Master's Thesis

Practical Aspects of Reduced Complexity LDPC Decoding using Forced Convergence

María García García
Muhammad Umar Farooq

Department of Electrical and Information Technology,
Faculty of Engineering, LTH, Lund University, 2016.

Practical Aspects of Reduced Complexity LDPC Decoding using Forced Convergence

María García García
Muhammad Umar Farooq

Department of Electrical and Information Technology
Lund University

Advisors: Muris Sarajlić
Ove Edfors

Examiner: Fredrik Rusek

June 28, 2016

Printed in Sweden
E-huset, Lund, 2016

Abstract

This thesis is an extension to the previous work done at Department of EIT in investigating the complexity reduction of LDPC decoding using the stochastic approximation method to improve the energy efficiency of energy-limited applications, such as in mobile phones.

The research carried out in this thesis investigates the complexity and convergence results of adaptive performance control algorithm (PCAA) that changes the forced convergence threshold in OMS LDPC FC decoding algorithm. The simulations are performed on IEEE802.11n OFDM communication system for Additive White Gaussian Noise (AWGN), indoor Rayleigh time varying frequency flat and Rayleigh frequency selective fading channels. Performance results are obtained by running various combinations of LDPC decoding parameters and results are documented for parameter profiles that meet the specified BLER target criteria.

In order to visualize the results, simulations are performed by implementing OFDM, PCAA and relevant channel models in MATLAB© and LDPC forced convergence decoder in C. By doing that it was possible to shorten the simulation time as LDPC decoder takes longer in MATLAB© compared to C. Performance metrics such BLER against E_b/N_0, complexity and convergence for various modulation and coding (MCSs) schemes are produced via this setup.

The aim of the thesis is accomplished and results obtained indicate significant convergence time reduction compared to previous methodology [1]. It is also concluded that optimum complexity and convergence trade-off is obtained when $\Delta\theta_d$ is set to Block Error Rate target ($BLER_t$) in AWGN channel set-up. By applying this optimum configuration to both time varying frequency flat fading and frequency selective fading channel encouraging results are observed. PCAA tracks the channel changes in both fading cases while meeting specified $BLER_t$ and with significant complexity savings relative to non-forced convergence LDPC decoding.

Acknowledgements

First we would like to express our gratitude to our supervisors, Muris Sarajlić and Ove Edfors for giving us the opportunity of working on this project. We wouldn't have been able to get through the project in time without their support.

Especially we are highly obliged to Muris Sarajlić for being our main contact throughout the thesis, for continuously guiding and supporting us on all the major aspects of the research topic and giving us the valuable advices when we were stuck at critical stages of thesis. It was truly a remarkable experience doing the thesis under his supervision and learning a great deal of knowledge in the research topic from him and we aspire to become as authentic and motivated as he is on the research.

We would like to express our sincere thank and appreciation to our teachers at EIT for sharing their knowledge and help, and for being such a great inspiration and examples for us.

Last but not least, we would like to thank our families and friends for sharing their love and support. We are extremely grateful to our parents and siblings for being there for us in much needed times of our lives because without them it wouldn't have been possible.

Preface

This thesis work is carried out by both Umar and María in collaboration with the Department of Electrical and Information technology (EIT) Lund University. Both authors pursued the same goal that was to implement the IEEE802.11n LDPC coded OFDM communication system to optimize the algorithm performance in improving the energy efficiency of energy limited applications. Both of the authors have taken active part in all the key steps of the project and its hard to separate the individual work in those. During the initial phase of the project main responsibilities of María were to implement 802.11n OFDM communication system on Matlab and Umar implemented the LDPC forced convergence decoder in C. Simulation stages for AWGN, frequency flat and frequency selective fading channel of the projects were done by both María and Umar. Umar wrote chapter 2 and 3 and María wrote chapter 4 and 5, the rest of the report was written together.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Wireless communication technologies are getting more complex due to the requirements of delivering a high data rate while maintaining reliable communication. Reliable communication across a wireless medium requires proper channel encoding and decoding. That is, to add enough redundancy to the information while keeping the information rate as high as possible. A common approach is to select combinations of modulation schemes and coding rate adaptively according to the channel conditions and the received Signal to Noise Ratio (SNR). Poor channel conditions result in lower coding rate (high number of redundant bits) thus, resulting in higher complexity at the receiver side. Higher complexity computation comes at the cost of more power consumption, which is a limited resource especially at Uplink (UL).

An attempt to reduce the complexity of such systems is discussed in [1]. In that paper, an IEEE802.11n Orthogonal Frequency-Division Multiplexing (OFDM) communication framework with Low-density Parity-check (LDPC) as Forward Error Correction (FEC) was employed to study the impact of the receiver complexity reduction on the system performance. A Forced Convergence (FC) algorithm is used to reduce the complexity of LDPC decoder. LDPC decoder works in an iterative manner on a received block, and it provides improved estimates of the codeword bits in a block. As the iterations increases, the improvement will also increase, which will increase the complexity as well. The idea of the FC algorithm is to stop the improvement of such codeword bit that has achieved a predefined quality threshold (θ). As a result, the complexity of the LDPC decoder will decrease.

Determining an optimum θ is a challenging task due to the fact that the optimum θ changes depending on the SNR. Setting θ to a lower value than the optimum one will result in a poor system performance. The system performance criteria used in [1] is measured in Block Error Rate (BLER), which should in average meet a Block Error Rate target ($BLER_t$) of 10^{-2}. A significant improvement in complexity savings was observed. However, the optimization approach of the *stochastic approximation* method that determines the optimum θ of the FC algorithm results in slow convergence. Slow convergence means that the system will take a long time to find an optimum θ. The slow convergence impacts the tracking capability of the optimization algorithm on time varying channels. While the system is trying to find this optimum θ, the channel will change and the optimum θ determined on the previous channel state will not be useful any more. Due to this

the system will not converge to an optimum state, leading to a higher complexity than the possible optimum one.

In this project, we focus on reducing the complexity of an LDPC decoding block of the receiver. The aim is to find the value of θ in the FC algorithm that will minimize the decoding complexity of the LDPC in a IEEE802.11n system, while keeping the $BLER$ at a minimum $BLER_t$. Most importantly, the convergence time of finding the optimum θ should be as short as possible, compared with the convergence time in [1]. To find the optimum θ in this project, an adaptive optimization approach of Performance Control Adaptation Algorithm ($PCAA$) is used. PCAA is a technique used in LTE power control algorithm for adjusting SNR offsets. The appeal of this algorithm is its ability to meet the minimum performance criteria of BLER along with faster system convergence. The PCAA will change the value of θ of the FC LDPC decoder according to the block decoding success or failure. Lastly, an extensive analysis of decoding optimization parameters under AWGN and frequency flat/selective fading channels, is performed and documented.

The work uses the same communication framework as used in [1], IEEE 802.11n standard. The communication system implements an OFDM access method and an LDPC FEC. Simulations are performed on an AWGN channel during the first stage of the project. This is the stage where the optimization parameters have been tested over a wide range of values. By trying different values and combinations for these parameters, a better trade-off between complexity and convergence is investigated. Performance and complexity calculations are shown in section 3.2.1. The average complexity savings, convergence, and θ behaviour are presented as a result of the simulations.

After testing the system in an AWGN channel, simulations are performed using an indoor environment of Rayleigh flat fading, time varying channel and finally for a frequency selective, time varying channel. These set-ups use the optimum decoding parameters obtained from the first stage of the project. The equalization in figure 1.1 is performed in time domain for the frequency flat fading, whereas it is performed in frequency domain for the frequency selective fading.

