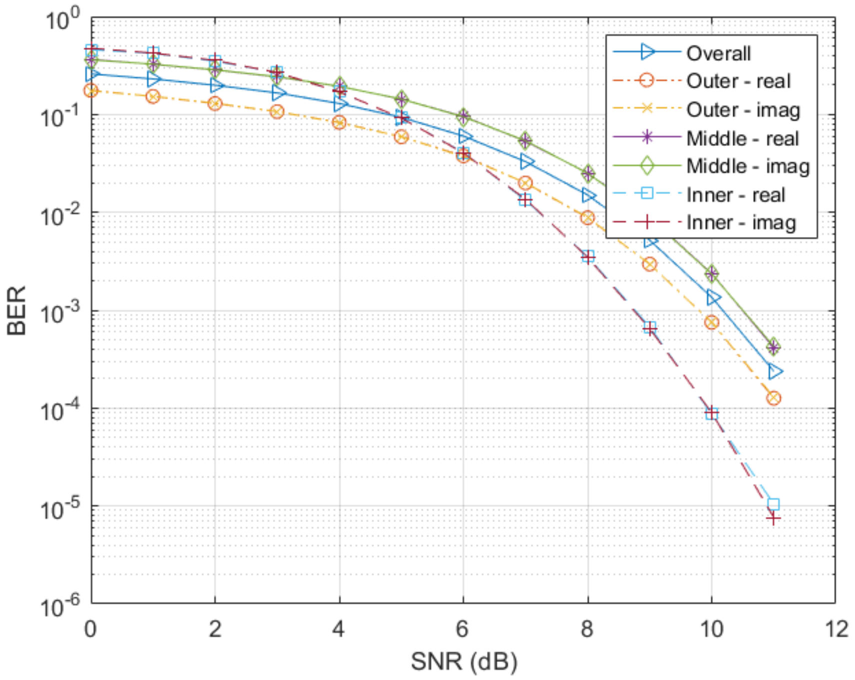


Figure 4.6: Three user, three layer system model - $h_1 = 1+2\mathbb{i}$, $h_2 = -3+\mathbb{i}$, $h_3 = 2+2\mathbb{i}$, Inner layer (16,1,16) repetition code, middle layer (16,11,4) Extended Hamming code, outer layer uncoded

4.5.2 RM(1,3), RM(3,5) & uncoded layer BER results

Again, the same RM(1,3), RM(3,5) and uncoded layer codes are evaluated to determine successful decoding and show the performance of the system using the same channel coefficients as the RM(0,4),RM(2,4) and uncoded layer scenario. As with the RM(0,4), RM(2,4) scheme Figure 4.7 shows the successful decodability of a three user system model and validation of the derived expressions of the expected output at each code layer.

With comparison to Figure 4.5 the BER results demonstrate the same performance with the three user case in Figure 4.7 which is expected and shows that with the increase of a user the performance has not degraded. The main result from this is that the decodability of the derived expressions and design of the decoders provide decoding structure that is feasible for C&F.

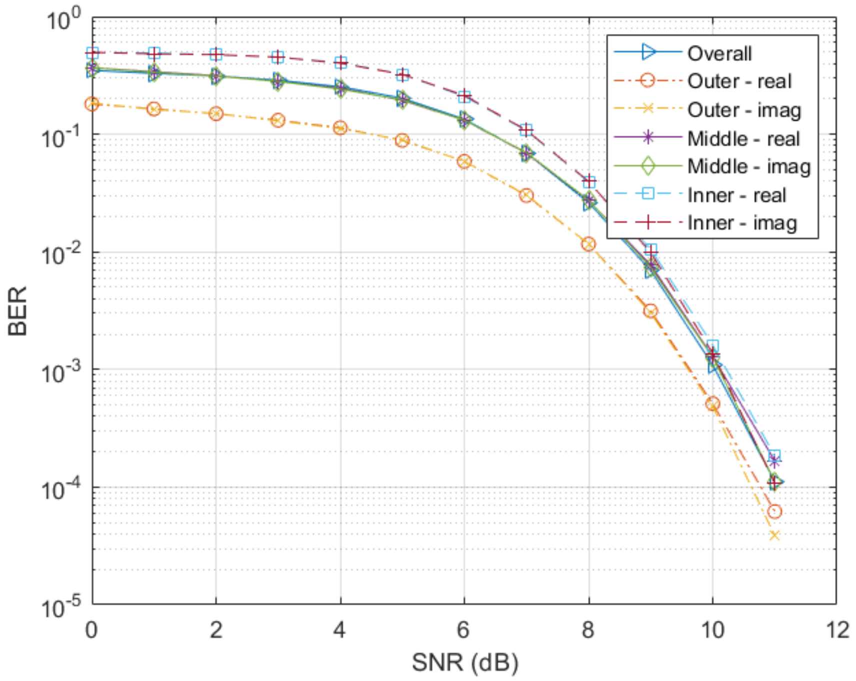


Figure 4.7: Three user, three layer system model - $h_1 = 1 + 2\imath, h_2 = -3 + \imath, h_3 = 2 + 2\imath$, Inner layer (32,6,16) Hadamard code, middle layer (32,26,4) Extended Hamming code, outer layer uncoded

4.6 Decoding at the hub/CPU

The compute-and-forward scheme describes base stations as decoding for equations of messages. For this section we focus on decoding at the hub and in particular uncovering the individual codewords and datawords at each source. We consider a two-way relay channel model and detail the transmitted signals to the hub and highlight known information at each user i.e. the users know their own codewords for all layers. We also determine that the carry term is known for each layer and therefore can be used to recover the combination of codewords. We assume that the hub also shares knowledge of both integer coefficients to both sources. We demonstrate this using a two-way relay channel as shown in Figure 2.1 (b). The transmission of this two way relay channel is described as full-duplex as both sources can transmit at the same time, as opposed to Figure 2.1 (a) which is half-duplex. As mentioned in subsection 2.4.4 this configuration means that $\mathbf{Q}$ maybe singular under some circumstances and therefore potentially not all messages can be recovered. In the case where $\mathbf{Q}$ is singular it may be possible to choose different coefficients which allow all the data to be recovered.

For this work we base our analysis on the two user, three layer system model as in the previous sections. Discussion is presented for the case of repetition code for the inner layer, extended Hamming code for the middle layer and leave the outer layer uncoded. Discussion is also presented for general codes and their nature to decode the individual data messages.

4.6.1 Unit valued channels and real only signals

We begin by using the real unit valued case where the codewords are in the reals $\mathbb{R}^N$ and the channel coefficients are unit valued. We know that the expected decoded output of the inner layer is the XOR of the inner layer codewords by using (3.33). By employing the two-way relay channel model we assume that each source knows its own information, for example S_1 knows $\mathbf{c}_{\{0,1,2\}1}$ and S_2 knows $\mathbf{c}_{\{0,1,2\}2}$. If we apply this knowledge we can decode the individual codewords from each source on the inner layer. For source 1 we apply:

$$(\mathbf{c}_{01} \oplus \mathbf{c}_{02}) \oplus \mathbf{c}_{01} \tag{4.19}$$

which leaves $\mathbf{c}_{02}$.

For source 2:

$$(\mathbf{c}_{01} \oplus \mathbf{c}_{02}) \oplus \mathbf{c}_{02} \tag{4.20}$$

which leaves $\mathbf{c}_{01}$. From the XOR of the codewords we can determine that this also gives the XOR of the datawords.

$$\mathbf{w}_{01} \oplus \mathbf{w}_{02} \tag{4.21}$$

In general we could also determine the datawords by applying the inner layer decoder to (4.19) and (4.20), however for the repetition code it is straight forward to determine the data bit from the decoded individual codes $\mathbf{c}_{01}$ and $\mathbf{c}_{02}$. The data messages can then be encoded using the inner layer encoder to generate the code words and be used as information at the next layer.

We now examine this for the middle layer. We know that the expected decoded output can be determined by (3.33) and also now know additional information of the codewords, $\mathbf{c}_{01}$ and $\mathbf{c}_{02}$. In order to decode the individual codewords of the middle layer we would like to remove the AND combination of $\mathbf{c}_{01}$ and $\mathbf{c}_{02}$ which is the carry term expressed using (3.35). In order to do so we apply the XOR operation of this carry term on the decoded output.

$$(\mathbf{c}_{11} \oplus \mathbf{c}_{12} \oplus (\mathbf{c}_{01} \cdot \mathbf{c}_{02})) \oplus (\mathbf{c}_{01} \cdot \mathbf{c}_{02}) \tag{4.22}$$

This leaves,

$$\mathbf{c}_{11} \oplus \mathbf{c}_{12} \tag{4.23}$$

where we can again use known information of the codewords on the middle layer from both source 1 and source 2 to decode for the individual codewords. We apply the XOR between the known codeword of source 1 and (4.23) to decode for the codeword of source 2.

$$(\mathbf{c}_{11} \oplus \mathbf{c}_{12}) \oplus \mathbf{c}_{11} \tag{4.24}$$

leaving $\mathbf{c}_{12}$. Equally we can apply the same and decode for the codeword $\mathbf{c}_{11}$ at source 2.

$$(\mathbf{c}_{11} \oplus \mathbf{c}_{12}) \oplus \mathbf{c}_{12} \tag{4.25}$$

leaving $\mathbf{c}_{11}$. We again can determine that the XOR of the codewords also gives the XOR of the datawords and can employ the same process for layer 1 as in layer 0 to decode the individual data messages.

$$\mathbf{w}_{11} \oplus \mathbf{w}_{12} \tag{4.26}$$

Specifically, the use of the (16,11,4) extended Hamming code means that we can decode the dataword for both sources. The dataword for both source 1 and 2 is given by the first 11 bits of $\mathbf{c}_{11}$ and $\mathbf{c}_{12}$. For general codes we can determine that if they are of systematic form then the data bits can be recovered where the remaining bits are classed as parity bits. It can also be shown that once the data bits have been recovered the parity bits can be added to the data in order to regenerate the codeword which can be passed on as known information used for decoding the next layer.

For the outer layer we apply the use of the known codewords and the known interaction terms using the inner layer codewords $\mathbf{c}_{01}, \mathbf{c}_{02}$ and middle layer codewords $\mathbf{c}_{11}, \mathbf{c}_{12}$. The decoded output using (3.33) is:

$$(\mathbf{c}_{21} \oplus \mathbf{c}_{22}) \oplus ((\mathbf{c}_{11} \cdot \mathbf{c}_{12}) \oplus (\mathbf{c}_{11} \oplus \mathbf{c}_{12}) \cdot (\mathbf{c}_{01} \cdot \mathbf{c}_{02})) \tag{4.27}$$

We apply the XOR operation with interaction term to the decoded output and the result shows the XOR of $\mathbf{c}_{21}$ and $\mathbf{c}_{22}$.

$$\begin{split}\Big((\mathbf{c}_{21} \oplus \mathbf{c}_{22}) \oplus \Big((\mathbf{c}_{11} \cdot \mathbf{c}_{12}) \oplus (\mathbf{c}_{11} \oplus \mathbf{c}_{12}) \cdot (\mathbf{c}_{01} \cdot \mathbf{c}_{02})\Big)\Big) \\ \oplus \Big((\mathbf{c}_{11} \cdot \mathbf{c}_{12}) \oplus (\mathbf{c}_{11} \oplus \mathbf{c}_{12}) \cdot (\mathbf{c}_{01} \cdot \mathbf{c}_{02})\Big) = \mathbf{c}_{21} \oplus \mathbf{c}_{22}\end{split} \tag{4.28}$$

As in the previous layers we can apply the known information for each of the sources. For source 2:

$$(\mathbf{c}_{21} \oplus \mathbf{c}_{22}) \oplus \mathbf{c}_{22} = \mathbf{c}_{21} \tag{4.29}$$

For source 1:

$$(\mathbf{c}_{21} \oplus \mathbf{c}_{22}) \oplus \mathbf{c}_{21} = \mathbf{c}_{22} \tag{4.30}$$

As the outer layer has been left uncoded the codewords $\mathbf{c}_{22}$ and $\mathbf{c}_{21}$ are also the datawords.

We can simply extend the work of this subsection to incorporate complex signals if we split the real and imaginary parts and analyse the complex components separately. We can therefore follow the same process using equation (3.41) but treating the complex components separately. For this reason we do not explicitly detail these steps however we will extend this work in the next subsection and choose an integer valued channel and real only signals as another demonstration for recovering the codewords and datawords from each source.

4.6.2 Integer valued channels and real only signals

We now extend the work in the previous subsection to incorporate integer valued channels using the equations highlighting the expected decoded output at each layer established in chapter 3. Again, we assume that each source knows its own information and also has knowledge of channel information, where this is used in order to decode for the individual codewords and datawords from each source. We begin with the inner layer. Using equation (3.50) we can establish the output at layer 0 as:

$$(\mathbf{a}_1[0] \cdot \mathbf{c}_{01}) \oplus (\mathbf{a}_2[0] \cdot \mathbf{c}_{02}) \tag{4.31}$$

and as in subsection 4.6.1, to decode for the codeword from source 1 we can apply the XOR of the known information of source 2:

$$(\mathbf{a}_1[0] \cdot \mathbf{c}_{01}) \oplus (\mathbf{a}_2[0] \cdot \mathbf{c}_{02}) \oplus (\mathbf{a}_2[0] \cdot \mathbf{c}_{02}) = \mathbf{a}_1[0] \cdot \mathbf{c}_{01} \tag{4.32}$$

where this reveals the AND combination of the integer coefficient value with the codeword $\mathbf{a}_1[0]\mathbf{c}_{01}$. We can apply the same concept to decode for the combination of the integer coefficient and codeword originally generated at source 2.

$$(\mathbf{a}_1[0] \cdot \mathbf{c}_{01}) \oplus (\mathbf{a}_2[0] \cdot \mathbf{c}_{02}) \oplus (\mathbf{a}_1[0] \cdot \mathbf{c}_{01}) = \mathbf{a}_2[0] \cdot \mathbf{c}_{02} \tag{4.33}$$

Again, we can determine that the XOR of the codewords also gives us the XOR of the datawords, however now including the integer coefficients $a_{\{1,2\}}[0]$. Depending on the integer $\mathbf{a}_{\{1,2\}}$ values determines whether the individual codewords or indeed datawords can be decoded. For example in equations (4.32) and (4.33) respectively, if $\mathbf{a}_1[0] = 0$ and $\mathbf{a}_2[0] = 0$ then both $\mathbf{c}_{01}$ and $\mathbf{c}_{02}$ cannot be decoded. If both integer coefficients are 1 then both codewords $\mathbf{c}_{0\{1,2\}}$ can be decoded. If we either 1 of the integer coefficients from either source is 1 then only the codeword corresponding to that integer coefficient can be decoded. We see that this will have an effect on decoding at outer layers.

Once the data has been decoded (if possible) then we can re-establish the codeword by putting the dataword back through the original encoder and pass the information to the next layer.