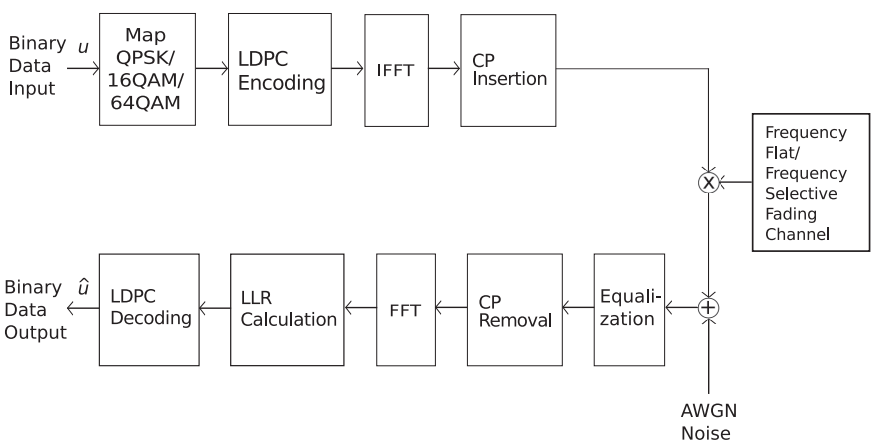


Figure 1.1: Block Diagram of Communication System

1.1 Outline of the Thesis

The thesis report firstly covers the concepts of parity-check constraint, Tanner graph representation of a parity-check matrix H, log likelihood algebra, and channel values LLR calculation. Finally, the major LDPC decoding algorithm is covered in detail with a pseudo code and an example in chapter 2. Chapter 3 provides a detailed overview of the FC OMS and PCAA algorithms that are used to solve the described problem and the development of the analytical model in complexity calculation of LDPC decoding. Pseudo code and numerical example are provided for a better understanding of these algorithms as well.

In chapter 4, the basics of OFDM communication transmission and its main limitations are discussed. Later, the chapter covers the signal power and SNR calculation formulae. Simulation parameters for the OFDM system, LDPC, PCAA and various channel configurations with simulation methodology are introduced in chapter 5.

In chapter 6, we present simulation results of LDPC coded OFDM signal transmission over an AWGN, frequency flat and frequency selective fading channel for various configurations of simulation parameters. This chapter also concludes the best configuration that provides a reasonable complexity reduction with the fastest convergence on AWGN and performance of this configuration in frequency flat and frequency selective channels. Finally, in chapter 7 concluding remarks with discussion on future works are presented.

Chapter 2

Theoretical Background of LDPC

Low-density parity-check (LDPC) codes are a class of forward error correction (FEC) codes that were first proposed by R. G. Gallager [2] in his PhD thesis in 1960's. The code remains neglected due to high computational effort in implementation until they were recovered by Mackay and Neal [3] in 1990's.

Significant changes has happened in the error control coding (ECC) field with the invention of turbo codes, the class of Shannon capacity approaching codes, by Berrou, Glavioux and Thitimajshima in 1993 replacing the widely used algebraic approach for successful error correction codes. Turbo codes feature iterative decoding, focus on average rather than worst case performance and use soft information (probability) of channel with manageable implementation complexity.

While research on turbo codes was active in 1990's, two researchers, McKay and Neal introduced a new class of block codes that were similar to turbo codes. It was later proved that the algorithm used to decode turbo code is a special case of LDPC decoding algorithm presented by Gallager in 1960's.

The availability of hardware that accommodates high computational complexity of LDPC decoding in 1990's initiated a huge research in LDPC codes leading to new generalization, irregular LDPC codes for instance, of LDPC codes presented by Gallager. Irregular LDPC codes outperform the best turbo codes and offer practical advantages in terms of implementation.

2.1 Error Correcting Codes

In this project, binary data messages are considered and hence the information sequence is strings of 0's and 1's. Error correcting codes in Galois field (GF(2)) map a sequence of binary data bits $\mathbf{u}$ into a longer codeword sequence $\mathbf{v}$. ECC has the capability to correct the transmission errors in $\mathbf{v}$ over a noisy channel and recover the data bits $\mathbf{u}$ given that the number of errors does not exceed the error correction capability.

The encoding of binary data sequence can be performed by using the following two different methods,

1. Block coding
2. Stream coding

In block coding, the data sequence is divided into blocks of K bits and coding is applied to each block. This application produces a codeword of N bits for each information bits block. In stream coding the data sequence is fed to a finite state machine (FSM) that can be represented by a trellis. FSM then produces one or more code bits for each data bit.

2.1.1 Error Correction using parity-checks

The essential idea of FEC coding is to add redundancy to the information bits in the form of check bits to produce a codeword for the message. The simplest coding scheme is to add a single parity bit check (SPC) to the message bits. In an even SPC code, the additional bit is added to the message bits in a way to make an even number of 1's in the codeword. For instance, for a 3 bits long message with even SPC we define a codeword $\mathbf{v}$ to have the following structure

$$v = [C_1\ C_2\ C_3\ C_4],$$

where each C_i can be 0 or 1 and all codewords satisfies the condition

$$C_1 \oplus C_2 \oplus C_3 \oplus C_4 = 0. \tag{2.1}$$

Equation (2.1) is called parity-check equation and the symbol $\oplus$ represents modulo-2 addition.

The SPC codeword is not powerful enough to detect more than one error in the transmitted bits sequence correctly. Furthermore, if there are even number of errors then the parity-check will fail to detect the error. In order to detect more than a single bit error, more redundancy in terms of parity-check bits is needed.

Example 1 A codeword structure is as follows

$$v = [c_1\ c_2\ c_3\ c_4\ c_5\ c_6],$$

and in order to be a valid codeword it is required for it to satisfy the following parity-check equations

$$\begin{aligned} C_1 \oplus C_2 \oplus C_4 &= 0 \\ C_2 \oplus C_3 \oplus C_5 &= 0 \\ C_1 \oplus C_3 \oplus C_4 \oplus C_6 &= 0. \end{aligned} \tag{2.2}$$

The codeword constraints are usually written in a matrix form and therefore constraints defined in equation (2.2) become

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

The matrix H is called a parity-check matrix. Each row corresponds to a parity-check equation and each column of H represents a codeword bit. A codeword $v = [c_1\ c_2\ c_3\ c_4\ c_5\ c_6]$ is a valid codeword if and only if it satisfies the following constraint equation

$$vH^T = 0, \tag{2.3}$$

where H is an $((N-K)\times N)$ order matrix with K as the number of message bits and N as the number of codeword bits. To generate the codeword for information bits the code constraints can be written in a systematic generator form

$$\begin{aligned} C_4 &= C_1 \oplus C_2 \\ C_5 &= C_2 \oplus C_3 \\ C_6 &= C_1 \oplus C_3 \oplus C_4 \,, \end{aligned} \tag{2.4}$$

and a systematic generator matrix can be written as follows

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

Given a 3 bit information message $\mathbf{u} = [c_1\ c_2\ c_3\ ]$, a codeword v can be obtained by using the following equation

$$\mathbf{v} = uG. \tag{2.5}$$

2.2 Description of LDPC Codes

LDPC codes are linear block codes characterized by a sparse parity-check matrix H. H is a low density matrix with a majority of zero entries. An (d_v, d_c) LDPC code determines that every codeword bit involves d_v parity-check equations and every parity-check equation involves d_c codeword bits. In irregular LDPC codes d_v and d_c vary for variable and check node and in regular LDPC codes d_v and d_c degree of each node remains the same for all the variable and check nodes. An example of regular (2,4) LDPC code ensemble is as follows

$$H = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 & 1 \end{bmatrix}. \tag{2.6}$$

An error correction code can be described by more than one parity-check matrix as long as equation (2.3) holds for all the codewords in a code. Two parity-check matrices for a same code are not necessarily required to have the same number of rows; what necessary is that the rank of GF(2) for both matrices should be the same and thus making it possible to have more than one ensemble of H matrix for the same code. The number of information bits, k, in a binary code is

$$n - k = \text{rank}(H), \tag{2.7}$$

where the rank of H matrix is the linearly independent rows of H over GF(2). Each parity-check equation typically reduces the number of degrees of freedom by one, the design rate of the (d_v, d_c) regular LDPC code is

$$R(d_v, d_c) \geq \frac{N-K}{N} = 1 - \frac{d_v}{d_c}, \tag{2.8}$$

where equality holds when H is a full rank matrix.