For the middle layer we again use the equation (3.50) to establish the expected output from the decoder on that layer. We can establish that the decoded output is:

$$(\mathbf{a}_1[0]\cdot\mathbf{c}_{11}) \oplus (\mathbf{a}_2[0]\cdot\mathbf{c}_{12}) \oplus (\mathbf{a}_1[1]\cdot\mathbf{c}_{01}) \oplus (\mathbf{a}_2[1]\cdot\mathbf{c}_{02}) \oplus (\mathbf{a}_1[0]\cdot\mathbf{c}_{01})\cdot(\mathbf{a}_2[0]\cdot\mathbf{c}_{02}) \quad (4.34)$$

Following the same process as in subsection 4.6.1 and using known information of the carry term we can decode the combination of codewords from both sources on layer 1.

$$\Big((\mathbf{a}_1[0]\cdot\mathbf{c}_{11}) \oplus (\mathbf{a}_2[0]\cdot\mathbf{c}_{12}) \oplus (\mathbf{a}_1[1]\cdot\mathbf{c}_{01}) \oplus (\mathbf{a}_2[1]\cdot\mathbf{c}_{02}) \oplus (\mathbf{a}_1[0]\cdot\mathbf{c}_{01}) \cdot (\mathbf{a}_2[0]\cdot\mathbf{c}_{02})\Big) \oplus \Big((\mathbf{a}_1[0]\cdot\mathbf{c}_{01}) \cdot (\mathbf{a}_2[0]\cdot\mathbf{c}_{02})\Big) \quad (4.35)$$

This leaves:

$$\Big((\mathbf{a}_1[0]\cdot\mathbf{c}_{11}) \oplus (\mathbf{a}_2[0]\cdot\mathbf{c}_{12}) \oplus (\mathbf{a}_1[1]\cdot\mathbf{c}_{01}) \oplus (\mathbf{a}_2[1]\cdot\mathbf{c}_{02})\Big) \quad (4.36)$$

Therefore we again apply the XOR of $\mathbf{a}_1[1]\cdot\mathbf{c}_{01}$ and $\mathbf{a}_2[1]\cdot\mathbf{c}_{02}$. However this is only possible if $\mathbf{c}_{0\{1,2\}}$ have both been decoded on the previous layer i.e. if $\mathbf{a}_{\{1,2\}}[0]$ are not 0. If decoding was possible then:

$$(\mathbf{a}_1[0]\cdot\mathbf{c}_{11}) \oplus (\mathbf{a}_2[0]\cdot\mathbf{c}_{12}) \oplus (\mathbf{a}_1[1]\cdot\mathbf{c}_{01}) \oplus (\mathbf{a}_2[1]\cdot\mathbf{c}_{02}) \oplus (\mathbf{a}_1[1]\cdot\mathbf{c}_{01}) \oplus (\mathbf{a}_2[1]\cdot\mathbf{c}_{02}) \quad (4.37)$$

Which now leaves:

$$(\mathbf{a}_1[0]\cdot\mathbf{c}_{11}) \oplus (\mathbf{a}_2[0]\cdot\mathbf{c}_{12}) \quad (4.38)$$

At this point we are left with the combination of codewords on layer 1. To decode $\mathbf{a}_1[0]\cdot\mathbf{c}_{11}$ and $\mathbf{a}_2[0]\cdot\mathbf{c}_{12}$ then we apply the known information of the codewords and integer coefficients at each source. At source 1:

$$(\mathbf{a}_1[0]\cdot\mathbf{c}_{11}) \oplus (\mathbf{a}_2[0]\cdot\mathbf{c}_{12}) \oplus (\mathbf{a}_1[0]\cdot\mathbf{c}_{11}) = \mathbf{a}_2[0]\cdot\mathbf{c}_{12} \quad (4.39)$$

At source 2:

$$(\mathbf{a}_1[0]\cdot\mathbf{c}_{11}) \oplus (\mathbf{a}_2[0]\cdot\mathbf{c}_{12}) \oplus (\mathbf{a}_2[0]\cdot\mathbf{c}_{12}) = \mathbf{a}_1[0]\cdot\mathbf{c}_{11} \quad (4.40)$$

As shown for layer 0 decoding for the individual codewords can only be performed if either $\mathbf{a}_{\{1,2\}}[0]$ is equal to 1. If decoding for either codeword is possible then we can decode for the data. As in subsection 4.6.1 the data can be recovered from the code and for general codes if in systematic form. Knowledge of the parity check bits can be removed from the code to reveal the data. This also means that we can add the parity bits to generate the codeword for use at layer 2.

For layer 2, the outer layer, we demonstrate the expected decoded output using (3.50):

$$\begin{gathered}(\mathbf{a}_1[0]\cdot\mathbf{c}_{21})\oplus(\mathbf{a}_2[0]\cdot\mathbf{c}_{22})\oplus(\mathbf{a}_1[1]\cdot\mathbf{c}_{11})\oplus(\mathbf{a}_2[1]\cdot\mathbf{c}_{12})\oplus(\mathbf{a}_1[2]\cdot\mathbf{c}_{01})\oplus(\mathbf{a}_2[2]\cdot\mathbf{c}_{02})\\ \oplus\ \text{andxor2}\ ((\mathbf{a}_1[0]\cdot\mathbf{c}_{11}),\ (\mathbf{a}_2[0]\cdot\mathbf{c}_{12}),\ (\mathbf{a}_1[1]\cdot\mathbf{c}_{01}),\ (\mathbf{a}_2[1]\cdot\mathbf{c}_{02})\oplus\Big((\mathbf{a}_1[0]\cdot\mathbf{c}_{01})\cdot(\mathbf{a}_2[0]\cdot\mathbf{c}_{02})\Big)\end{gathered} \tag{4.41}$$

By applying the XOR of the carry term (second line of eq. 4.41) with equation (4.41) we can uncover the XOR combination of codewords from all three layers with corresponding significant bit.

$$\begin{gathered}(\mathbf{a}_1[0]\cdot\mathbf{c}_{21})\oplus(\mathbf{a}_2[0]\cdot\mathbf{c}_{22})\oplus(\mathbf{a}_1[1]\cdot\mathbf{c}_{11})\oplus(\mathbf{a}_2[1]\cdot\mathbf{c}_{12})\oplus(\mathbf{a}_1[2]\cdot\mathbf{c}_{01})\oplus(\mathbf{a}_2[2]\cdot\mathbf{c}_{02})\\ \oplus\ \text{andxor2}\ \Big((\mathbf{a}_1[0]\cdot\mathbf{c}_{11}),\ (\mathbf{a}_2[0]\cdot\mathbf{c}_{12}),\ (\mathbf{a}_1[1]\cdot\mathbf{c}_{01}),\ (\mathbf{a}_2[1]\cdot\mathbf{c}_{02})\Big)\oplus\Big((\mathbf{a}_1[0]\cdot\mathbf{c}_{01})\cdot(\mathbf{a}_2[0]\cdot\mathbf{c}_{02})\Big)\\ \oplus\ \text{andxor2}\ \Big((\mathbf{a}_1[0]\cdot\mathbf{c}_{11}),\ (\mathbf{a}_2[0]\cdot\mathbf{c}_{12}),\ (\mathbf{a}_1[1]\cdot\mathbf{c}_{01}),\ (\mathbf{a}_2[1]\cdot\mathbf{c}_{02})\Big)\oplus\Big((\mathbf{a}_1[0]\cdot\mathbf{c}_{01})\cdot(\mathbf{a}_2[0]\cdot\mathbf{c}_{02})\Big)\end{gathered} \tag{4.42}$$

This leaves:

$$(\mathbf{a}_1[0]\cdot\mathbf{c}_{21})\oplus(\mathbf{a}_2[0]\cdot\mathbf{c}_{22})\oplus(\mathbf{a}_1[1]\cdot\mathbf{c}_{11})\oplus(\mathbf{a}_2[1]\cdot\mathbf{c}_{12})\oplus(\mathbf{a}_1[2]\cdot\mathbf{c}_{01})\oplus(\mathbf{a}_2[2]\cdot\mathbf{c}_{02}) \tag{4.43}$$

In certain circumstances we may not be able to continue to decode any further than this. For example if we were not able to decode for layer 0 individual codewords and subsequently layer 1 codewords due to integer coefficient bit 0 of either $\mathbf{a}_{\{1,2\}}[0]$ being of value 0 then layer 2 individual codewords would not be able to be decoded. We see this propagation throughout the layers. If we are able to decode then we can now use the known information of the codewords from layer 0 and layer 1 and significant bits of the integer coefficients to decode for the XOR combination of the codewords on layer 2.

$$\begin{aligned}&(\mathbf{a}_1[0]\cdot\mathbf{c}_{21})\oplus(\mathbf{a}_2[0]\cdot\mathbf{c}_{22})\oplus(\mathbf{a}_1[1]\cdot\mathbf{c}_{11})\oplus(\mathbf{a}_2[1]\cdot\mathbf{c}_{12})\oplus(\mathbf{a}_1[2]\cdot\mathbf{c}_{01})\oplus(\mathbf{a}_2[2]\cdot\mathbf{c}_{02})\\&\oplus\Big((\mathbf{a}_1[1]\cdot\mathbf{c}_{11})\oplus(\mathbf{a}_2[1]\cdot\mathbf{c}_{12})\oplus(\mathbf{a}_1[2]\cdot\mathbf{c}_{01})\oplus(\mathbf{a}_2[2]\cdot\mathbf{c}_{02})\Big)\\&=(\mathbf{a}_1[0]\cdot\mathbf{c}_{21})\oplus(\mathbf{a}_2[0]\cdot\mathbf{c}_{22})\end{aligned}\tag{4.44}$$

To decode for $\mathbf{a}_2[0]\cdot\mathbf{c}_{22}$ and $\mathbf{a}_1[0]\cdot\mathbf{c}_{12}$ we apply the known information at each source. At source 1:

$$(\mathbf{a}_1[0]\cdot\mathbf{c}_{21})\oplus(\mathbf{a}_2[0]\cdot\mathbf{c}_{22})\oplus(\mathbf{a}_1[0]\cdot\mathbf{c}_{21})=\mathbf{a}_2[0]\cdot\mathbf{c}_{22}\tag{4.45}$$

At source 2:

$$(\mathbf{a}_1[0]\cdot\mathbf{c}_{21})\oplus(\mathbf{a}_2[0]\cdot\mathbf{c}_{22})\oplus(\mathbf{a}_1[0]\cdot\mathbf{c}_{22})=\mathbf{a}_1[0]\cdot\mathbf{c}_{21}\tag{4.46}$$

Again, if $\mathbf{a}_1[0]=0$ then we are unable to decode for $\mathbf{c}_{21}$ and likewise if $\mathbf{a}_2[0]=0$ then we cannot decode for $\mathbf{c}_{22}$. We can again determine that the XOR of the codewords are the XOR of the datawords. For layer 2 we assume that it is uncoded and therefore the codeword is also the dataword.

The work in this subsection can be straight forwardly extended to integer channels and complex signals by splitting the In-phase and Quadrature components therefore we omit this explanation and move to complex integer valued channels and complex signals.

4.6.3 Complex integer valued channels and complex signals

We now extend the work to incorporate complex integer valued channels and complex signals, employing the same system model of three layers and two users. Known information of the individual codes for each layer and integer coefficients for each layer is again assumed to be available. By using equation (3.63) we can determine the expected output at the decoder on layer 0.

$$\begin{aligned}&\Big((a_1^{\mathfrak{R}}[0]\cdot\mathbf{c}_{01}^{I})\oplus(a_1^{-\mathfrak{I}}[0]\cdot\mathbf{c}_{01}^{Q})\oplus(a_2^{\mathfrak{R}}[0]\cdot\mathbf{c}_{02}^{I})\oplus(a_2^{-\mathfrak{I}}[0]\cdot\mathbf{c}_{02}^{Q})\Big)\\&+\mathbb{i}\Big((a_1^{\mathfrak{R}}[0]\cdot\mathbf{c}_{01}^{Q})\oplus(a_1^{\mathfrak{I}}[0]\cdot\mathbf{c}_{01}^{I})\oplus(a_2^{\mathfrak{R}}[0]\cdot\mathbf{c}_{02}^{Q})\oplus(a_2^{\mathfrak{I}}[0]\cdot\mathbf{c}_{02}^{I})\Big)\end{aligned}\tag{4.47}$$

We split the real and imaginary components and analyse these separately. For the In-phase component we can apply the XOR of the real parts of layer 0 using known information from each source. At source 2 we use source 2 information and decode:

$$\begin{aligned}&\Big((a_1^{\Re}[0]\cdot \mathbf{c}_{01}^{I}) \,\oplus\, (a_1^{-\Im}[0]\cdot \mathbf{c}_{01}^{Q}) \,\oplus\, (a_2^{\Re}[0]\cdot \mathbf{c}_{02}^{I}) \,\oplus\, (a_2^{-\Im}[0]\cdot \mathbf{c}_{02}^{Q})\Big)\\ &\oplus\, \Big((a_2^{\Re}[0]\cdot \mathbf{c}_{02}^{I}) \,\oplus\, (a_2^{-\Im}[0]\cdot \mathbf{c}_{02}^{Q})\Big)\\ &= (a_1^{\Re}[0]\cdot \mathbf{c}_{01}^{I}) \,\oplus\, (a_1^{-\Im}[0]\cdot \mathbf{c}_{01}^{Q})\end{aligned} \tag{4.48}$$

At source 1 we use source 1 information and decode:

$$\begin{aligned}&\Big((a_1^{\Re}[0]\cdot \mathbf{c}_{01}^{I}) \,\oplus\, (a_1^{-\Im}[0]\cdot \mathbf{c}_{01}^{Q}) \,\oplus\, (a_2^{\Re}[0]\cdot \mathbf{c}_{02}^{I}) \,\oplus\, (a_2^{-\Im}[0]\cdot \mathbf{c}_{02}^{Q})\Big)\\ &\oplus\, \Big((a_1^{\Re}[0]\cdot \mathbf{c}_{01}^{I}) \,\oplus\, (a_1^{-\Im}[0]\cdot \mathbf{c}_{01}^{Q})\Big)\\ &= (a_2^{\Re}[0]\cdot \mathbf{c}_{02}^{I}) \,\oplus\, (a_2^{-\Im}[0]\cdot \mathbf{c}_{02}^{Q})\end{aligned} \tag{4.49}$$

For the Quadrature components:

$$\begin{aligned}&\Big((a_1^{\Re}[0]\cdot \mathbf{c}_{01}^{Q}) \,\oplus\, (a_1^{\Im}[0]\cdot \mathbf{c}_{01}^{I}) \,\oplus\, (a_2^{\Re}[0]\cdot \mathbf{c}_{02}^{Q}) \,\oplus\, (a_2^{\Im}[0]\cdot \mathbf{c}_{02}^{I})\Big)\\ &\oplus \Big((a_2^{\Re}[0]\cdot \mathbf{c}_{02}^{Q}) \,\oplus\, (a_2^{\Im}[0]\cdot \mathbf{c}_{02}^{I})\Big)\\ &= (a_1^{\Re}[0]\cdot \mathbf{c}_{01}^{Q}) \oplus (a_1^{\Im}[0]\cdot \mathbf{c}_{01}^{I})\end{aligned} \tag{4.50}$$

$$\begin{aligned}&\Big((a_1^{\Re}[0]\cdot \mathbf{c}_{01}^{Q}) \,\oplus\, (a_1^{\Im}[0]\cdot \mathbf{c}_{01}^{I}) \,\oplus\, (a_2^{\Re}[0]\cdot \mathbf{c}_{02}^{Q}) \,\oplus\, (a_2^{\Im}[0]\cdot \mathbf{c}_{02}^{I})\Big)\\ &\oplus\, \Big((a_1^{\Re}[0]\cdot \mathbf{c}_{01}^{Q}) \,\oplus\, (a_1^{\Im}[0]\cdot \mathbf{c}_{01}^{I})\Big)\\ &= (a_2^{\Re}[0]\cdot \mathbf{c}_{02}^{Q}) \,\oplus\, (a_2^{\Im}[0]\cdot \mathbf{c}_{02}^{I})\end{aligned} \tag{4.51}$$

Each source knows information of the integer coefficients where this can be used to decode for the individual complex components of the codewords in equations (4.48 - 4.51). For example at source 1, using the resulting expression (4.51), if we know that $a_2^{\Re}[0] = 0$ and $a_2^{\Im}[0] = 1$ then we can determine $\mathbf{c}_{02}^{I}$. If we apply $a_2^{\Im}[0] = 0$ and $a_2^{\Re}[0] = 1$ then we can determine $\mathbf{c}_{02}^{Q}$. We also see that there is common information of $\mathbf{a}_2^{\Re}[0]$ in both (4.49) and (4.51) and common information of $\mathbf{a}_1^{\Re}[0]$ in (4.48) and (4.50) which if $\mathbf{a}_{\{1,2\}}^{\Re}[0] = 0$ and components $\mathbf{a}_2^{-\Im}[0] = 1$ and $\mathbf{a}_1^{\Im}[0] = 1$ then we can successfully decode for $\mathbf{c}_{0\{1,2\}}^{\{I,Q\}}$. However, at each source, if more than one component of $\mathbf{a}_{\{1,2\}}^{\{-\Im,\Im,\Re\}}[0]$ is 0 then both components of the codewords cannot be decoded, unless $\mathbf{a}_{\{1,2\}}^{\Re}[0] = 1$ because of common information.