2.2.1 Graphical Representation of LDPC Codes

A Tanner graph is an effective way to describe LDPC codes graphically [4]. It may be used to describe the constraints that a codeword must fulfil in order to belong to a particular linear code block. Furthermore, it is convenient to use a Tanner graph to describe the iterative message passing decoding. In a bipartite Tanner graph, there are two sets of nodes: variable nodes (v-nodes) for codeword bits and check nodes (c-nodes) for parity-check equations. An edge connects a variable node to a check node if that codeword bit is included in a parity-check equation. A node has a degree i if i branches leave from that node and connect to i different nodes of other kind. Figure 2.1 shows a Tanner graph which is related to a parity-check described matrix H in section 2.2.

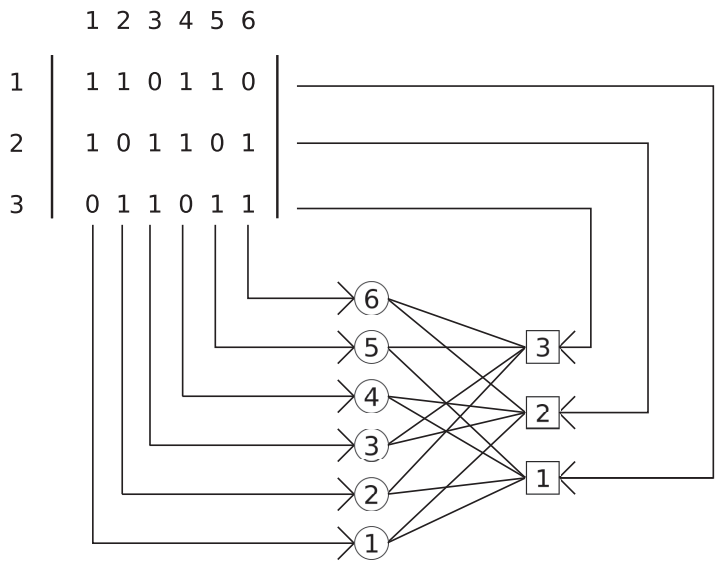


Figure 2.1: Tanner graph representation of regular (2,4) LDPC code

2.3 Decoding Algorithms for LDPC Codes

Decoding of LDPC code is performed by using the message passing algorithms since their operation can be explained by sending the message along the edges of a Tanner graph. These algorithms are also known as iterative decoding algorithms as the message passes back and forth in between parity and check nodes. The most common type of algorithms, such as belief propagation (BP), use probabilities as messages to exchange in between variable and check nodes. V-node sends

an estimated *a priori* message to its connected c-node. The c-node, after receiving *a priori* messages from its connected v-node, performs the parity constraints computation and returns the extrinsic information. This processing of variable and check node constitutes one iteration. It is often easier to use log likelihood ratio to represent estimated probabilities, in that case BP decoding is often called sum-product (SP) decoding.

Before describing these algorithms in detail, an overview of Log Likelihood Ratios (LLR) calculations of probability distributions is described in section 2.3.1. In this project, low-complexity approximation of sum-product (SP) algorithm is used. This low complexity version of SP algorithm is known as the offset min-sum (OMS) algorithm [7]. *Layered* scheduling [8], [9] is used for block decoding. In layered scheduling, the decoder begins with first layer, corresponding to the first row of parity-check matrix, and exchange the LLR messages within v-nodes and c-node of a layer. Decoding follows the same procedure on all layers in a sequential manner. At the end of the last layer decoding, the LDPC decoder is said to have completed one iteration with updated *a posteriori* LLR at its v-nodes.

2.3.1 Channel Model and Log-Likelihood (LLR) Calculations

We here consider the following system model

$$y = hx + n, \tag{2.9}$$

where $x = \{1, -1\}, y, n \in \Re$ represents the channel input, output and noise respectively. $\Re$ denotes the set of real numbers and $h \geq 0$ is the fading channel gain with Rayleigh distribution by its probability distribution. In this study n is white Gaussian noise with zero mean and variance σ_n^2 and channel knowledge is assumed to be known at the receiver.

For binary codes we use soft values for *a priori* probabilities of received symbols. These soft values are also termed as *a priori* Log Likelihood Ratio (LLR) of bit x. A log likelihood ratio for the bit x with an *a priori* probability $p(x)$ is defined in equation (2.10), it is to be noted that 0 is mapped to $+1$ and 1 is mapped to -1

$$L(x) = \ln \frac{p(x=0)}{p(x=1)}. \tag{2.10}$$

If $p(x=0) > p(x=1)$ then $L(x)$ will be positive and negative otherwise. If $L(x) = 0$ then we have $p(x=0) = p(x=1)$. The sign of $L(x)$ provides hard decision on bit x and the magnitude $|L(x)|$ indicates reliability of the decision.

Similarly the LLR value of an *a posteriori* probability $p(x|y)$ for each codeword bit can be computed by using

$$L(x|y) = \ln \frac{p(x=0|y)}{p(x=1|y)}, \tag{2.11}$$

and for the channel transition probability $p(y|x)$ by using

$$L(y|x) = \ln \frac{p(y|x=0)}{p(y|x=1)}. \tag{2.12}$$

By using Bayes' theorem

$$L(x|y) = L(y|x) + L(x), \tag{2.13}$$

where $L(x)$ is *a priori* information of a codeword bit or v-node. The check node will perform the box-plus $\boxplus$ operation to retrieve extrinsic or refined *a priori* information for a particular v-node from the incoming messages of connected v-nodes. $\boxplus$ expresses the reliability of a modulo-2 sum of bits in terms of individual bits LLRs. $L(x)$ is initialized to zero assuming equally likely bits in the first iteration. However, the estimate of $L(x)$ will be refined with successive iterations.

$$L(x_1) \boxplus L(x_2) = L(x_1 \oplus x_2),$$

LLR value for the $\boxplus$ operation is

$$L(x_1) \boxplus L(x_2) = \ln \frac{p(x_1 \oplus x_2 = 0)}{p(x_1 \oplus x_2 = 1)},$$

$$L(x_1) \boxplus L(x_2) = \ln \frac{p(x_1 = 0)p(x_2 = 0) + p(x_1 = 1)p(x_2 = 1)}{p(x_1 = 1)p(x_2 = 0) + p(x_1 = 0)p(x_2 = 1)}. \tag{2.14}$$