As in the real case the combination of codewords also determines the combination of datawords, therefore the same process can be applied to:

$$(a_1^{\Re}[0] \cdot \mathbf{w}_{01}^{I}) \ \oplus \ (a_1^{-\Im}[0] \cdot \mathbf{w}_{01}^{Q}) \tag{4.52}$$

$$(a_2^{\Re}[0] \cdot \mathbf{w}_{02}^{I}) \ \oplus \ (a_2^{-\Im}[0] \cdot \mathbf{w}_{02}^{Q}) \tag{4.53}$$

$$(a_1^{\Re}[0] \cdot \mathbf{w}_{01}^{Q}) \ \oplus \ (a_1^{\Im}[0] \cdot \mathbf{w}_{01}^{I}) \tag{4.54}$$

$$(a_2^{\Re}[0] \cdot \mathbf{w}_{02}^{Q}) \ \oplus \ (a_2^{\Im}[0] \cdot \mathbf{w}_{02}^{I}) \tag{4.55}$$

With the information regarding the data recovered it is straight forward to subject this to the original encoder and integer coefficient to generate the codewords again.

For layer 1 the expected decoded output using (3.63) for the In-phase component is:

$$\begin{aligned}
&\Big((a_1^{\Re}[1] \cdot \mathbf{c}_{01}^{I}) \oplus (a_1^{-\Im}[1] \cdot \mathbf{c}_{01}^{Q}) \oplus (a_2^{\Re}[1] \cdot \mathbf{c}_{02}^{I}) \oplus (a_2^{-\Im}[1] \cdot \mathbf{c}_{02}^{Q}) \\
&\oplus (a_1^{\Re}[0] \cdot \mathbf{c}_{11}^{I}) \oplus (a_1^{-\Im}[0] \cdot \mathbf{c}_{11}^{Q}) \oplus (a_2^{\Re}[0] \cdot \mathbf{c}_{12}^{I}) \oplus (a_2^{-\Im}[0] \cdot \mathbf{c}_{12}^{Q})\Big) \\
&\oplus \text{andxor2}\Big((a_1^{\Re}[0] \cdot \mathbf{c}_{01}^{I}),\ (a_1^{-\Im}[0] \cdot \mathbf{c}_{01}^{Q}),\ (a_2^{\Re}[0] \cdot \mathbf{c}_{02}^{I}),\ (a_2^{-\Im}[0] \cdot \mathbf{c}_{02}^{Q})\Big)
\end{aligned} \tag{4.56}$$

if we apply the XOR of the carry term:

$$\begin{aligned}
&\Big((a_1^{\Re}[1] \cdot \mathbf{c}_{01}^{I}) \oplus (a_1^{-\Im}[1] \cdot \mathbf{c}_{01}^{Q}) \oplus (a_2^{\Re}[1] \cdot \mathbf{c}_{02}^{I}) \oplus (a_2^{-\Im}[1] \cdot \mathbf{c}_{02}^{Q}) \\
&\oplus (a_1^{\Re}[0] \cdot \mathbf{c}_{11}^{I}) \oplus (a_1^{-\Im}[0] \cdot \mathbf{c}_{11}^{Q}) \oplus (a_2^{\Re}[0] \cdot \mathbf{c}_{12}^{I}) \oplus (a_2^{-\Im}[0] \cdot \mathbf{c}_{12}^{Q})\Big) \\
&\oplus \text{andxor2}\Big((a_1^{\Re}[0] \cdot \mathbf{c}_{01}^{I}),\ (a_1^{-\Im}[0] \cdot \mathbf{c}_{01}^{Q}),\ (a_2^{\Re}[0] \cdot \mathbf{c}_{02}^{I}),\ (a_2^{-\Im}[0] \cdot \mathbf{c}_{02}^{Q})\Big) \\
&\oplus \text{andxor2}\Big((a_1^{\Re}[0] \cdot \mathbf{c}_{01}^{I}),\ (a_1^{-\Im}[0] \cdot \mathbf{c}_{01}^{Q}),\ (a_2^{\Re}[0] \cdot \mathbf{c}_{02}^{I}),\ (a_2^{-\Im}[0] \cdot \mathbf{c}_{02}^{Q})\Big)
\end{aligned} \tag{4.57}$$

This leaves:

$$\begin{aligned}
&\Big((a_1^{\Re}[1] \cdot \mathbf{c}_{01}^{I}) \oplus (a_1^{-\Im}[1] \cdot \mathbf{c}_{01}^{Q}) \oplus (a_2^{\Re}[1] \cdot \mathbf{c}_{02}^{I}) \oplus (a_2^{-\Im}[1] \cdot \mathbf{c}_{02}^{Q}) \\
&\oplus (a_1^{\Re}[0] \cdot \mathbf{c}_{11}^{I}) \oplus (a_1^{-\Im}[0] \cdot \mathbf{c}_{11}^{Q}) \oplus (a_2^{\Re}[0] \cdot \mathbf{c}_{12}^{I}) \oplus (a_2^{-\Im}[0] \cdot \mathbf{c}_{12}^{Q})\Big)
\end{aligned} \tag{4.58}$$

where we can apply the XOR of codewords from layer 0 and known information of the integer coefficients to decode for the codes from source 1 and 2. This is only possible if both complex components of $\mathbf{c}_{01}$ and $\mathbf{c}_{02}$ have been decoded at the previous layer.

$$\begin{aligned}&\Big((a_1^{\Re}[1]\cdot\mathbf{c}_{01}^{I})\oplus(a_1^{-\Im}[1]\cdot\mathbf{c}_{01}^{Q})\oplus(a_2^{\Re}[1]\cdot\mathbf{c}_{02}^{I})\oplus(a_2^{-\Im}[1]\cdot\mathbf{c}_{02}^{Q})\\&\oplus(a_1^{\Re}[0]\cdot\mathbf{c}_{11}^{I})\oplus(a_1^{-\Im}[0]\cdot\mathbf{c}_{11}^{Q})\oplus(a_2^{\Re}[0]\cdot\mathbf{c}_{12}^{I})\oplus(a_2^{-\Im}[0]\cdot\mathbf{c}_{12}^{Q})\Big)\\&\oplus\Big((a_1^{\Re}[1]\cdot\mathbf{c}_{01}^{I})\oplus(a_1^{-\Im}[1]\cdot\mathbf{c}_{01}^{Q})\oplus(a_2^{\Re}[1]\cdot\mathbf{c}_{02}^{I})\oplus(a_2^{-\Im}[1]\cdot\mathbf{c}_{02}^{Q})\Big)\end{aligned}\tag{4.59}$$

leaving:

$$\Big((a_1^{\Re}[0]\cdot\mathbf{c}_{11}^{I})\oplus(a_1^{-\Im}[0]\cdot\mathbf{c}_{11}^{Q})\oplus(a_2^{\Re}[0]\cdot\mathbf{c}_{12}^{I})\oplus(a_2^{-\Im}[0]\cdot\mathbf{c}_{12}^{Q})\Big)\tag{4.60}$$

where the codewords can be recovered using the sources known information of the codewords. At source 2:

$$\begin{aligned}&\Big((a_1^{\Re}[0]\cdot\mathbf{c}_{11}^{I})\ \oplus\ (a_1^{-\Im}[0]\cdot\mathbf{c}_{11}^{Q})\ \oplus\ (a_2^{\Re}[0]\cdot\mathbf{c}_{12}^{I})\ \oplus\ (a_2^{-\Im}[0]\cdot\mathbf{c}_{12}^{Q})\Big)\\&\oplus\ \Big((a_2^{\Re}[0]\cdot\mathbf{c}_{12}^{I})\ \oplus\ (a_2^{-\Im}[0]\cdot\mathbf{c}_{12}^{Q})\Big)=(a_1^{\Re}[0]\cdot\mathbf{c}_{11}^{I})\ \oplus\ (a_1^{-\Im}[0]\cdot\mathbf{c}_{11}^{Q})\end{aligned}\tag{4.61}$$

At source 1:

$$\begin{aligned}&\Big((a_1^{\Re}[0]\cdot\mathbf{c}_{11}^{I})\oplus(a_1^{-\Im}[0]\cdot\mathbf{c}_{11}^{Q})\oplus(a_2^{\Re}[0]\cdot\mathbf{c}_{12}^{I})\oplus(a_2^{-\Im}[0]\cdot\mathbf{c}_{12}^{Q})\Big)\\&\oplus\Big((a_1^{\Re}[0]\cdot\mathbf{c}_{11}^{I})\oplus(a_1^{-\Im}[0]\cdot\mathbf{c}_{11}^{Q})\Big)=(a_2^{\Re}[0]\cdot\mathbf{c}_{12}^{I})\ \oplus\ (a_2^{-\Im}[0]\cdot\mathbf{c}_{12}^{Q})\end{aligned}\tag{4.62}$$

For the Quadrature component the decoded output is:

$$\begin{aligned}&\Big((a_1^{\Re}[1]\cdot\mathbf{c}_{01}^{Q})\oplus(a_1^{\Im}[1]\cdot\mathbf{c}_{01}^{I})\oplus(a_2^{\Re}[1]\cdot\mathbf{c}_{02}^{Q})\oplus(a_2^{\Im}[1]\cdot\mathbf{c}_{02}^{I})\\&\oplus(a_1^{\Re}[0]\cdot\mathbf{c}_{11}^{Q})\oplus(a_1^{\Im}[0]\cdot\mathbf{c}_{11}^{I})\oplus(a_2^{\Re}[0]\cdot\mathbf{c}_{12}^{Q})\oplus(a_2^{\Im}[0]\cdot\mathbf{c}_{12}^{I})\Big)\\&\oplus\text{andxor2}\Big((a_1^{\Re}[0]\cdot\mathbf{c}_{01}^{Q}),\ (a_1^{\Im}[0]\cdot\mathbf{c}_{01}^{I}),\ (a_2^{\Re}[0]\cdot\mathbf{c}_{02}^{Q}),\ (a_2^{\Im}[0]\cdot\mathbf{c}_{02}^{I})\Big)\end{aligned}\tag{4.63}$$

We proceed with similar steps to that of the In-phase component and apply the XOR of the carry term.

$$\begin{aligned}&\Big((a_1^{\Re}[1]\cdot\mathbf{c}_{01}^{Q})\oplus(a_1^{\Im}[1]\cdot\mathbf{c}_{01}^{I})\oplus(a_2^{\Re}[1]\cdot\mathbf{c}_{02}^{Q})\oplus(a_2^{\Im}[1]\cdot\mathbf{c}_{02}^{I})\\&\oplus(a_1^{\Re}[0]\cdot\mathbf{c}_{11}^{Q})\oplus(a_1^{\Im}[0]\cdot\mathbf{c}_{11}^{I})\oplus(a_2^{\Re}[0]\cdot\mathbf{c}_{12}^{Q})\oplus(a_2^{\Im}[0]\cdot\mathbf{c}_{12}^{I})\Big)\\&\oplus\text{andxor2}\Big((a_1^{\Re}[0]\cdot\mathbf{c}_{01}^{Q}),\ (a_1^{\Im}[0]\cdot\mathbf{c}_{01}^{I}),\ (a_2^{\Re}[0]\cdot\mathbf{c}_{02}^{Q}),\ (a_2^{\Im}[0]\cdot\mathbf{c}_{02}^{I})\Big)\\&\oplus\text{andxor2}\Big((a_1^{\Re}[0]\cdot\mathbf{c}_{01}^{Q}),\ (a_1^{\Im}[0]\cdot\mathbf{c}_{01}^{I}),\ (a_2^{\Re}[0]\cdot\mathbf{c}_{02}^{Q}),\ (a_2^{\Im}[0]\cdot\mathbf{c}_{02}^{I})\Big)\end{aligned}\tag{4.64}$$

leaving:

$$\begin{aligned}&\Big((a_1^{\mathfrak{R}}[1]\cdot\mathbf{c}_{01}^{Q})\oplus(a_1^{\mathfrak{I}}[1]\cdot\mathbf{c}_{01}^{I})\oplus(a_2^{\mathfrak{R}}[1]\cdot\mathbf{c}_{02}^{Q})\oplus(a_2^{\mathfrak{I}}[1]\cdot\mathbf{c}_{02}^{I})\\&\oplus(a_1^{\mathfrak{R}}[0]\cdot\mathbf{c}_{11}^{Q})\oplus(a_1^{\mathfrak{I}}[0]\cdot\mathbf{c}_{11}^{I})\oplus(a_2^{\mathfrak{R}}[0]\cdot\mathbf{c}_{12}^{Q})\oplus(a_2^{\mathfrak{I}}[0]\cdot\mathbf{c}_{12}^{I})\Big)\end{aligned}\tag{4.65}$$

where again we can apply the XOR of the known information from layer 0 to decode for the Quadrature components of the codewords on layer 1. Again, this is only possible if both complex components from both sources on layer 0 have been successfully decoded.

$$\begin{aligned}&\Big((a_1^{\mathfrak{R}}[1]\cdot\mathbf{c}_{01}^{Q})\oplus(a_1^{\mathfrak{I}}[1]\cdot\mathbf{c}_{01}^{I})\oplus(a_2^{\mathfrak{R}}[1]\cdot\mathbf{c}_{02}^{Q})\oplus(a_2^{\mathfrak{I}}[1]\cdot\mathbf{c}_{02}^{I})\\&\oplus(a_1^{\mathfrak{R}}[0]\cdot\mathbf{c}_{11}^{Q})\oplus(a_1^{\mathfrak{I}}[0]\cdot\mathbf{c}_{11}^{I})\oplus(a_2^{\mathfrak{R}}[0]\cdot\mathbf{c}_{12}^{Q})\oplus(a_2^{\mathfrak{I}}[0]\cdot\mathbf{c}_{12}^{I})\Big)\\&\oplus\Big((a_1^{\mathfrak{R}}[1]\cdot\mathbf{c}_{01}^{Q})\oplus(a_1^{\mathfrak{I}}[1]\cdot\mathbf{c}_{01}^{I})\oplus(a_2^{\mathfrak{R}}[1]\cdot\mathbf{c}_{02}^{Q})\oplus(a_2^{\mathfrak{I}}[1]\cdot\mathbf{c}_{02}^{I})\Big)\end{aligned}\tag{4.66}$$

leaving:

$$\Big((a_1^{\mathfrak{R}}[0]\cdot\mathbf{c}_{11}^{Q})\oplus(a_1^{\mathfrak{I}}[0]\cdot\mathbf{c}_{11}^{I})\oplus(a_2^{\mathfrak{R}}[0]\cdot\mathbf{c}_{12}^{Q})\oplus(a_2^{\mathfrak{I}}[0]\cdot\mathbf{c}_{12}^{I})\Big)\tag{4.67}$$

Again, we can apply the XOR of the known information from each source to decode the codewords and their complex components.

At source 2:

$$\begin{aligned}&\Big((a_1^{\mathfrak{R}}[0]\cdot\mathbf{c}_{11}^{Q})\ \oplus\ (a_1^{\mathfrak{I}}[0]\cdot\mathbf{c}_{11}^{I})\ \oplus\ (a_2^{\mathfrak{R}}[0]\cdot\mathbf{c}_{12}^{Q})\ \oplus\ (a_2^{\mathfrak{I}}[0]\cdot\mathbf{c}_{12}^{I})\Big)\\&\oplus\Big((a_2^{\mathfrak{R}}[0]\cdot\mathbf{c}_{12}^{Q})\ \oplus\ (a_2^{\mathfrak{I}}[0]\cdot\mathbf{c}_{12}^{I})\Big)=(a_1^{\mathfrak{R}}[0]\cdot\mathbf{c}_{11}^{Q})\ \oplus\ (a_1^{\mathfrak{I}}[0]\cdot\mathbf{c}_{11}^{I})\end{aligned}\tag{4.68}$$

At source 1:

$$\begin{aligned}&\Big((a_1^{\mathfrak{R}}[0]\cdot\mathbf{c}_{11}^{Q})\ \oplus\ (a_1^{\mathfrak{I}}[0]\cdot\mathbf{c}_{11}^{I})\ \oplus\ (a_2^{\mathfrak{R}}[0]\cdot\mathbf{c}_{12}^{Q})\ \oplus\ (a_2^{\mathfrak{I}}[0]\cdot\mathbf{c}_{12}^{I})\Big)\\&\oplus\Big((a_1^{\mathfrak{R}}[0]\cdot\mathbf{c}_{11}^{Q})\ \oplus\ (a_1^{\mathfrak{I}}[0]\cdot\mathbf{c}_{11}^{I})\Big)=(a_2^{\mathfrak{R}}[0]\cdot\mathbf{c}_{12}^{Q})\ \oplus\ (a_2^{\mathfrak{I}}[0]\cdot\mathbf{c}_{12}^{I})\end{aligned}\tag{4.69}$$

The individual components can be decoded using known information at each source. The decoding process can therefore also be carried out on the combination of data messages to decode the individual components of the data. This can then be encoded again to be passed to layer 2. As shown at layer 0 we can only decode for the individual codewords and their complex components under certain conditions. The same applies to this layer using (4.61), (4.62), (4.68) and (4.69).

For layer 2 we begin with the expected decoded output using (3.63) for the In-phase component:

$$
\begin{gathered}
\Big((a_1^{\mathfrak{R}}[2]\cdot\mathbf{c}_{01}^{I}) \oplus (a_1^{-\mathfrak{I}}[2]\cdot\mathbf{c}_{01}^{Q}) \oplus (a_2^{\mathfrak{R}}[2]\cdot\mathbf{c}_{02}^{I}) \oplus (a_2^{-\mathfrak{I}}[2]\cdot\mathbf{c}_{02}^{Q})\\
\oplus (a_1^{\mathfrak{R}}[1]\cdot\mathbf{c}_{11}^{I}) \oplus (a_1^{-\mathfrak{I}}[1]\cdot\mathbf{c}_{11}^{Q}) \oplus (a_2^{\mathfrak{R}}[1]\cdot\mathbf{c}_{12}^{I}) \oplus (a_2^{-\mathfrak{I}}[1]\cdot\mathbf{c}_{12}^{Q})\\
\oplus (a_1^{\mathfrak{R}}[0]\cdot\mathbf{c}_{21}^{I}) \oplus (a_1^{-\mathfrak{I}}[0]\cdot\mathbf{c}_{21}^{Q}) \oplus (a_2^{\mathfrak{R}}[0]\cdot\mathbf{c}_{22}^{I}) \oplus (a_2^{-\mathfrak{I}}[0]\cdot\mathbf{c}_{22}^{Q})\Big)\\
\oplus \,\mathrm{andxor2}\Big((a_1^{\mathfrak{R}}[0]\cdot\mathbf{c}_{11}^{I}),\ (a_1^{-\mathfrak{I}}[0]\cdot\mathbf{c}_{11}^{Q}),\ (a_2^{\mathfrak{R}}[0]\cdot\mathbf{c}_{12}^{I}),\ (a_2^{-\mathfrak{I}}[0]\cdot\mathbf{c}_{12}^{Q}),\\
(a_1^{\mathfrak{R}}[1]\cdot\mathbf{c}_{01}^{I}),\ (a_1^{-\mathfrak{I}}[1]\cdot\mathbf{c}_{01}^{Q}),\ (a_2^{\mathfrak{R}}[1]\cdot\mathbf{c}_{02}^{I}),\ (a_2^{-\mathfrak{I}}[1]\cdot\mathbf{c}_{02}^{Q})\Big)\\
\oplus \,\mathrm{andxor2}\Big((a_1^{\mathfrak{R}}[0]\cdot\mathbf{c}_{01}^{I}),\ (a_1^{-\mathfrak{I}}[0]\cdot\mathbf{c}_{01}^{Q}),\ (a_2^{\mathfrak{R}}[0]\cdot\mathbf{c}_{02}^{I}),\ (a_2^{-\mathfrak{I}}[0]\cdot\mathbf{c}_{02}^{Q})\Big)
\end{gathered}
\tag{4.70}
$$

We can apply the XOR of the carry terms to decode the XOR of the combination of codewords on all three layers.

$$
\begin{gathered}
\Big((a_1^{\mathfrak{R}}[2]\cdot\mathbf{c}_{01}^{I}) \oplus (a_1^{-\mathfrak{I}}[2]\cdot\mathbf{c}_{01}^{Q}) \oplus (a_2^{\mathfrak{R}}[2]\cdot\mathbf{c}_{02}^{I}) \oplus (a_2^{-\mathfrak{I}}[2]\cdot\mathbf{c}_{02}^{Q})\\
\oplus (a_1^{\mathfrak{R}}[1]\cdot\mathbf{c}_{11}^{I}) \oplus (a_1^{-\mathfrak{I}}[1]\cdot\mathbf{c}_{11}^{Q}) \oplus (a_2^{\mathfrak{R}}[1]\cdot\mathbf{c}_{12}^{I}) \oplus (a_2^{-\mathfrak{I}}[1]\cdot\mathbf{c}_{12}^{Q})\\
\oplus (a_1^{\mathfrak{R}}[0]\cdot\mathbf{c}_{21}^{I}) \oplus (a_1^{-\mathfrak{I}}[0]\cdot\mathbf{c}_{21}^{Q}) \oplus (a_2^{\mathfrak{R}}[0]\cdot\mathbf{c}_{22}^{I}) \oplus (a_2^{-\mathfrak{I}}[0]\cdot\mathbf{c}_{22}^{Q})\Big)\\
\oplus \,\mathrm{andxor2}\Big((a_1^{\mathfrak{R}}[0]\cdot\mathbf{c}_{11}^{I}),\ (a_1^{-\mathfrak{I}}[0]\cdot\mathbf{c}_{11}^{Q}),\ (a_2^{\mathfrak{R}}[0]\cdot\mathbf{c}_{12}^{I}),\ (a_2^{-\mathfrak{I}}[0]\cdot\mathbf{c}_{12}^{Q}),\\
(a_1^{\mathfrak{R}}[1]\cdot\mathbf{c}_{01}^{I}),\ (a_1^{-\mathfrak{I}}[1]\cdot\mathbf{c}_{01}^{Q}),\ (a_2^{\mathfrak{R}}[1]\cdot\mathbf{c}_{02}^{I}),\ (a_2^{-\mathfrak{I}}[1]\cdot\mathbf{c}_{02}^{Q})\Big)\\
\oplus \,\mathrm{andxor2}\Big((a_1^{\mathfrak{R}}[0]\cdot\mathbf{c}_{01}^{I}),\ (a_1^{-\mathfrak{I}}[0]\cdot\mathbf{c}_{01}^{Q}),\ (a_2^{\mathfrak{R}}[0]\cdot\mathbf{c}_{02}^{I}),\ (a_2^{-\mathfrak{I}}[0]\cdot\mathbf{c}_{02}^{Q})\Big)\\
\oplus \,\mathrm{andxor2}\Big((a_1^{\mathfrak{R}}[0]\cdot\mathbf{c}_{11}^{I}),\ (a_1^{-\mathfrak{I}}[0]\cdot\mathbf{c}_{11}^{Q}),\ (a_2^{\mathfrak{R}}[0]\cdot\mathbf{c}_{12}^{I}),\ (a_2^{-\mathfrak{I}}[0]\cdot\mathbf{c}_{12}^{Q}),\\
(a_1^{\mathfrak{R}}[1]\cdot\mathbf{c}_{01}^{I}),\ (a_1^{-\mathfrak{I}}[1]\cdot\mathbf{c}_{01}^{Q}),\ (a_2^{\mathfrak{R}}[1]\cdot\mathbf{c}_{02}^{I}),\ (a_2^{-\mathfrak{I}}[1]\cdot\mathbf{c}_{02}^{Q})\Big)\\
\oplus \,\mathrm{andxor2}\Big((a_1^{\mathfrak{R}}[0]\cdot\mathbf{c}_{01}^{I}),\ (a_1^{-\mathfrak{I}}[0]\cdot\mathbf{c}_{01}^{Q}),\ (a_2^{\mathfrak{R}}[0]\cdot\mathbf{c}_{02}^{I}),\ (a_2^{-\mathfrak{I}}[0]\cdot\mathbf{c}_{02}^{Q})\Big)
\end{gathered}
\tag{4.71}
$$

which leaves:

$$
\begin{gathered}
\Big((a_1^{\mathfrak{R}}[2]\cdot\mathbf{c}_{01}^{I}) \oplus (a_1^{-\mathfrak{I}}[2]\cdot\mathbf{c}_{01}^{Q}) \oplus (a_2^{\mathfrak{R}}[2]\cdot\mathbf{c}_{02}^{I}) \oplus (a_2^{-\mathfrak{I}}[2]\cdot\mathbf{c}_{02}^{Q})\\
\oplus (a_1^{\mathfrak{R}}[1]\cdot\mathbf{c}_{11}^{I}) \oplus (a_1^{-\mathfrak{I}}[1]\cdot\mathbf{c}_{11}^{Q}) \oplus (a_2^{\mathfrak{R}}[1]\cdot\mathbf{c}_{12}^{I}) \oplus (a_2^{-\mathfrak{I}}[1]\cdot\mathbf{c}_{12}^{Q})\\
\oplus (a_1^{\mathfrak{R}}[0]\cdot\mathbf{c}_{21}^{I}) \oplus (a_1^{-\mathfrak{I}}[0]\cdot\mathbf{c}_{21}^{Q}) \oplus (a_2^{\mathfrak{R}}[0]\cdot\mathbf{c}_{22}^{I}) \oplus (a_2^{-\mathfrak{I}}[0]\cdot\mathbf{c}_{22}^{Q})\Big)
\end{gathered}
\tag{4.72}
$$

where successfully decoded and therefore known information from layer 0 and layer 1 has been passed on and we can perform the XOR operation of those with the above result. If the previous layers have not been successfully decoded then the codewords of layer 2 cannot be decoded.

$$\begin{aligned}&\Big((a_1^{\Re}[2]\cdot\mathbf{c}_{01}^{I})\oplus(a_1^{-\Im}[2]\cdot\mathbf{c}_{01}^{Q})\oplus(a_2^{\Re}[2]\cdot\mathbf{c}_{02}^{I})\oplus(a_2^{-\Im}[2]\cdot\mathbf{c}_{02}^{Q})\\&\oplus(a_1^{\Re}[1]\cdot\mathbf{c}_{11}^{I})\oplus(a_1^{-\Im}[1]\cdot\mathbf{c}_{11}^{Q})\oplus(a_2^{\Re}[1]\cdot\mathbf{c}_{12}^{I})\oplus(a_2^{-\Im}[1]\cdot\mathbf{c}_{12}^{Q})\\&\oplus(a_1^{\Re}[0]\cdot\mathbf{c}_{21}^{I})\oplus(a_1^{-\Im}[0]\cdot\mathbf{c}_{21}^{Q})\oplus(a_2^{\Re}[0]\cdot\mathbf{c}_{22}^{I})\oplus(a_2^{-\Im}[0]\cdot\mathbf{c}_{22}^{Q})\Big)\\&\oplus\Big((a_1^{\Re}[2]\cdot\mathbf{c}_{01}^{I})\oplus(a_1^{-\Im}[2]\cdot\mathbf{c}_{01}^{Q})\oplus(a_2^{\Re}[2]\cdot\mathbf{c}_{02}^{I})\oplus(a_2^{-\Im}[2]\cdot\mathbf{c}_{02}^{Q})\\&\oplus(a_1^{\Re}[1]\cdot\mathbf{c}_{11}^{I})\oplus(a_1^{-\Im}[1]\cdot\mathbf{c}_{11}^{Q})\oplus(a_2^{\Re}[1]\cdot\mathbf{c}_{12}^{I})\oplus(a_2^{-\Im}[1]\cdot\mathbf{c}_{12}^{Q})\Big)\end{aligned}\tag{4.73}$$

This leaves:

$$(a_1^{\Re}[0]\cdot\mathbf{c}_{21}^{I})\oplus(a_1^{-\Im}[0]\cdot\mathbf{c}_{21}^{Q})\oplus(a_2^{\Re}[0]\cdot\mathbf{c}_{22}^{I})\oplus(a_2^{-\Im}[0]\cdot\mathbf{c}_{22}^{Q})\tag{4.74}$$

where the combination of the codewords can be decoded by the following.
At source 2:

$$\begin{aligned}&(a_1^{\Re}[0]\cdot\mathbf{c}_{21}^{I})\ \oplus\ (a_1^{-\Im}[0]\cdot\mathbf{c}_{21}^{Q})\ \oplus\ (a_2^{\Re}[0]\cdot\mathbf{c}_{22}^{I})\ \oplus\ (a_2^{-\Im}[0]\cdot\mathbf{c}_{22}^{Q})\\&\oplus\ \Big((a_2^{\Re}[0]\cdot\mathbf{c}_{22}^{I})\ \oplus\ (a_2^{-\Im}[0]\cdot\mathbf{c}_{22}^{Q})\Big)=(a_1^{\Re}[0]\cdot\mathbf{c}_{21}^{I})\oplus(a_1^{-\Im}[0]\cdot\mathbf{c}_{21}^{Q})\end{aligned}\tag{4.75}$$