The individual bit probabilities can be obtained from equation (2.10) and the fact that $p(x = 0) + p(x = 1) = 1$. The expression for bit probability, given that the LLR of that bit is available, is given in equations (2.15) and (2.16)

$$p(x = 0) = \frac{e^{L(x)}}{1 + e^{L(x)}}, \tag{2.15}$$

and

$$p(x = 1) = \frac{e^{-L(x)}}{1 + e^{-L(x)}}. \tag{2.16}$$

Using equations (2.14), (2.15) and (2.16), an expression to compute extrinsic message is as follows

$$L(x_1) \boxplus L(x_2) = \ln \frac{1 + e^{L(x_1)} e^{L(x_2)}}{1 + e^{L(x_1)} e^{L(x_2)}} = \ln \frac{1 + \tanh(L(x_1)/2) \tanh(L(x_2)/2)}{1 - \tanh(L(x_1)/2) \tanh(L(x_2)/2)}, \tag{2.17}$$

where tangent hyperbolic function is

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}.$$

Since $L(x)$ is initialized as zero, for the first iteration the *a posteriori* LLR $L(x|y)$ in equation (2.13) yields to a channel transition probability $L_{ch} = L(y|x)$. For an AWGN channel, the channel LLR is obtained by using the following relation

$$L_{ch} = \ln \frac{p(y|x = +1, h)}{p(u|x = -1, h)}, \tag{2.18}$$

$$p(y|x,h) = \frac{1}{\sqrt{2\pi\sigma_n^2}} e^{-\frac{(y-hx)^2}{2\sigma_n^2}}. \tag{2.19}$$

By using Gaussian distribution (equation (2.19)) in equation (2.18), the expression of the channel LLR for BPSK modulation is

$$L_{ch} = \frac{2\alpha y}{\sigma_n^2}, \tag{2.20}$$

where $\alpha = \mathbb{E}[|h|^2]$ is the gain of channel h that a block experienced.

2.3.2 Sum-Product (SP) Algorithm

Sum-product LDPC decoding algorithm is an iterative technique that takes *a priori* and channel transition probabilities of the received symbols. For the SP algorithm these probabilities are expressed in terms of LLR. The method to calculate the LLR is described in section 2.3.1. The decoder returns the *a posteriori* LLRs of codeword bits at each iteration. Binary hard decision is performed on the *a posteriori* LLRs at the end of each iteration followed by the parity-check equation (2.3) to see if the parity constrained is satisfied or not. In case the equation (2.3) is valid for the decoded bits, the decoder will terminate the iterations.

SP algorithms compute a *maximum a posteriori probability* (MAP) LLR for each codeword bit i, $P_i = P(v_i = 1|N)$ which is the probability that the i_{th} bit is 1 (in this project binary 0 is mapped to 1) conditioned on the event that all parity-check constraints are satisfied. The MAP LLR of each codeword bit will increase with each decoding iteration and thus improving the reliability of hard decision. The major reason of this improvement is because of the messages exchange within a layer decoding. After performing parity-checking constraints using equation (2.17), the c-node in a layer sends an extra information, also known as *extrinsic* information, of the codeword bit i to its v-node i which improves the MAP estimates. Figure 2.2 shows the c-node update process of the edge j using l and d_c edges. The method for calculating the edges messages update and the *a posteriori* LLRs update is described in *"Algorithm Sum-Product Decoding"* subsection.

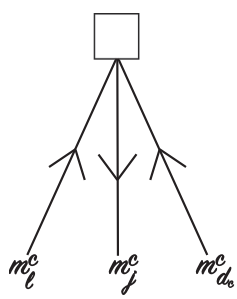


Figure 2.2: Check node update and extrinsic information transmission

An important point worth highlighting is that the sum-product algorithm is optimal only when the graph of the code is cycle free. It means that the estimates

of the MAP are exact only for the cycle free code. As LDPC codes involves cycles, their decoding by SP algorithm leads to self-confirmation of bit which degrades the convergence and make the algorithm sub-optimal.

The following section describes briefly the SP algorithm assuming that the received signal y. Channel LLR (L_{ch}) is calculated using equation (2.20) for BPSK, AWGN non fading channel.

Algorithm SP Decoding

Terminologies used in the SP algorithm description and their meaning are

- C and V stands for check nodes and variable nodes respectively.
- C-node j connecting to v-node i is denoted by $e_{ij=k}$ and c-node update is represented by $L_{cv}(e_k)$.
- $L_{vc}(e_k)$ stands for a v-node edge message and $L_v(v)$ represents an *a posteriori* probability of v-node v.
- $d_c(C)$ represents the number of edges connecting c-node to its associated v-nodes.

1. **Initialization:** Compute $L_{ch}(v)$ using equation (2.20), set iteration (itr) = 1, $L_v(v) = L_{ch}(v)$, $L^0_{vc}(e_{ij}) = L_v(v)$ and $L^0_{cv}(e_{ij}) = 0$, where e_{ij} represents edge connecting v-node i to c-node j.
2. **Variable node edges update:** For all v-nodes v and edges e_k

$$L^{(itr)}_{vc}(e_k) = L^{(itr-1)}_{v}(v) - L^{(itr-1)}_{cv}(e_k).$$

3. **Check node edges update:** For all c-nodes c and edges $e_{(ij=k)}$.

$$\begin{aligned} L^{(itr)}_{cv}(e_k) &= \boxplus_{k' \neq k} L^{(itr)}_{vc}(e_{k'}) \\ &= \tanh^{-1}\Big(\prod_{k' \neq k} \tanh\big(\tfrac{1}{2} L^{(itr)}_{vc}(e_{k'})\big)\Big). \end{aligned} \tag{2.21}$$

4. **Variable node updates:**

$$L^{(itr)}_{v}(v) = L^{itr}_{vc}(e_k) + L^{itr}_{cv}(e_k).$$

2.3.3 Min Sum Decoding

The min Sum algorithm is a reduced complexity version of SP decoding. This is the main algorithm used in this project for LDPC decoding and to study the impact of PCAA on overall complexity reduction of decoding.

Simplification to SP algorithm [6] is achieved by replacing equation (2.21) with

$$L^{(itr)}_{c}(e_k) = \text{sign}\Big(\prod_{k' \neq k} L^{(itr)}_{v}(e_{k'})\Big) \cdot \min_{k' \neq k} \big|L^{(itr)}_{v}(e_{k'})\big| \approx \boxplus_{k' \neq k} L^{(itr)}_{v}(e_{k'}). \tag{2.22}$$

Derivation of equation (2.22) can be seen in [5]. This reduces the extrinsic message calculation operation in equation (2.22) with finding the absolute minimum and product of signs of incoming messages at the check nodes. A brief description of the Min Sum algorithm is given in algorithm 1.

Algorithm 1 Min Sum Layered LDPC Decoding

procedure DECODING(r) ▷ r is channel values $L_{ch}(v)$ of received bits
  $itr = 0$
  for $C \leftarrow 1, layers$ **do** ▷ Initialization
    for $k \leftarrow 1, d_c(C)$ **do**
      $L_{cv}^{(itr)}(e_k) = 0$ ▷ Initialize check node edges to zero
    end for
  end for
  for $v \leftarrow 1, N$ **do**
    $L_{v}^{(itr)}(v) = L_{ch}(v)$ ▷ Initialize variable nodes
  end for
  for $itr \leftarrow 1, MaxIterations$ **do** ▷ LDPC decoding iterations
    for $C \leftarrow 1, layers$ **do**
      for $k \leftarrow 1, d_c(C)$ **do** ▷ variable node edges update
        $L_{vc}^{(itr)}(k) = L_{v}^{(itr-1)}(v) - L_{cv}^{(itr-1)}(k)$
      end for
      for $k \leftarrow 1, d_c(C)$ **do** ▷ Check node edges update
        $L_{cv}^{(itr)}(k) = \text{sign}(\prod_{k' \neq k} L_{vc}^{(itr)}(e_{k'})) \cdot \min_{k' \neq k} |L_{vc}^{(itr)}(e_{k'})|$
      end for
      for $k \leftarrow 1, d_c(C)$ **do** ▷ variable node update
        $L_{v}^{(itr)}(v) = L_{vc}^{itr}(e_k) + L_{cv}^{itr}(e_k)$
      end for
    end for
    $\mathbf{z}$ = Hard decision decoding on L_v
    if $zH^T = 0$ **then** ▷ parity-check constraint
      Exit LDPC Decoding Program
    end if
  end for
end procedure

Simplification of c-node edges update in the Min Sum algorithm introduces performance loss compared to SP algorithm. To account for this loss, an additive correction factor ω is applied to c-node edges update. In this project, the Offset Min Sum (OMS) algorithm is applied, which introduces an additive factor of $\omega = .3$ to the c-node edges update. The OMS algorithm is described in section 3.1.