At source 1:

$$\begin{aligned}&(a_1^{\Re}[0]\cdot\mathbf{c}_{21}^{I})\ \oplus\ (a_1^{-\Im}[0]\cdot\mathbf{c}_{21}^{Q})\ \oplus\ (a_2^{\Re}[0]\cdot\mathbf{c}_{22}^{I})\ \oplus\ (a_2^{-\Im}[0]\cdot\mathbf{c}_{22}^{Q})\\&\oplus\Big((a_1^{\Re}[0]\cdot\mathbf{c}_{21}^{I})\ \oplus\ (a_1^{-\Im}[0]\cdot\mathbf{c}_{21}^{Q})\Big)=(a_2^{\Re}[0]\cdot\mathbf{c}_{22}^{I})\oplus(a_2^{-\Im}[0]\cdot\mathbf{c}_{22}^{Q})\end{aligned}\tag{4.76}$$

Again we can determine that the XOR of the codewords is also the XOR of the datawords, however in this case the outer layer is uncoded and therefore the codewords are also the datawords. We now complete the same process for the Quadrature component. Starting with the expected decoded output on this layer.

$$
\begin{aligned}
&\Big((a_1^{\Re}[2]\cdot\mathbf{c}_{01}^{Q}) \oplus (a_1^{\Im}[2]\cdot\mathbf{c}_{01}^{I}) \oplus (a_2^{\Re}[2]\cdot\mathbf{c}_{02}^{Q}) \oplus (a_2^{\Im}[2]\cdot\mathbf{c}_{02}^{I})\\
&\oplus (a_1^{\Re}[1]\cdot\mathbf{c}_{11}^{Q}) \oplus (a_1^{\Im}[1]\cdot\mathbf{c}_{11}^{I}) \oplus (a_2^{\Re}[1]\cdot\mathbf{c}_{12}^{Q}) \oplus (a_2^{\Im}[1]\cdot\mathbf{c}_{12}^{I})\\
&\oplus (a_1^{\Re}[0]\cdot\mathbf{c}_{21}^{Q}) \oplus (a_1^{\Im}[0]\cdot\mathbf{c}_{21}^{I}) \oplus (a_2^{\Re}[0]\cdot\mathbf{c}_{22}^{Q}) \oplus (a_2^{\Im}[0]\cdot\mathbf{c}_{22}^{I})\Big)\\
&\oplus \mathrm{andxor2}\Big((a_1^{\Re}[0]\cdot\mathbf{c}_{11}^{Q}),\ (a_1^{\Im}[0]\cdot\mathbf{c}_{11}^{I}),\ (a_2^{\Re}[0]\cdot\mathbf{c}_{12}^{Q}),\ (a_2^{\Im}[0]\cdot\mathbf{c}_{12}^{I}),\\
&(a_1^{\Re}[1]\cdot\mathbf{c}_{01}^{Q}),\ (a_1^{\Im}[1]\cdot\mathbf{c}_{01}^{I}),\ (a_2^{\Re}[1]\cdot\mathbf{c}_{02}^{Q}),\ (a_2^{\Im}[1]\cdot\mathbf{c}_{02}^{I})\Big)\\
&\oplus \mathrm{andxor2}\Big((a_1^{\Re}[0]\cdot\mathbf{c}_{01}^{Q}),\ (a_1^{\Im}[0]\cdot\mathbf{c}_{01}^{I}),\ (a_2^{\Re}[0]\cdot\mathbf{c}_{02}^{Q}),\ (a_2^{\Im}[0]\cdot\mathbf{c}_{02}^{I})\Big)
\end{aligned}
\tag{4.77}
$$

We can apply the XOR of the carry term:

$$
\begin{aligned}
&\Big((a_1^{\Re}[2]\cdot\mathbf{c}_{01}^{Q}) \oplus (a_1^{\Im}[2]\cdot\mathbf{c}_{01}^{I}) \oplus (a_2^{\Re}[2]\cdot\mathbf{c}_{02}^{Q}) \oplus (a_2^{\Im}[2]\cdot\mathbf{c}_{02}^{I})\\
&\oplus (a_1^{\Re}[1]\cdot\mathbf{c}_{11}^{Q}) \oplus (a_1^{\Im}[1]\cdot\mathbf{c}_{11}^{I}) \oplus (a_2^{\Re}[1]\cdot\mathbf{c}_{12}^{Q}) \oplus (a_2^{\Im}[1]\cdot\mathbf{c}_{12}^{I})\\
&\oplus (a_1^{\Re}[0]\cdot\mathbf{c}_{21}^{Q}) \oplus (a_1^{\Im}[0]\cdot\mathbf{c}_{21}^{I}) \oplus (a_2^{\Re}[0]\cdot\mathbf{c}_{22}^{Q}) \oplus (a_2^{\Im}[0]\cdot\mathbf{c}_{22}^{I})\Big)\\
&\oplus \mathrm{andxor2}\Big((a_1^{\Re}[0]\cdot\mathbf{c}_{11}^{Q}),\ (a_1^{\Im}[0]\cdot\mathbf{c}_{11}^{I}),\ (a_2^{\Re}[0]\cdot\mathbf{c}_{12}^{Q}),\ (a_2^{\Im}[0]\cdot\mathbf{c}_{12}^{I}),\\
&(a_1^{\Re}[1]\cdot\mathbf{c}_{01}^{Q}),\ (a_1^{\Im}[1]\cdot\mathbf{c}_{01}^{I}),\ (a_2^{\Re}[1]\cdot\mathbf{c}_{02}^{Q}),\ (a_2^{\Im}[1]\cdot\mathbf{c}_{02}^{I})\Big)\\
&\oplus \mathrm{andxor2}\Big((a_1^{\Re}[0]\cdot\mathbf{c}_{01}^{Q}),\ (a_1^{\Im}[0]\cdot\mathbf{c}_{01}^{I}),\ (a_2^{\Re}[0]\cdot\mathbf{c}_{02}^{Q}),\ (a_2^{\Im}[0]\cdot\mathbf{c}_{02}^{I})\Big)\\
&\oplus \mathrm{andxor2}\Big((a_1^{\Re}[0]\cdot\mathbf{c}_{11}^{Q}),\ (a_1^{\Im}[0]\cdot\mathbf{c}_{11}^{I}),\ (a_2^{\Re}[0]\cdot\mathbf{c}_{12}^{Q}),\ (a_2^{\Im}[0]\cdot\mathbf{c}_{12}^{I}),\\
&(a_1^{\Re}[1]\cdot\mathbf{c}_{01}^{Q}),\ (a_1^{\Im}[1]\cdot\mathbf{c}_{01}^{I}),\ (a_2^{\Re}[1]\cdot\mathbf{c}_{02}^{Q}),\ (a_2^{\Im}[1]\cdot\mathbf{c}_{02}^{I})\Big)\\
&\oplus \mathrm{andxor2}\Big((a_1^{\Re}[0]\cdot\mathbf{c}_{01}^{Q}),\ (a_1^{\Im}[0]\cdot\mathbf{c}_{01}^{I}),\ (a_2^{\Re}[0]\cdot\mathbf{c}_{02}^{Q}),\ (a_2^{\Im}[0]\cdot\mathbf{c}_{02}^{I})\Big)
\end{aligned}
\tag{4.78}
$$

where this leaves the XOR of the codewords on each layer, given as:

$$
\begin{aligned}
&\Big((a_1^{\Re}[2]\cdot\mathbf{c}_{01}^{Q}) \oplus (a_1^{\Im}[2]\cdot\mathbf{c}_{01}^{I}) \oplus (a_2^{\Re}[2]\cdot\mathbf{c}_{02}^{Q}) \oplus (a_2^{\Im}[2]\cdot\mathbf{c}_{02}^{I})\\
&\oplus (a_1^{\Re}[1]\cdot\mathbf{c}_{11}^{Q}) \oplus (a_1^{\Im}[1]\cdot\mathbf{c}_{11}^{I}) \oplus (a_2^{\Re}[1]\cdot\mathbf{c}_{12}^{Q}) \oplus (a_2^{\Im}[1]\cdot\mathbf{c}_{12}^{I})\\
&\oplus (a_1^{\Re}[0]\cdot\mathbf{c}_{21}^{Q}) \oplus (a_1^{\Im}[0]\cdot\mathbf{c}_{21}^{I}) \oplus (a_2^{\Re}[0]\cdot\mathbf{c}_{22}^{Q}) \oplus (a_2^{\Im}[0]\cdot\mathbf{c}_{22}^{I})\Big)
\end{aligned}
\tag{4.79}
$$

We can only now apply the XOR of the known information from the lower layers (layer 0 and layer 1) to decode for the XOR combination of codewords on layer 2:

$$\begin{aligned}&\Big((a_1^{\mathfrak{R}}[2]\cdot\mathbf{c}_{01}^{Q})\oplus(a_1^{\mathfrak{I}}[2]\cdot\mathbf{c}_{01}^{I})\oplus(a_2^{\mathfrak{R}}[2]\cdot\mathbf{c}_{02}^{Q})\oplus(a_2^{\mathfrak{I}}[2]\cdot\mathbf{c}_{02}^{I})\\&\oplus(a_1^{\mathfrak{R}}[1]\cdot\mathbf{c}_{11}^{Q})\oplus(a_1^{\mathfrak{I}}[1]\cdot\mathbf{c}_{11}^{I})\oplus(a_2^{\mathfrak{R}}[1]\cdot\mathbf{c}_{12}^{Q})\oplus(a_2^{\mathfrak{I}}[1]\cdot\mathbf{c}_{12}^{I})\\&\oplus(a_1^{\mathfrak{R}}[0]\cdot\mathbf{c}_{21}^{Q})\oplus(a_1^{\mathfrak{I}}[0]\cdot\mathbf{c}_{21}^{I})\oplus(a_2^{\mathfrak{R}}[0]\cdot\mathbf{c}_{22}^{Q})\oplus(a_2^{\mathfrak{I}}[0]\cdot\mathbf{c}_{22}^{I})\Big)\\&\oplus\Big((a_1^{\mathfrak{R}}[2]\cdot\mathbf{c}_{01}^{Q})\oplus(a_1^{\mathfrak{I}}[2]\cdot\mathbf{c}_{01}^{I})\oplus(a_2^{\mathfrak{R}}[2]\cdot\mathbf{c}_{02}^{Q})\oplus(a_2^{\mathfrak{I}}[2]\cdot\mathbf{c}_{02}^{I})\\&\oplus(a_1^{\mathfrak{R}}[1]\cdot\mathbf{c}_{11}^{Q})\oplus(a_1^{\mathfrak{I}}[1]\cdot\mathbf{c}_{11}^{I})\oplus(a_2^{\mathfrak{R}}[1]\cdot\mathbf{c}_{12}^{Q})\oplus)a_2^{\mathfrak{I}}[1]\cdot\mathbf{c}_{12}^{I})\Big)\end{aligned}\tag{4.80}$$

This can only be carried out if the previous individual complex components of the codewords have been successfully decoded. This leaves:

$$\Big((a_1^{\mathfrak{R}}[0]\cdot\mathbf{c}_{21}^{Q})\oplus(a_1^{\mathfrak{I}}[0]\cdot\mathbf{c}_{21}^{I})\oplus(a_2^{\mathfrak{R}}[0]\cdot\mathbf{c}_{22}^{Q})\oplus(a_2^{\mathfrak{I}}[0]\cdot\mathbf{c}_{22}^{I})\Big)\tag{4.81}$$

Finally, the complex components of the codewords from both sources can be decoded by applying the XOR of the known information from one source to recover the other.

At source 2:

$$\begin{aligned}&\Big((a_1^{\mathfrak{R}}[0]\cdot\mathbf{c}_{21}^{Q})\oplus(a_1^{\mathfrak{I}}[0]\cdot\mathbf{c}_{21}^{I})\oplus(a_2^{\mathfrak{R}}[0]\cdot\mathbf{c}_{22}^{Q})\oplus(a_2^{\mathfrak{I}}[0]\cdot\mathbf{c}_{22}^{I})\Big)\\&\oplus\Big((a_2^{\mathfrak{R}}[0]\cdot\mathbf{c}_{22}^{Q})\oplus(a_2^{\mathfrak{I}}[0]\cdot\mathbf{c}_{22}^{I})\Big)=(a_1^{\mathfrak{R}}[0]\cdot\mathbf{c}_{21}^{Q})\oplus(a_1^{\mathfrak{I}}[0]\cdot\mathbf{c}_{21}^{I})\end{aligned}\tag{4.82}$$

At source 1:

$$\begin{aligned}&\Big((a_1^{\mathfrak{R}}[0]\cdot\mathbf{c}_{21}^{Q})\oplus(a_1^{\mathfrak{I}}[0]\cdot\mathbf{c}_{21}^{I})\oplus(a_2^{\mathfrak{R}}[0]\cdot\mathbf{c}_{22}^{Q})\oplus(a_2^{\mathfrak{I}}[0]\cdot\mathbf{c}_{22}^{I})\Big)\\&\oplus\Big((a_1^{\mathfrak{R}}[0]\cdot\mathbf{c}_{21}^{Q})\oplus(a_1^{\mathfrak{I}}[0]\cdot\mathbf{c}_{21}^{I})\Big)=(a_2^{\mathfrak{R}}[0]\cdot\mathbf{c}_{22}^{Q})\oplus(a_2^{\mathfrak{I}}[0]\cdot\mathbf{c}_{22}^{I})\end{aligned}\tag{4.83}$$

With the outer layer being left uncoded the complex components of the outer layer are also the datawords of the complex components. For the general case once the codewords have been decoded for each source the decoder for that layer can be used to recover the datawords. Again equations (4.75), (4.76), (4.82) and (4.83) are in the same format as shown for layer 0 where we described the conditions at which successful decoding of the individual components of the codewords can take place.

4.7 Summary

Within this chapter we demonstrate successful decoding for two code models using Reed-Muller codes. Detail regarding the encoder(s) and decoder(s) of each scenario and the codes implemented on each code layer are detailed in Section 4.1 and 4.2. Successful decoding is demonstrated in the form of BER performance results, where this demonstrates versatility of the derived expressions as we expand from the two user case in Section 4.4 to three user case in Section 4.5. In Section 4.6 demonstration of decoding at the hub in the form of a two-way relay channel to highlight decoding the individual codewords and datawords is explored. We see that unsuccessful decoding on the inner layer (layer 0) has a propagation effect to the outer layers when the integer coefficients are not unit valued. The result of this becomes even more significant and increasingly complex when integer coefficients are complex and complex signals are employed. In this particular case clearly there are limitations to the practicality in this topology. The two-way relay channel model is explored as an introductory topology, further topologies that allow base station cooperation could be investigated to improve results.

Chapter 5

Lattice Decoding of C&F using Construction D and Low complexity Coefficient Selection Algorithms

5.1 Introduction

C&F has been reviewed as a promising technique for handling increasing capacity and interference [2]. However, we have stated that impracticality can arise from 1) the proposed implementation of construction A lattice and 2) coefficient selection algorithms. Whilst we have taken a construction D approach to point 1 in chapter 3 it is still necessary to investigate point 2. We extend our system model with two code scenarios to non-integer static channels and Rayleigh fading channels. We address point 2 by employing a coefficient selection algorithm from recent work of [5] where throughput results are presented. We draw comparison on each of these results for both code scenarios and also draw attention to the work of [1] and detail the differences between code choices between this work and theirs. We also show that we can decode using varying decoders within the multilayer decoder and have employed the derived expressions to account for the interaction terms at each layer.