Min Sum Algorithm Example

Example 2 Consider a system with BPSK mapping of 0 to 1 and 1 to -1 through an AWGN channel. A parity-check matrix (H) and received channel values for transmitted codeword sequence are

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

$$L_{ch} = [1.0 \quad 0.5 \quad 0.5 \quad 2.0 \quad 1.0 \quad -1.5 \quad 1.5 \quad -1.0]$$

corresponding to the transmitted codeword bits of

$$\mathbf{v} = [0 \quad 0 \quad 0 \quad 0 \quad 1 \quad 1 \quad 0 \quad 1]$$

Initialization: Before LDPC decoder starts decoding iterations, all v-nodes and c-nodes edges are initialized.

$L_{vc}^{(0)}(e_k) = L_{ch}(v)$ for all $v = v_1, ...v_8$ and $L_{cv}^{(0)}(e_k) = 0$ for all c-nodes edges

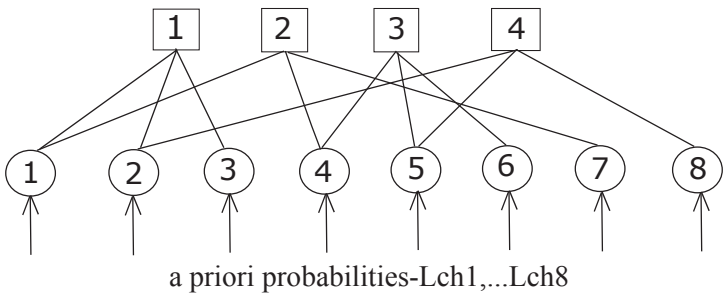
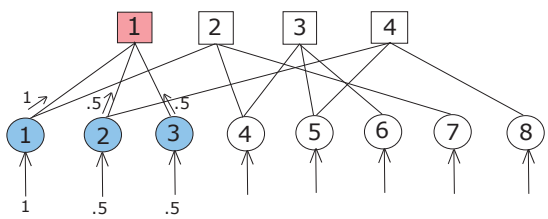


Figure 2.3: Example 2 Tanner graph of H and variable nodes initialization

Iteration 1: After initialization, decoder performs its first iteration and updates the v-nodes and c-nodes in a manner as described in algorithm 1.

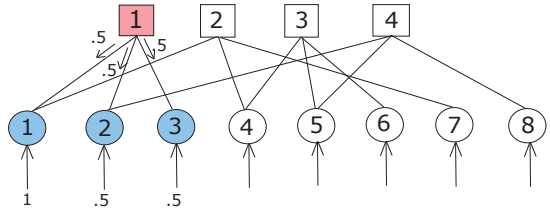
Variable node edges update layer 1

$Lv_{11}^{(1)}(1) = 1 - 0 = 1$
$Lv_{21}^{(1)}(2) = .5 - 0 = .5$
$Lv_{31}^{(1)}(3) = .5 - 0 = .5$



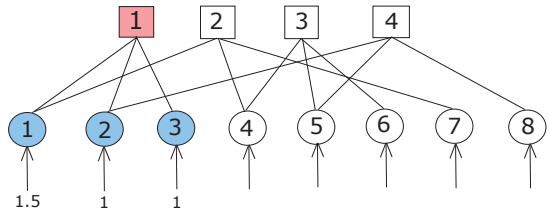
Check node edges update layer 1

$Lc_{11}^{(1)}(1) = .5$
$Lc_{12}^{(1)}(2) = .5$
$Lc_{13}^{(1)}(3) = .5$



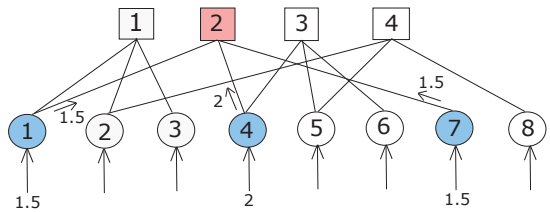
Variable nodes update layer 1

$Lv^{(1)}(1) = 1.5$
$Lv^{(1)}(2) = 1$
$Lv^{(1)}(3) = 1$



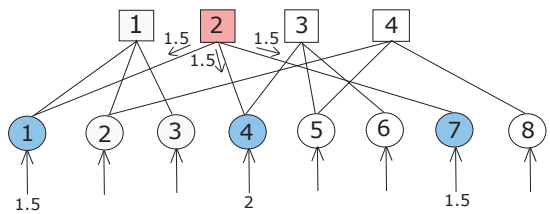
Variable node edges update layer 2

$Lv_{12}^{(1)}(1) = 1.5$
$Lv_{42}^{(1)}(2) = 2$
$Lv_{72}^{(1)}(3) = 1.5$



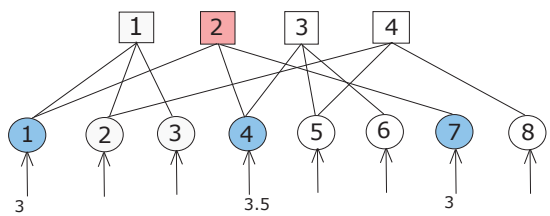
Check node edges update layer 2

$Lc_{21}^{(1)}(1) = 1.5$
$Lc_{24}^{(1)}(2) = 1.5$
$Lc_{27}^{(1)}(3) = 1.5$



Variable nodes update layer 2

$Lv^{(1)}(1) = 3$
$Lv^{(1)}(4) = 3.5$
$Lv^{(1)}(7) = 3$



Following the same procedure, updates for layer 3 and layer 4 are

Layer 3 decoding results

$Lv_{43}^{(1)}(1) = 3.5$ $Lc_{34}^{(1)}(1) = 1.5$ $Lv^{(1)}(4) = 2.5$
$Lv_{53}^{(1)}(2) = 1$ $Lc_{35}^{(1)}(2) = 1.5$ $Lv^{(1)}(5) = -.5$
$Lv_{63}^{(1)}(3) = -1.5$ $Lc_{36}^{(1)}(3) = 1.5$ $Lv^{(1)}(6) = -.5$

Layer 4 decoding results

$Lv_{24}^{(1)}(1) = 1$ $Lc_{42}^{(1)}(1) = .5$ $Lv^{(1)}(2) = 1.5$
$Lv_{54}^{(1)}(2) = -.5$ $Lc_{45}^{(1)}(2) = -1$ $Lv^{(1)}(5) = -1.5$
$Lv_{84}^{(1)}(3) = -1.0$ $Lc_{48}^{(1)}(3) = -.5$ $Lv^{(1)}(8) = -1.5$

At the end of 4^{th} layer decoding, decoder will have following *a posteriori* LLRs

$$L_{out} = [3.0 \quad 1.5 \quad 1.0 \quad 2.5 \quad -1.5 \quad -0.5 \quad 3.0 \quad -1.5].$$

Hard decision decoding of L_{out} yields to the correct transmitted codeword sequence after the first iteration. In case of an invalid parity constraint check (equation (2.3)), the decoder will keep iterating until the validity of parity constraint check or the number of iterations become equal to the maximum permissible iterations.