Coefficient selection is a technique employed to evaluate the most appropriate integer coefficient vector in order to estimate the equation of messages $\hat{\mathbf{u}}_b$. This means that the complexity becomes dependent on encoding and decoding techniques. There are two main approaches to coefficient selection, local selection and global selection:

- local selection: Each base station selects the integer coefficient vector to maximise its local computation rate $R_b(h_b, a_b)$.
- global selection: Each base station chooses the integer coefficient vector such that they form a matrix whose rank is equal to the number of users. This is in order to recover the data of all the individual users.

For global selection the most common method is to transmit multiple linear equations to the hub. The full rank matrix is then chosen to optimise globally and then passed to the base stations for decoding. This relies on the selected matrix being of full rank in order to successfully decode for all users. Other approaches of this technique require known information in order to reduce the probability of rank deficiency. The work of [20] proposes two global coefficient selection algorithms, 'Blind C&F' and 'Partially Coordinated C&F'. This work focuses on non-cooperation of base station nodes and partial-cooperation respectively where the $\mathbf{Q}$ matrix is not singular in contrast to research [21] [22] whose algorithms require all base stations to have full cooperation. The original C&F paper [2] introduces a local selection approach where each base station obtains the optimal solution by performing an exhaustive search. Other research such as [23] has introduced the topic as a shortest vector problem where many examples of such algorithms are lattice reduction based such as [24]. The work of [5] states that algorithms such as this have disadvantages as with increasing number of users performance degrades. For this reason we look at recent research such as [5] which demonstrate lower complexity. For this work we focus on a single base station and therefore investigate local selection.

The latest research for low complexity coefficient selection based on local selection is proposed in [5]. The work of [5] explores low complexity algorithms to aid decoding and introduces two low complexity algorithms for both the real and complex scenarios. The author of [5] shows that the algorithms outperform LLL and achieve the same computation rate as exhaustive search. The work of [5] explores the complexities of these algorithms and shows that the complexity is significantly reduced when compared to exhaustive search and lower in complexity to LLL except at very high SNR. The complex exhaustive-II is shown to be an optimal algorithm as such selecting the best integer coefficient vector for both $\mathbb{Z}[\mathbb{i}]$ and $\mathbb{Z}[\omega]$ lattices. For this work we only focus on the integers $\mathbb{Z}[\mathbb{i}]$.

In this chapter we focus on the Complex Exhaustive-II algorithm for $\mathbb{Z}[\mathbb{i}]$ lattices only. We consider the complex scenario only to demonstrate the system for static and Rayleigh fading channels. The derived expressions from the previous chapter are employed in order to decode successfully and therefore show decodability. The main contributions of this chapter are as follows:

- Introduce complex Exhaustive-II from [5] into the C&F system model applying the selected integer coefficient vector into the multi layer structure.

- We implement RM(0,4) and RM(2,4) codes of length 16 in our C&F system model, the inner layer RM(0,4) code and RM(2,4) code for the middle layer, outer layer remains uncoded.

- We also implement Reed-Müller codes of length 32 in our C&F system model. We employ RM(1,5) code for the inner layer, RM(3,5) code for the middle layer and leave the outer layer uncoded.

- As in chapter 4 we adapt and employ soft decision decoder [18] for decoding the middle layer RM(3,5) code and implement our own decoder using Fast Walsh Hadamard Transform (FWHT) for the inner layer RM(1,5) code.

- We evaluate the system by presenting throughput results for static channels and fading channels, giving further validation of our derived expressions for decoding each code layer. We draw comparison of the throughput performance of the two schemes and also discuss the differences between codes in [1] with Figure 1.a and 1.b in the same work.

As with the previous chapters we choose the lattices to be defined over the integers $\mathbb{Z}[\mathbb{i}]$ and therefore adapt Complex-Exhaustive-II algorithm of [5] to this use only. Where this is shown in algorithm 1. We begin by briefly describing the operation of Exhaustive-II for the complex scenario only, as this is the basis as to which we demonstrate our results.

5.2 Complex-Exhaustive-II algorithm and adaptation

The aim of the complex exhaustive-II algorithm is to select the optimum value of α that moves the channel coefficients as close as possible to the nearest integers, hence selecting the optimum integer coefficient vector. The search area over α is determined by the Voronoi regions of the lattice(s) $\mathcal{V}_j$, given by each user, and the upper bound $\alpha < \sqrt{SNR}$ as defined in [5]. The Voronoi regions of this area are defined by the vertices of the lattices and their crossing points. The author of [5] establishes two different subsets for calculating the vertices and crossing points of the edges, given as $\mathcal{S}-\mathrm{I}$ and $\mathcal{S}-\mathrm{II}$. $\mathcal{S}-\mathrm{I}$ assumes that when calculating the crossing points in $\mathcal{V}_{j,a_j}$, which we denote as cp_1 and cp_2, the edges belong to the same set Ψ_j. This is outlined in lines 5-11 in Algorithm 1. $\mathcal{S}-\mathrm{II}$ outlines that the calculation is between an intersection of two sets of parallel lines. This is the case where cp_1 and cp_2 belong to different Ψ_j. Both subsets $\mathcal{S}-\mathrm{I}$ and $\mathcal{S}-\mathrm{II}$ are adopted in this algorithm in order to evaluate the complete $\mathcal{V}_{\mathbf{a}}$ for all $\mathbf{a}$. Once the vertices and crossing points are calculated the edges of $\mathcal{V}_{\mathbf{a}}$ are established. The work of [5] employs a 'full-direction' quantiser such that all regions are visited by considering all possible α. It is not satisfactory to only quantise in a certain direction(s) for all vertices as the number of vertices of each region is unknown, therefore taking this approach could miss potential $\mathcal{V}_{\mathbf{a}}$. In order to prevent this, the 'full-direction' quantiser calculates in all directions for all vertices. In this particular scenario we employ two users for $\mathbb{Z}[\mathrm{i}]$ lattices. For each user there are multiple numbers of vertices, as we consider the crossing points of the edges as well as the vertices of the Voronoi region of each lattice, all of which the quantiser would return 4 equally likely $\mathbf{a}$. The optimum $\mathbf{a}$ is then evaluated by searching over α and calculating the computation rate, given $\mathbf{h}_j$, $\mathbf{a}$ and snr where $\mathrm{snr} = P/\sigma^2$ and is defined as signal to noise ratio. α_{mmse} is given by:

$$\alpha_{mmse}(\mathbf{a}_j) = \frac{\mathrm{snr}\mathbf{h}_j^H\mathbf{a}_j}{1+\mathrm{snr}\|\mathbf{h}_j\|^2} \tag{5.1}$$

and computation rate is shown in Algorithm 1 - Phase 2, line 5.

```
Algorithm 1 Algorithm 1 - Complex Exhaustive-II
Input: channel vector $\mathbf{h} = [h_1, h_2, \ldots, h_J] \in \mathbb{C}^j$, SNR, integer domain $\mathbb{A}[\mathbb{Z}[\mathbb{i}]]$ with Basis $\mathbb{Z}[\mathbb{i}]$
Output optimal coefficient vector $\mathbf{a}_{\text{opt}}$

Phase 1: Obtain the representatives of $\alpha$, stored in set $\mathcal{S}$. The initial set $\mathcal{S} = \emptyset$
According to proposition 1 in [5], calculate the range of $\alpha : \tilde{\Phi}$
for $j = 1 : J$ do
    Generate $\Lambda_j$ according to $\Lambda_j \triangleq \{\lambda : \lambda = \frac{a_j}{h_j} \in \mathbb{A}\}$. Find $\{\lambda : \lambda \in \Lambda_j \cap \tilde{\Phi}\}$, and store these $\lambda$ into $\Omega_j = \{\alpha^*_{j,1}, \alpha^*_{j,2}, \ldots, \alpha^*_{j,v_j}\}$
$\mathcal{S}_j = \emptyset, \Psi_j = \emptyset$
    for $v = 1 : V_j$ do
        $a_l = \alpha^*_{j,v_j} h_j$
        Find the vertices of the corresponding $\mathcal{V}_{j,a_j}$, calculated by $\alpha^*_{j,v} + \frac{1}{h_j}\frac{z_0}{2}$, where $z_0 = \pm 1 \pm \mathbb{i}$
        Store these vertices in set $\mathcal{S}_j$
        Calculate the linear equation of the edge of $\mathcal{V}_{j,a_j}$, add them into set $\Psi_j$
    end for
    $\mathcal{S} = \mathcal{S} \cup \mathcal{S}_{\bar{j}}$
end for
for $\bar{j} = 1 : J - 1$ do
    for $\hat{j} = \bar{j} + 1 : J$ do
        Find all combinations of $\{\mathbf{cp}_1, \mathbf{cp}_2\}$, with $\text{cp}_1 \in \Psi_{\bar{n}}$ and $\text{cp}_2 \in \Psi_{\hat{j}}$. Calculate the crossing point of $\text{cp}_1$ and $\text{cp}_2$: the crossing points which are not in $\tilde{\Phi}$ should be discarded. Store the remaining set in $\mathcal{S}_{\bar{j},\hat{j}}$
        $\mathcal{S}_{\bar{j}} = \mathcal{S}_{\bar{j}} \cup \mathcal{S}_{\bar{j},\hat{j}}$
    end for
    $\mathcal{S} = \mathcal{S} \cup \mathcal{S}_{\bar{j}}$
end for

Phase 2: Obtain the candidate set $\mathcal{I}$ and select $\mathbf{a}_{\text{opt}}$
for all representative $\alpha \in \mathcal{S}$ do
    $\mathbf{a} = \mathcal{Q}^*_{\mathbb{A}}(\alpha\mathbf{h})$, discard the repeated $\mathbf{a}$, take the remaining $\mathbf{a}$ as the candidate set $\mathcal{I}$
end for
for all $\mathbf{a} \in \mathcal{I}$ do
    Calculate $\mathcal{R}(\mathbf{h}, \mathbf{a}) = \text{maxlog}^+\left(\frac{P}{\alpha_j^2\sigma^2 + P\|\alpha_j\mathbf{h}_j - \mathbf{a}_j\|^2}\right)$
end for
Return $\mathbf{a}_{\text{opt}} = \text{argmax}_{\mathbf{a}\in\mathcal{I}}\mathcal{R}(\mathbf{h}, \mathbf{a})$
```

Algorithm 1 has been adapted from [5] for our particular scenario, using $\mathbb{Z}[\mathfrak{i}]$, and as an Exhaustive search algorithm it completes a search over a wide range of values, limited only by an upper bound described in [5] as $\alpha < \sqrt{SNR}$, to determine the best integer coefficient vector. In order to calculate the computation rate and select the maximum rate for the best **a** vector we detail the calculation for average power P. As we implement unipolar codes and employ CCF using equation (2.8) for three layers, the code values are in the positive reals therefore the Average power is evaluated as:

$$P = 17.5 = \frac{0^2 + 1^2 + 2^2 + 3^2 + 4^2 + 5^2 + 6^2 + 7^2}{8} \tag{5.2}$$

Usually codes wouldn't reach a real value of 7 as bipolar codes would be cancel each other out however due to our use of unipolar codes our work could be easily adapted by applying an offset to the codes.

5.3 Throughput performance results

Within this section the throughput performance of each code scenario is evaluated. The throughput is calculated based on the frame error rate as the frame error rate alone is not suitable for codes that have varying effective rates. The throughput is defined as the complement of the frame error rate (FER) multiplied by the codes message rate R_{mes}. $R_{\text{mes}} = (2/N) * \sum k_l$ and Throughput$(TP) = R_{\text{mes}}(1 - FER)$.

5.3.1 Static channel results

Throughput results of RM(0,4), RM(2,4) and uncoded codes using static channels We begin this subsection with results for the static channels $h_1 = 1.17 + 2.15\mathfrak{i}$ and $h_2 = 1.25 - 1.63\mathfrak{i}$ using repetition code on the inner layer, extended Hamming code on the middle layer and the outer layer remaining uncoded. As in previous chapters we use codes of length $N = 16$ complex symbols per In-phase and Quadrature component.

The performance of the system employing this choice of codes is shown in Figure 5.1. Following the blue triangles in Figure 5.1 we see that at 16dB the throughput begins to increase quickly compared with the lower SNR, and by 23dB the maximum throughput has been reached. It can also be observed that there is an initial small increase in throughput from 5dB to 14dB where the codes are not able to overcome the noise and interference.

The throughput curve shows that for our system model using multilayer decoding and the derived expressions at each layer successful decoding can be achieved. In order to draw comparison to the work of [1], results shown in green triangles in Figure 5.1, an explanation is given on their work and the code model implemented. The work of [1] implements multilayer decoding using lifted construction D. The author demonstrates the throughput of the chosen codes for static channels where they employ the same channel coefficients $h_1 = 1.17 + 2.15\mathrm{i}$ and $h_2 = 1.25 - 1.63\mathrm{i}$. Their work aims to demonstrate the performance difference between choices of codes at each layer. They employ a one, two and three layer scheme and consistently employ a $\frac{1}{2}$ rate convolutional code on the inner layer. The comparison between the three scenarios shows how the choice of codes can impact performance. For the first scenario they employ:

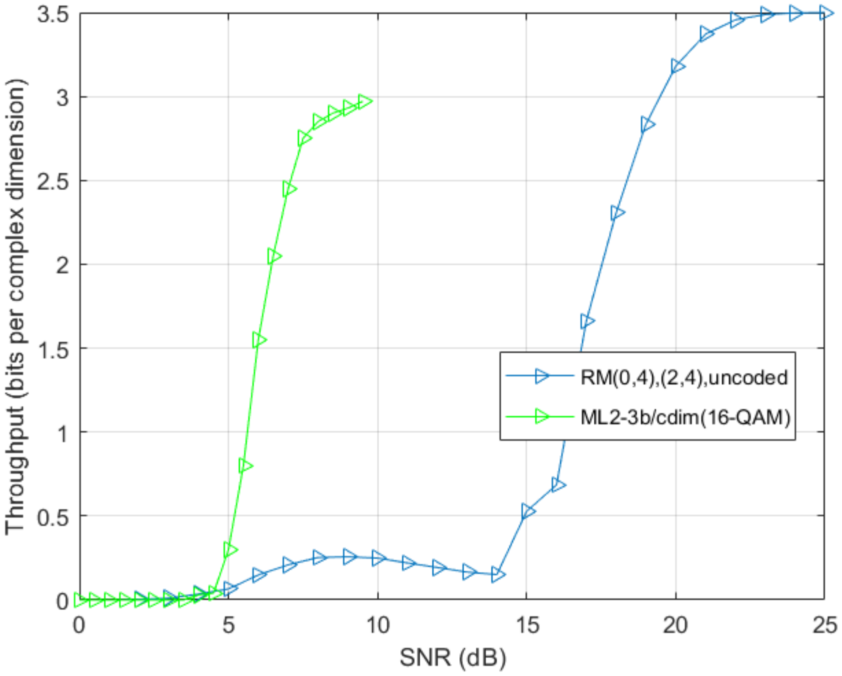


Figure 5.1: Throughput performance of a two user, three layer system model using RM(0,4) RM(2,4) codes, results from [1] for a two user, two layer system model

- A single code - $\frac{1}{2}$ rate convolutional code of length 200 complex symbols per In-phase and Quadrature.
- A two layer scheme using the same $\frac{1}{2}$ rate convolutional code from the first scenario and the second layer is uncoded. This leaves the effective rate of 2.94b/cdim. The

codes are also 200 complex symbols.

- A three layer scheme that again uses the convolutional code of the first layer, the remaining layers, 1 and 2, are uncoded and therefore the effective rate of this scheme is 4.94b/cdim. The codes are again 200 complex symbols.