2.4 *A posteriori* LLRs evolution with decoding iterations

With each decoding iteration reliability of *a posteriori* probabilities (LLRs) will increase. It is possible that the reliability of some bits grows at faster rate compared to others in a block. Furthermore, the LLRs growth with iterations at higher SNR increases at a much faster rate compared to the LLRs growth with iteration at lower SNRs. Comparison of decoding iteration 1, 7 and 15 for the average LLRs of all the codeword bits and the individual bits of BPSK, $N = 648, R = 1/2$ LDPC encoded codeword is depicted in figure 2.4 and 2.5.

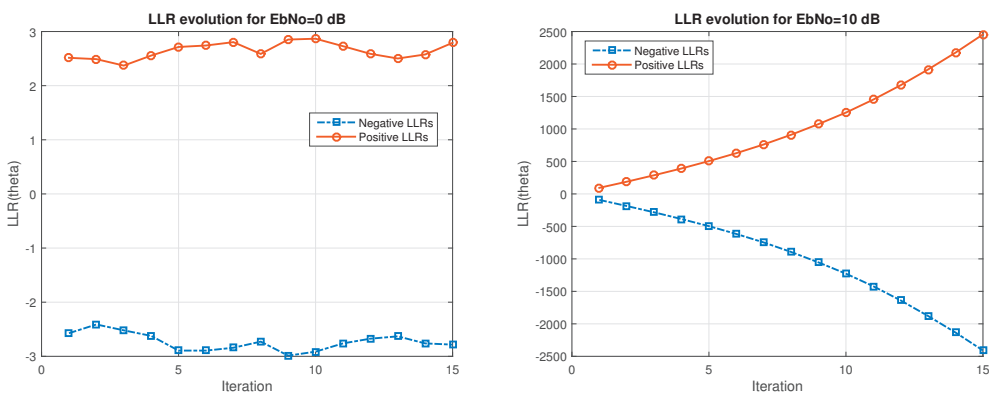


Figure 2.4: Average LLR of variable nodes with iteration

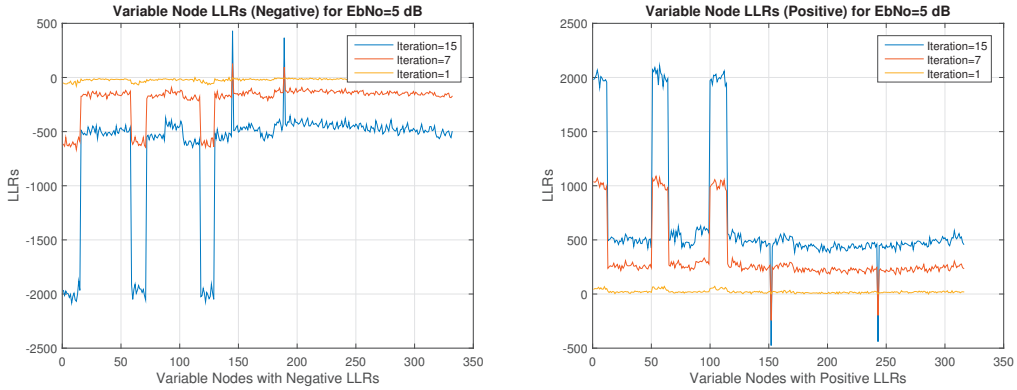


Figure 2.5: Individual bits LLRs with iterations

Chapter 3

Forced Convergence and PCAA

As described in the previous section, each v-node has an *a posteriori* LLR $L_v(v)$ that grows as the iteration increases. As a result of this process the reliability of a hard decision decoding on these LLRs will increase as well. For $SNR \geq SNR_{operating}$, where $SNR_{operating}$ is the range of SNRs on which plain OMS (non FC LDPC decoding) achieves $BLER \leq BLER_t = 10^{-2}$, the average LLR values of v-nodes increases as the iteration progresses. However, further updates on highly reliable nodes will increase the computation cost with less benefits in terms of performance gain. The idea of FC is to stop updating the v-node (v) *a posteriori* LLR once it achieves a certain LLR threshold, termed as θ in this project, to reduce the complexity of decoding while maintaining a desired system performance $BLER_t$. This θ is the same quality threshold as mentioned in the introduction.

The choice of $BLER_t$ impacts the quality of the system. Since the reliability of a certain v-node impacts the reliability calculation of the associated v-nodes in a layer, freezing the v-node to a very low value of θ may result in performance degradation. Controlling θ adaptively, given a minimum performance criterion, ensures that the required performance is achieved while keeping decoding complexity as minimum as possible.

This chapter covers the technical details of the solution that is proposed to reduce the complexity of the receiver as well as to make the FC LDPC decoder convergence faster to achieve the optimum θ. FC LDPC decoding algorithm, complexity computations and adaptive θ control algorithm are discussed in brief in later sections.

3.1 Forced Convergence

For this project, a modified layered scheduling MS algorithm, Offset Min Sum (OMS), is used with FC. Layered Min Sum algorithm is covered in detail in section 2.3.3 and in example 2. An important observation of equation (2.22) is that the c-node edges update is influenced by the minimum value of L_{vc}, which makes it possible to freeze a v-node to a large enough θ value [1]. The OMS FC algorithm can be realized by applying the following modifications to the Min Sum algorithm

1. **Variable node edges L_{vc} calculation:**

$$L_{vc}^{(itr)}(k) = \begin{cases} L_{v,frozen}, & \text{if } v \text{ is frozen} \\ L_{v}^{(itr-1)}(v) - L_{cv}^{(itr-1)}(k), & \text{otherwise.} \end{cases}$$

2. **Check node edges L_{cv} calculation:**

$$L_{cv}^{(itr)}(k) = \text{sign}\Big(\prod_{k' \neq k} L_{vc}^{(itr)}(e_{k'})\Big) \cdot \max\Big\{\min_{k' \neq k} \big|L_{vc}^{(itr)}(e_{k'})\big| - \omega, 0\Big\},$$

where ω is the offset, k is the edge in update and k' are set of remaining edges in the layer connecting the check node to the variable nodes. The most commonly used offset, $\omega = .3$, is applied for correcting the check node estimates in this project. However, it is discussed in [14] that an adaptive offset application yields to better performance than a constant offset term. The application of the Adaptive Offset Min Sum (AOMS) is out of scope of this project and it can be used as a future extension.

3. **Variable nodes L_v calculation:**

If a variable node (v) is frozen then the variable node update will not be performed for that particular v-node, however, if the v-node is still active then the following procedure is applied

$$L_{v}^{(itr)}(v) = L_{vc}^{(itr)}(e_k) + L_{cv}^{(itr)}(e_k),$$

if $|L_{v}^{(itr)}(v)| \geq \theta$ then node will be frozen with an updated value of $(\text{sign}(L_{v}^{(itr)}(v)) \cdot \theta)$.

The pseudo code formulation of the modified algorithm is described in algorithm 2. The algorithm uses the same notations as given in algorithm 1. Additional terminologies used in the algorithm are as follows,

- $L_{vc}(min1)$ and $L_{vc}(min2)$ represents the two least valued v-nodes edges in a layer. Edge1 terminates at v-node v_{min1} and Edge2 terminates at v_{min2}.
- ω is the offset applied in the OMS algorithm. For the current project, $\omega = .3$ is used.
- z is the hard decision vector obtained after processing the L_v *a posteriori* LLRs.

Algorithm 2 Offset Min Sum with FC and extrinsic message simplification

```
1: procedure DECODING(r)  ▷ r is channel values L_ch(v) of received bits
2:     itr = 0
3:     for C ← 1, layers do                                  ▷ Initialization
4:         for k ← 1, d_c(C) do
5:             L_cv^(itr)(e_k) = 0          ▷ Initialize check node edges to zero
6:         end for
7:     end for
```

Algorithm 3 Offset Min Sum with FC and extrinsic message simplification (continued)

for $v \leftarrow 1, N$ **do**
  $L_v^{(itr)}(v) = L_{ch}(v)$ ▷ Initialize variable nodes
end for
for $itr \leftarrow 1, MaxIterations$ **do** ▷ LDPC decoding iterations
  for $C \leftarrow 1, layers$ **do**
    for $k \leftarrow 1, d_c(C)$ **do** ▷ variable node edges update
      if $v \in Inact$ **then**
        $L_{vc}^{(itr)}(k) \leftarrow L_v(v)$
      else
        $L_{vc}^{(itr)}(k) \leftarrow L_v^{(itr-1)}(v) - L_{cv}^{(itr-1)}(k)$
      end if
    end for
    $L_{vc}(min1) \leftarrow \min_{v \in d_c(C)} \{|L_{vc}(v)|\}$ ▷ Check node edges update
    $L_{vc}(min2) \leftarrow \min_{v \in d_c(C), v \neq v_{min1}} \{|L_{vc}(v)|\}$
    $L_{vc}(min1) \leftarrow \max\{L_{vc}(min1) - \omega, 0\}$
    $L_{vc}(min2) \leftarrow \max\{L_{vc}(min2) - \omega, 0\}$
    $S = \prod_{v \in d_c(C)} \text{sign}(L_{vc}(v))$
    for $v \leftarrow 1, d_c(C)$ **do**
      if $v \notin Inact$ **then**
        if $v = v_{min1}$ **then**
          $L_{cv}(v) \leftarrow \text{sign}(L_{vc}(v)) \cdot S \cdot L_{vc}(min2)$
        else
          $L_{cv}(v) \leftarrow \text{sign}(L_{vc}(v)) \cdot S \cdot L_{vc}(min1)$
        end if
      end if
    end for
    for $k \leftarrow 1, d_c(C)$ **do** ▷ variable node update
      if $v \notin Inact$ **then**
        $L_v^{(itr)}(v) = L_{vc}^{(itr)}(e_k) + L_{cv}^{(itr)}(e_k)$
        if $|L_v^{(itr)}(v)| > \theta$ **then**
          $|L_v^{(itr)}(v)| \leftarrow \theta$
          $v :\in Inact$
        end if
      end if
    end for
  end for
  if $zH^T = 0$ **then** ▷ parity-check constraint
    Exit LDPC Decoding Program
  end if
end for
end procedure

3.1.1 Example FC

Example 3

Consider example 2 but this time instead of Min Sum decoding, OMS-FC decoding with $\theta = 2$ is performed.

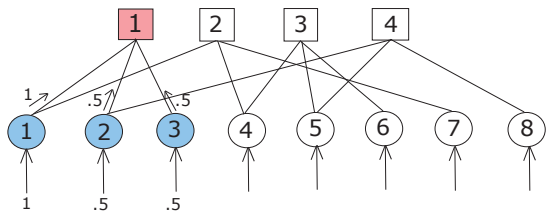
Initialization: Decoder first initializes the variable nodes and check nodes before starting iterations.

$L_{vc}^{(0)}(e_k) = L_{ch}(v)$ for all $v = v_1, ...v_8$, and $L_{cv}^{(0)}(e_k) = 0$ for all check nodes edges.

Iteration 1

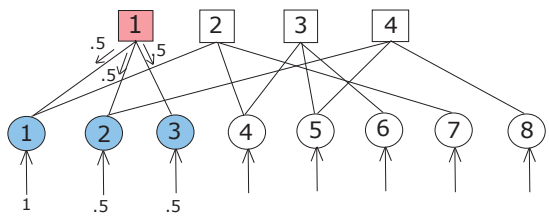
Variable node edges update layer 1

$Lv_{11}^{(1)}(1) = 1 - 0 = 1$
$Lv_{21}^{(1)}(2) = .5 - 0 = .5$
$Lv_{31}^{(1)}(3) = .5 - 0 = .5$



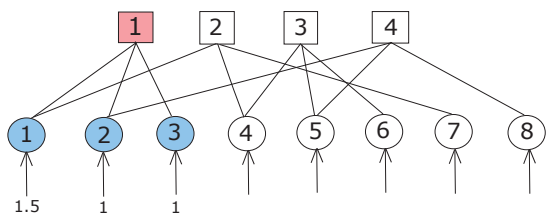
Check node edges update layer 1

$Lc_{11}^{(1)}(1) = .5$
$Lc_{12}^{(1)}(2) = .5$
$Lc_{13}^{(1)}(3) = .5$



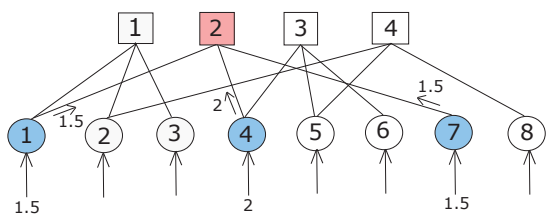
Variable nodes update layer 1

$Lv^{(1)}(1) = 1.5$
$Lv^{(1)}(2) = 1$
$Lv^{(1)}(3) = 1$



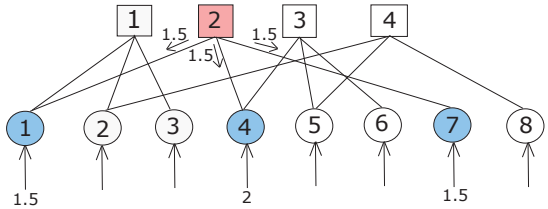
Variable node edges update layer 2

$Lv_{12}^{(1)}(1) = 1.5$
$Lv_{42}^{(1)}(2) = 2$
$Lv_{72}^{(1)}(3) = 1.5$



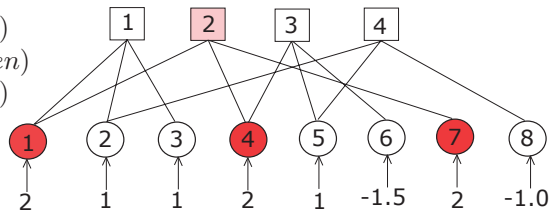
Check node edges update layer 2

$Lc_{21}^{(1)}(1) = 1.5$
$Lc_{24}^{(1)}(2) = 1.5$
$Lc_{27}^{(1)}(3) = 1.5$



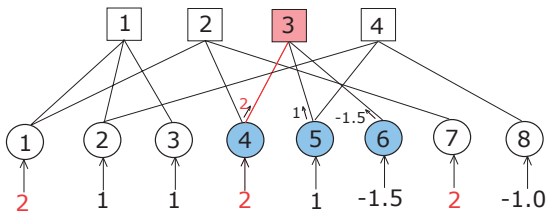
Variable nodes update layer 2

$Lv^{(1)}(1) = 3 > \theta = 2(Frozen)$
$Lv^{(1)}(4) = 3.5 > \theta = 2(Frozen)$
$Lv^{(1)}(7) = 3 > \theta = 2(Frozen)$