We compare our throughput results with the two layer scheme as they have similar effective rates to this scheme of 3.5b/cdim. The work of [1] shows that the maximum rate is achieved at a much faster rate even though it is lower than our proposed scheme, and at lower SNR. This difference is affected by the implementation of two layers oppose to three and the use of much longer codes.

Throughput results for RM(1,5), RM(3,5) and uncoded codes using static channels

The codes employed are the (32,6,16) Hadamard code for the inner layer, (32,26,4) extended Hamming code on the middle layer and the outer layer remaining uncoded. To calculate the throughput we now evaluate this using the new code length $N = 32$. By observing Figure 5.2 we can see that the introduction of Reed-Muller RM(1,3) and RM(3,5) codes into our system model can demonstrate performance improvement oppose to the implementation of the repetition code and Extended Hamming code shown in Figure 5.1. The maximum throughput reached for this system is 4b/cdim. Figure 5.2 has a sharper increase in throughput rate at 16dB compared with Figure 5.1, which has a slower incline to maximum throughput. The maximum throughput when implementing this choice of codes is 4 which from Figure 5.2 this is shown to be achieved at 23dB.

We now observe and compare the performance of the three layer scheme of [1] as that is the most relevant to our work. As stated this scheme employs a $\frac{1}{2}$ rate convolutional code on the inner layer and leaves the two remaining layers uncoded. This means that from the middle to the outer layer the construction D constraint is not fulfilled and from the inner to the middle layer it is where the minimum Hamming distance of 4 is always achieved in this case. Effectively this implementation is just CCF without adhering to the rules to generate a Construction D lattice and therefore not Construction D.

The result of [1], shown as the dark blue triangles in Figure 5.2, shows that the maximum throughput using this scheme is 5b/cdim which is higher than what is achieved in

Figure 5.1 and Figure 5.2. This is due to the increased length of the codewords employed, where we employ length 16 and 32 complex symbols respectively, compared to 200 complex symbols and the increased difference in Hamming distance that can be achieved when employing uncoded codes for both outer layers. We also see that the dark blue triangle line crosses our results shown in lighter blue triangles at approximately 25dB where our system model had achieved maximum throughput at approximately 23dB. On the one hand the Construction D constraints could improve BER performance however because [1] have employed much longer codes their performance is better.

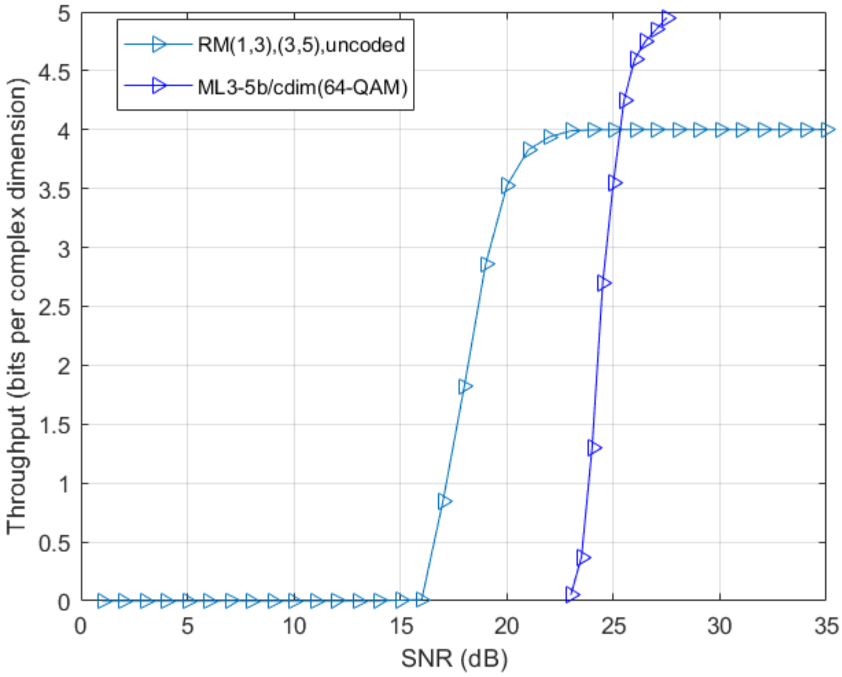


Figure 5.2: Throughput performance of a two user, three layer system model using RM(1,5), RM(3,5) codes, results from [1] using a two user, three layer system model

5.3.2 Fading channel results

For the fading channels we employ the same system model with two users and three layers. We show the results for both scenarios as we have done in subsection 5.3.1.

0 and 1st order Reed-Muller codes

Again, we employ the (16,1,16) repetition code for the inner layer and (16,11,4) extended Hamming code for the middle layer, leaving the outer layer uncoded. By employing the coefficient selection algorithm of [5] we can predict from Figure 5.3 that the system can approach the maximum throughput of 3.5 bits per complex dimension at approximately 35dB. This is predicted due to the simulation needing longer to complete. The performance of Rayleigh fading channels is naturally going to be degraded compared to static channels however the coefficient selection algorithm has been able to choose suitable integer coefficients and α values such that the C&F scheme is able to successfully decode at higher SNR. Gradually as the SNR increases and the channel is not completely overcome with noise the throughput increases showing that at 29dB the system can achieve throughput of 3.32 bits per complex dimension. Figure 5.3 also shows that with the use of the derived mathematical expressions for the expected decoded output on each layer, successful decoding in the multilayer decoder is achieved with Rayleigh fading channels in our system model.

Due to our use of a repetition code and extended Hamming code on the inner and middle layers respectively and leaving the outer layer uncoded the curve for our scenario carries the same shape as shown in Figure 1(b) of [1] for the three layer system model however it does not achieve the same maximum throughput at a particular SNR. This is because with the use of the repetition code and Extended Hamming code the minimum euclidean distance is smaller than the distance between the layers used in [1]. The maximum throughput of 5b/cdim in [1] can't be achieved in this scenario however we show that good performance and successful decoding can be achieved within the C&F system based on construction D.

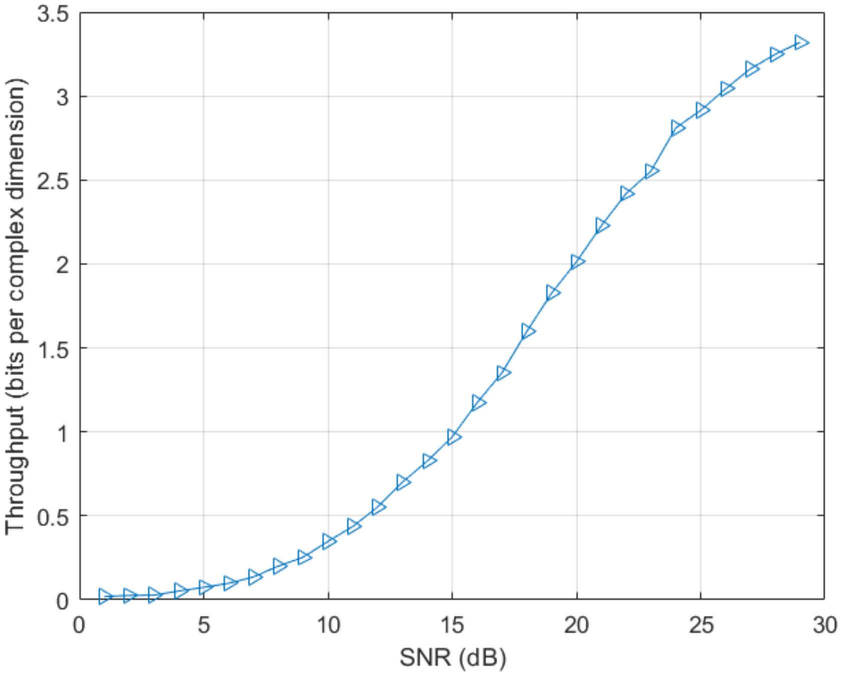


Figure 5.3: Throughput performance of two user, three layer system with channels subject to Rayleigh fading. Layers are $l = 0, 1, 2$ RM(0,4), RM(2,4) and uncoded respectively

1st and 3rd order Reed-Muller codes

We continue with our second code scenario and employ the (32,6,16) Hadamard code for the inner layer and the (32,26,4) extended Hamming code for the middle layer, leaving the outer layer uncoded. The maximum throughput of this code scenario is 4 bits per complex dimension. We can observe Figure 5.4 to see that at 30dB the throughput is at 3.7 bits per complex dimension. The slope does not reach maximum throughput as the simulation has not run for long enough. However, we predict that the maximum throughput of 4 bits per complex dimension will be reached by between 35dB-40dB. The slope of the curve is fairly shallow when compared to the static channel case shown in Figure 5.2. The nature of the fading channels and the coefficient selection algorithm selecting a new α and integer coefficient vector at every trial run through the simulation the self-interference and capacity to overcome it becomes much more difficult.

If we draw comparison to the work of [1] in Figure 1b we can see that the maximum throughput of the three layer 64-QAM system is 5b/cdim which is 1b/cdim greater than

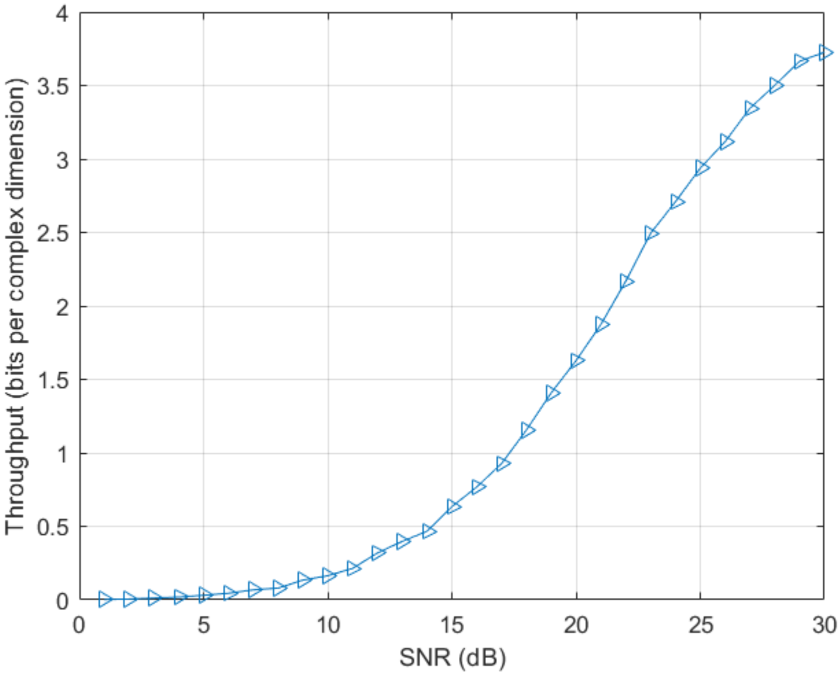


Figure 5.4: Throughput performance of two user, three layer system with channels subject to Rayleigh fading. Layers are $l = 0, 1, 2$ RM(1,5), RM(3,5) and uncoded respectively

our system model. With our prediction of maximum throughput of this system model being more closely approached at approximately 35dB-40dB we can say that the performance will be shifted to the right by some dB. Figure 5.4 demonstrates the successful implementation of multilayer decoding under construction D constraints whilst introducing more complex codes than that shown in [1] using Rayleigh fading channels.

5.4 Summary

In this chapter successful decoding within our C&F system model has been explored and the work from chapter 3, specifically two user case, extended to employ static non-integer channel coefficients and Rayleigh fading channels. Through investigation of low complexity coefficient selection algorithms the implementation of the Complex-exhaustive-II algorithm has been implemented into our system model in order to determine the best α and integer coefficients at various SNR. As with previous chapters this work considers the Gaussian integer lattice $\mathbb{Z}[\mathbb{i}]$.

Chapter 6

Conclusion and future work

6.1 Summary

In this thesis we have investigated the use of construction D in a C&F system model in order to move away from the originally introduced construction A and introduce the use of binary codes. We first demonstrated the discovery of an error floor which highlighted that an interaction between code layers was happening within the multilayer decoder. This initial analysis was based on the BCJR decoder and convolutional codes. Even though the BER performance of this two layer model was not the focus we notice that the BER performance could be improved and therefore following this evaluation better performing codes were investigated and implemented further on in this work. In order to implement construction D we employed a multilayer decoder and demonstrated the occurrence of an interaction between code layers, producing derived expressions to describe the signals mathematically. We evaluated the BER performance of the system for two instances of Reed-Muller codes, namely 2nd order extended Hamming code, Repetition code and 3rd order extended Hamming code, highlighting that with the expressions we are able to successfully decode. We demonstrate the operation of this system model for unit valued, complex integer and fading channels.

The aims introduced at the beginning of this thesis highlighted the approach to this work. Here, we describe how these aims have been met:

1) Choose practical codes, such as binary codes in order to derive a practical basis model for C&F

For the unit valued channel, integer, complex integer and fading channel scenario we have presented two different forms and orders of Reed-Muller codes that satisfy the construction D constraints.

2) Introduce existing encoding and in particular decoding algorithms that are employed within industrial applications leading to an implementable C&F
Initially we employed the $\frac{1}{2}$ rate convolutional encoder and BCJR algorithm for decoding a two layer model where this model demonstrated the error floor.
We then employed differing types of encoder(s) and decoder(s) for two scenarios, the first scenario: for the inner layer we use a (16,1,16) repetition code and the middle layer a (16,11,4) extended Hamming code, leaving the outer layer uncoded. We detail the adaptations of [18] regarding the decoder in chapter 4. The second scenario: the inner layer we employ the (32,6,16) Hadamard code, the middle layer we employ the (32,26,4) extended Hamming code and leave the outer layer uncoded. We detail the adaptation for this scenario of the work of [18]. All encoders and decoders are existing algorithms which have been well established.

3) Solve the problem of using lattice construction A by replacing this with a low complexity method that generates lattice construction D.
We investigated the implementation of Construction D methods to generate the lattice. The work of [4] demonstrated that there are differences in the application of methods in order to construct the correct lattice. Through investigating this research we employed Construction by Code Formula (CCF) for its properties that enable the use of conventional encoders and decoders.

4) Extend the practical C&F system model to handle complex signals and complex valued channels.
We extend the C&F system model such that it can manage complex signals and complex integer valued channels where the main challenge for this part of the work was determining the effect of the interaction signal between decoding layers. We present derived expressions through lemmas and proofs to evaluate the nature of the channels on the expected decoded output.

5) Investigate multilayer lattice decoding algorithms and the effect of this on decoding

nested lattice codes.
During the investigation of multilayer lattice decoding we uncovered the error floor and subsequent interaction between layers in the multilayer decoder. With the extension to complex signals we investigated the progression from construction D to Lifted construction D which applies the construction to the complex plane. This could have altered the operation of the decoder however during our research we discover that the complex components can be treated as separate and identical encoders and decoders. We implemented this into our system model whilst incorporating the knowledge of the interaction terms and demonstrated successful decoding.

6) Present low complexity coefficient selection algorithms within the C&F model for decoding random channels.
We incorporate the work of [5] complex exhaustive-II algorithm such that we extend our C&F system model to fading channels. We demonstrate the successful decoding of both scenarios of Reed-Muller codes within our system model by generating throughput results and also show that with higher order codes we can achieve higher performance.

7) Analyse the suited topology of the system and recovery of data at CPU.
We analysed the recovery of data at the hub or CPU through the two-way relay channel. We describe the process of recovery for both the real and complex signal case as well as unit, integer and complex integer channels and express the stages of recovery for each layer using logical expressions such as AND and XOR.

6.2 Future Work

Future work extending this research is given as follows:

- The investigation of lattice constructions focuses on those that can produce binary lattice codes for multilayer codes. The investigation into the use of other codes and their complexities for decoding may be considered.

- The system model within this work employed a single base station. Compute-and-Forward originally introduced in [2] describes multiple base stations. An area of exploration could be to extend this work to multiple base stations and their cooperation. A topic such as massive MIMO could be considered.

- Studies such as [25] have shown that Eisenstein integers $\mathbb{Z}[\omega]$ are employed within PNC due to their good coding qualities and packing structures. The best packing lattice structure is the hexagon. The low complexity coefficient selection algorithms of [5] do indeed extend to $\mathbb{Z}[\omega]$ and therefore the performance differences may be analyzed for $\mathbb{Z}[\omega]$ and $\mathbb{Z}[\mathbb{i}]$.

- Reed-Muller codes are chosen in order to adhere to the construction D constraints of the multilayer construction. Implications on the codes stem from the minimum distance between code layers having a difference of multiple of 4. In the work of [1] these constraints have been avoided due the middle and outer layers being left uncoded. This could be considered such that the flexibility on the choice of codes could be increased and in turn performance of the system. Evaluation of the performance for both adhering and not adhering to constraints could be analyzed.

- The lowest complexity algorithm of [5] namely Linear Search, will be implemented and the throughput performance difference analysed against the implementation of Complex Exhaustive-II.

- Complexity analysis of construction D and employed decoding strategies will be studied.

Bibliography

[1] P. R. B. da Silva and D. Silva, "Design of lattice network codes based on construction 'd'," *2014 International Telecommunications Symposium (ITS)*, 2014.

[2] B. Nazer and M. Gastpar, "Compute-and-forward: Harnessing interference through structured codes," *IEEE Transactions on Information Theory*, vol. 57, pp. 6463–6486, Oct. 2011.

[3] Y. Wang and A. Burr, "Soft detection of multilevel lattice network coding," *2015 European Conference on Networks and Communications (EuCNC)*, pp. 62–66, 2015.

[4] M. M. Molu, K. Cumanan, M. Bashar, and A. Burr, "On convolutional lattice codes and lattice decoding using trellis structure," *IEEE Access*, vol. 4, pp. 9702–9715, Jan. 2016.

[5] Q. Huang and A. Burr, "Low complexity coefficient selection algorithms for compute-and-forward," *IEEE Access*, vol. 5, pp. 19182–19193, 2017.

[6] S. Zhang, S. C. Liew, and P. Lam, "Hot topic: Physical-layer network coding," *Proceedings of the Annual International Conference on Mobile Computing and Networking, MOBICOM*, vol. 2006, pp. 358–365, 01 2006.

[7] R. Ahlswede, Ning Cai, S. . R. Li, and R. W. Yeung, "Network information flow," *IEEE Transactions on Information Theory*, vol. 46, no. 4, pp. 1204–1216, 2000.

[8] A. E. Gamal and Y.-H. Kim, *Network information theory.* Cambridge University Press, 2018.

[9] R. Zamir, *Lattice coding for signals and networks: A structured coding approach to quantization, modulation, and multi-user information theory.* Cambridge University Press, 2014.

[10] Bugler, *Guide to Abstract Algebra.* MacMillan, 1988.

[11] D. G. Northcott, *Lessons on rings, modules and multiplicities.* Cambridge Univ. Press, 1968.

[12] J. H. Conway and N. J. A. Sloane, *Sphere Packings, Lattices and Groups.* New York, NY, USA: Springer-Verlag, 3rd ed., 1999.

[13] Y. Huang and K. Narayanan, “Multistage compute-and-forward with multilevel lattice codes based on product constructions,” *2014 IEEE International Symposium on Information Theory*, 2014.

[14] Y. Wang and A. Burr, “Soft detection of multilevel lattice network coding,” *2015 European Conference on Networks and Communications (EuCNC)*, pp. 62–66, 2015.

[15] A. Viterbi, “Error bounds for convolutional codes and an asymptotically optimum decoding algorithm,” *IEEE Transactions on Information Theory*, vol. 13, no. 2, pp. 260–269, 1967.

[16] L. Bahl, J. Cocke, F. Jelinek, and J. Raviv, “Optimal decoding of linear codes for minimizing symbol error rate (corresp.),” *IEEE Transactions on Information Theory*, vol. 20, no. 2, pp. 284–287, 1974.

[17] G. Forney, “Coset codes. ii. binary lattices and related codes,” *IEEE Transactions on Information Theory*, vol. 34, no. 5, pp. 1152–1187, 1988.

[18] B. Muller, M. Holters, and U. Zolzer, “Low complexity soft-input soft-output hamming decoder,” *50th FITCE Congress - "ICT: Bridging an Ever Shifting Digital Divide"*, pp. 1–5, Aug. 2011.

[19] Khalil, “Bcjr decoder.” BCJR Decoder - File Exchange - MATLAB Central, May 2016.

[20] M. M. Molu, K. Cumanan, and A. Burr, “Low-complexity compute-and-forward techniques for multisource multirelay networks,” *IEEE Communications Letters*, vol. 20, no. 5, pp. 926–929, 2016.

[21] L. Wei and W. Chen, “Compute-and-forward network coding design over multi-source multi-relay channels,” *IEEE Transactions on Wireless Communications*, vol. 11, no. 9, pp. 3348–3357, 2012.

[22] Z. Chen, P. Fan, and K. B. Letaief, “Compute-and-forward: Optimization over multisource–multirelay networks,” *IEEE Transactions on Vehicular Technology*, vol. 64, no. 5, p. 1806–1818, 2015.

[23] D. S. C. Feng and F. R. Kschischang, “An algebraic approach to physical-layer network coding,” *Proc. ISIT 2010*, pp. 1017–1021, 2010.

[24] A. K. Lenstra, H. W. Lenstra, and L. Lovász, “Factoring polynomials with rational coefficients,” *Mathematische Annalen*, vol. 261, no. 4, p. 515–534, 1982.

[25] N. E. Tunali, Y.-C. Huang, J. J. Boutros, and K. R. Narayanan, “Lattices over eisenstein integers for compute-and-forward,” *IEEE Transactions on Information Theory*, vol. 61, no. 10, pp. 5306–5321, 2015.

[26] G. Kramer, M. Gastpar, and P. Gupta, “Cooperative strategies and capacity theorems for relay networks,” *IEEE Transactions on Information Theory*, vol. 51, no. 9, pp. 3037–3063, 2005.

[27] A. Burr and D. Fang, "Linear physical layer network coding based on rings," *2014 IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 370–375, 2014.

[28] G. Forney, M. Trott, and S.-Y. Chung, "Sphere-bound-achieving coset codes and multilevel coset codes," *IEEE Transactions on Information Theory*, vol. 46, no. 3, pp. 820–850, 2000.

[29] O. Ordentlich, J. Zhan, U. Erez, M. Gastpar, and B. Nazer, "Practical code design for compute-and-forward," *2011 IEEE International Symposium on Information Theory Proceedings*, pp. 1876–1880, 2011.

[30] P. Lusina, E. Gabidulin, and M. Bossert, "Efficient decoding of space-time hadamard codes using the hadamard transform," *Proceedings. 2001 IEEE International Symposium on Information Theory (IEEE Cat. No.01CH37252)*, pp. 243–, 2001.

[31] S. Jan and A. Burr, *Wireless physical layer network coding.* Cambridge University Press., 2018.

[32] U. Wachsmann, R. Fischer, and J. Huber, "Multilevel codes: theoretical concepts and practical design rules," *IEEE Transactions on Information Theory*, vol. 45, no. 5, pp. 1361–1391, 1999.

[33] S. Sahraei and M. Gastpar, "Compute-and-forward: Finding the best equation," *2014 52nd Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pp. 227–233, 2014.

[34] U. Erez and R. Zamir, "Achieving 1/2 log (1+snr) on the awgn channel with lattice encoding and decoding," *IEEE Transactions on Information Theory*, vol. 50, no. 10, pp. 2293–2314, 2004.

[35] Q. T. Sun, T. Huang, and J. Yuan, "On lattice-partition-based physical-layer network coding over gf(4)," *IEEE Communications Letters*, vol. 17, no. 10, pp. 1988–1991, 2013.

[36] A. Burr, D. Fang, and M. Molu, "Linear wireless physical-layer network coding based on binary matrices for multilayer relay networks," *2014 11th International Symposium on Wireless Communications Systems (ISWCS)*, pp. 982–986, 2014.

[37] M. A. Vázquez-Castro and F. Oggier, "Lattice network coding over euclidean domains," *2014 22nd European Signal Processing Conference (EUSIPCO)*, pp. 1148–1152, 2014.

[38] Q. T. Sun, J. Yuan, T. Huang, and K. W. Shum, "Lattice network codes based on eisenstein integers," *IEEE Transactions on Communications*, vol. 61, no. 7, pp. 2713–2725, 2013.

[39] J. Zhan, B. Nazer, U. Erez, and M. Gastpar, "Integer-forcing linear receivers," *IEEE Transactions on Information Theory*, vol. 60, no. 12, pp. 7661–7685, 2014.

[40] B. Nazer and M. Gastpar, "Reliable physical layer network coding," *Proceedings of the IEEE*, vol. 99, no. 3, pp. 438–460, 2011.

[41] Y. Zhao, R. Adve, and T. J. Lim, "Symbol error rate of selection amplify-and-forward relay systems," *IEEE Communications Letters*, vol. 10, no. 11, pp. 757–759, 2006.

[42] T. Koike-Akino, P. Popovski, and V. Tarokh, "Optimized constellations for two-way wireless relaying with physical network coding," *IEEE Journal on Selected Areas in Communications*, vol. 27, no. 5, pp. 773–787, 2009.

[43] S. Sahraei and M. Gastpar, "Compute-and-forward: Finding the best equation," *2014 52nd Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pp. 227–233, 2014.

[44] D. S. C. Feng and F. R. Kschischang, "Lattice network coding over finite rings," *2011 12th Canadian Workshop on Information Theory*, 2011.

[45] A. B. H. Fredj and J. Belfiore, "Binary subchannel decomposition of lattice-based multiple access schemes," *2017 IEEE 13th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, pp. 1–7, 2017.

[46] S. H. Lim, K. Young-Han, E. G. Abbas, and S.-Y. Chung, "Noisy network coding," *IEEE Transactions on Information Theory*, vol. 57, no. 5, pp. 3132–3152, 2011.

[47] 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, 1977.

[48] A. S. Avestimehr, S. N. Diggavi, and D. N. C. Tse, "Wireless network information flow: A deterministic approach," *IEEE Transactions on Information Theory*, vol. 57, no. 4, pp. 1872–1905, 2011.

[49] R. Zamir, "Lattices are everywhere," in *2009 Information Theory and Applications Workshop*, pp. 392–421, 2009.

[50] Y.-C. Huang and K. R. Narayanan, "Construction π_A and π_D lattices: Construction, goodness, and decoding algorithms," *IEEE Transactions on Information Theory*, vol. 63, no. 9, pp. 5718–5733, 2017.

[51] O. Shalvi, N. Sommer, and M. Feder, "Signal codes: Convolutional lattice codes," *IEEE Transactions on Information Theory*, vol. 57, no. 8, pp. 5203–5226, 2011.

[52] M. M. Molu and A. Burr, "Constructing convolutional lattices and its application in compute and forward," in *2015 IEEE International Conference on Communication Workshop (ICCW)*, pp. 2187–2193, 2015.

[53] J. Wolf, "Efficient maximum likelihood decoding of linear block codes using a trellis," *IEEE Transactions on Information Theory*, vol. 24, no. 1, pp. 76–80, 1978.

[54] Q. T. Sun, T. Huang, and J. Yuan, "On lattice-partition-based physical-layer network coding over gf(4)," *IEEE Communications Letters*, vol. 17, no. 10, pp. 1988–1991, 2013.

[55] R. Zamir and M. Feder, "On lattice quantization noise," in *Proceedings of IEEE Data Compression Conference (DCC'94)*, pp. 380–389, 1994.

[56] B. Hern and K. Narayanan, "Multilevel coding schemes for compute-and-forward," in *2011 IEEE International Symposium on Information Theory Proceedings*, pp. 1713–1717, 2011.

[57] T. Liu, P. Moulin, and R. Koetter, "On error exponents of modulo lattice additive noise channels," *IEEE Transactions on Information Theory*, vol. 52, no. 2, pp. 454–471, 2006.

[58] Y.-C. Huang, K. R. Narayanan, and P.-C. Wang, "Adaptive compute-and-forward with lattice codes over algebraic integers," in *2015 IEEE International Symposium on Information Theory (ISIT)*, pp. 566–570, 2015.

[59] Y.-C. Huang, K. R. Narayanan, and P.-C. Wang, "Lattices over algebraic integers with an application to compute-and-forward," *IEEE Transactions on Information Theory*, vol. 64, no. 10, pp. 6863–6877, 2018.

[60] W. Nam, S.-Y. Chung, and Y. H. Lee, "Capacity of the gaussian two-way relay channel to within $\frac{1}{2}$ bit," *IEEE Transactions on Information Theory*, vol. 56, no. 11, pp. 5488–5494, 2010.

[61] Y. Song and N. Devroye, "Lattice codes for the gaussian relay channel: Decode-and-forward and compress-and-forward," *IEEE Transactions on Information Theory*, vol. 59, no. 8, pp. 4927–4948, 2013.

[62] D. Chase, "Class of algorithms for decoding block codes with channel measurement information," *IEEE Transactions on Information Theory*, vol. 18, no. 1, p. 170–182, 1972.

[63] S. Hirst and B. Honary, "A simple soft-input/soft-output decoder for hamming codes," *Proceedings of the 8th IMA International Conference on Cryptography and Coding*, p. 38–43, 2001.

[64] B. Nazer and M. Gastpar, "Computing over multiple-access channels with connections to wireless network coding," *2006 IEEE International Symposium on Information Theory*, Jul 2006.

[65] J. Korner and K. Marton, "How to encode the modulo-two sum of binary sources (corresp.)," *IEEE Transactions on Information Theory*, vol. 25, no. 2, pp. 219–221, 1979.

[66] S.-Y. R. Li, Q. T. Sun, and Z. Shao, "Linear network coding: Theory and algorithms," *Proceedings of the IEEE*, vol. 99, no. 3, pp. 372–387, 2011.

[67] R. Zamir, "Lattices are everywhere," in *2009 Information Theory and Applications Workshop*, pp. 392–421, 2009.

[68] M. P. Wilson, K. Narayanan, H. D. Pfister, and A. Sprintson, "Joint physical layer coding and network coding for bidirectional relaying," *IEEE Transactions on Information Theory*, vol. 56, no. 11, pp. 5641–5654, 2010.

[69] W. Nam, S.-Y. Chung, and Y. H. Lee, "Nested lattice codes for gaussian relay networks with interference," *IEEE Transactions on Information Theory*, vol. 57, no. 12, pp. 7733–7745, 2011.

[70] A. Ashikhmin and S. Litsyn, "Simple map decoding of first-order reed-muller and hamming codes," *IEEE Transactions on Information Theory*, vol. 50, no. 8, pp. 1812–1818, 2004.

[71] J. Seberry, B. JWysocki, and T. AWysocki, "On some applications of hadamard matrices," *Metrika*, vol. 62, pp. 221–239, 2005.