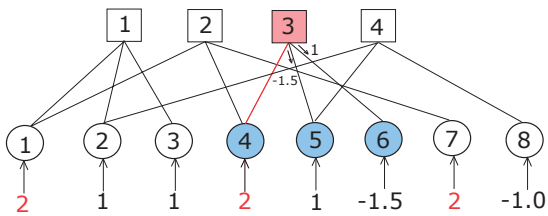
Variable node edges update layer 3

$Lv_{43}^{(1)}(1) = 2(Frozen)$
$Lv_{53}^{(1)}(2) = 1 - 0 = 1$
$Lv_{63}^{(1)}(3) = -1.5 - 0 = -1.5$



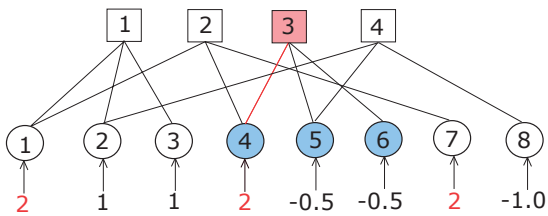
Check node edges update layer 3

$Lc_{34}^{(1)}(1) = NA(Frozen)$
$Lc_{35}^{(1)}(2) = -1.5$
$Lc_{36}^{(1)}(3) = 1$



Variable nodes update layer 3

$Lv^{(1)}(4) = NA(Frozen)$
$Lv^{(1)}(5) = -0.5$
$Lv^{(1)}(6) = -0.5$



For the 4^{th} layer there is no frozen node involved and its decoding results are,

Layer 4 decoding results

$Lv_{24}^{(1)}(1) = 1$ $Lc_{42}^{(1)}(1) = .5$ $Lv^{(1)}(2) = 1.5$
$Lv_{54}^{(1)}(2) = -.5$ $Lc_{45}^{(1)}(2) = -1$ $Lv^{(1)}(5) = -1.5$
$Lv_{84}^{(1)}(3) = -1.0$ $Lc_{48}^{(1)}(3) = -.5$ $Lv^{(1)}(8) = -1.5$

Finally, at the end of the 4^{th} layer decoding, the decoder will have the following *aposteriori* LLRs

$$L_{out} = [2.0 \quad 1.5 \quad 1.0 \quad 2 \quad -1.5 \quad -0.5 \quad 2.0 \quad -1.5].$$

In case of an invalid parity constraint check equation (2.3) on the hard decoded output of *a posteriori* LLRs, the decoder will keep iterating until the validity of equation (2.3) or the number of iterations become equal to the maximum permissible iterations.

3.2 LDPC Decoder Parameters Optimization

In [10], the θ for a general vector ρ of environment settings (such as SNR or fading properties) was adapted to the channel conditions by applying *stochastic approximation methods* to the complexity function $C(\theta, \rho)$ subjected to $BLER(\theta, \rho) \leq BLER_t$. In this project, the optimization algorithm PCAA is used to adaptively compute θ for the given channel conditions. It is of interest to see that how fast the LDPC decoder can converge to the optimum θ using PCAA and how much complexity savings can be achieved relative to a plain OMS decoding. The optimum θ will be referred as $\mathbb{E}[\theta_{critical}]$ and further details on $\mathbb{E}[\theta_{critical}]$ are discussed in the PCAA section.

3.2.1 Analytical Model of Complexity Computation

The OMS-FC algorithm 2 is the basis of developing an analytical model of complexity computation. In [10], the complexity is given in a number of additions (assumed equivalent in complexity as comparison) performed per decoded block. IEEE 802.11n employs irregular LDPC codes [13], therefore, the complexity of the layers may differ depending on the parity-check matrix d_c degree. For all layers with the same d_c, complexity of each layer will be the same. For performance comparison among IEEE 802.11n suggested MCSs [13], the complexity calculation algorithm described below [1] provides the complexity of decoding on the number of operations per bit resolution.

- Complexity of L_{vc} edges update in one layer is

$$\tilde{C}_{L_{vc}}^{(b,i,l)} = \tilde{n}_a^{(b,i,l)},$$

where $\sim$ denotes a random variable and b, i, l represent decoded blocks, in iteration i and layer l respectively. $\tilde{n}_a^{(b,i,l)}$ indicates the number of active v-nodes in layer l, iteration i and block b.

- L_{cv} edges update in one layer, first find the two minimum values, using an unsorted array method, in the set of $|L_{vc}|$ of the subjected layer. L_{cv} edge update will not be performed for frozen v-node, therefore, complexity of L_{cv} update on active nodes is

$$\tilde{n}_x^{(b,i,l)} = \min\{\tilde{n}_a^{(b,i,l)} + 1, n^{(l)}\},$$

and total complexity of this section is

$$\tilde{C}_{L_{cv}}^{(b,i,l)} = \tilde{n}_x^{(b,i,l)} + \lceil \log_2 \tilde{n}_x^{(b,i,l)} \rceil + 2. \tag{3.1}$$

The second term in equation (3.1) is the complexity of finding the two minimum elements in an unsorted array and the third term of equation is the four additions on line 22 and 23 of algorithm 2.

$n^{(l)}$ represents the number of associated v-nodes to c-node in a layer.

- Lastly the complexity of L_v update is

$$\tilde{C}_{L_v}^{(b,i,l)} = 2\tilde{n}_a^{(b,i,l)}.$$

Total complexity of decoding one layer is

$$\tilde{C}^{(b,i,l)} = 3\tilde{n}_a^{(b,i,l)} + \tilde{n}_x^{(b,i,l)} + \lceil \log_2 \tilde{n}_x^{(b,i,l)} \rceil + 2,$$

and the complexity of decoding one block for all iterations is

$$\tilde{C}^{(b)} = \sum_{i=1}^{\tilde{I}^{(b)}} \sum_{l=1}^{|c|} \tilde{C}^{(b,i,l)}. \tag{3.2}$$

$\tilde{I}^{(b)}$ in equation (3.2) is the number of iterations it takes to complete one block decoding and $|c|$ is the number of check nodes or layers.

3.2.2 Performance Control Adaptation Algorithm (PCAA)

PCAA, a terminology that is defined for this project, is a technique used in the Outer Loop Link Adaptation (OLLA) in LTE. This technique adjusts the SNR offsets used when choosing an appropriate modulation and coding scheme [11]. PCAA ensures that on average $BLER_t$ is met. The analytical expression for PCAA θ is

$$\Delta\theta_{up} = \left(\frac{1}{BLER_t} - 1\right) \Delta\theta_{down}. \tag{3.3}$$

$\Delta\theta_{up}$ and $\Delta\theta_{down}$ in equation (3.3) are upward and downward jumps in θ. For successful block decoding, θ will be decremented by $\Delta\theta_{down}$ and for block decoding

failure, θ will be increased by $\Delta\theta_{up}$. This will ensure that $BLER_t$ has been met on an average block level.

Algorithm 4 PCAA

```
1: procedure ADAPTATION(Block_Error)   ▷ PCAA after LDPC decoding of each block
2:     if Block_Error = False then      ▷ PCAA execution
3:         θ = θ − Δθ_down
4:     else
5:         θ = θ + Δθ_up
6:     end if
7: end procedure
```

PCAA Modelling and Behaviour Analysis

To understand the underlying mechanism of PCAA and how it ensures meeting $BLER_t$, a development of the probabilistic model is required that takes into account the LDPC optimization parameters and estimates the probability of system being into a state S_m at time t. The state represents the θ value calculated by PCAA for a certain block. Since the determination of transitions probabilities among states satisfies the Markov property, PCAA behaviour can be modelled by a discrete time Markov process, a technique used in analysis of OLLA as well [11].

In order to develop the model, the following is performed

1. Identification of possible states.
2. Identification of possible transition among states.
3. Transition probabilities from state m to state j calculation.
4. Stationary probabilities distribution (π_m) calculation of each state in steady state.

The Markov chain model with the states of the adaptive system is shown in figure 3.1.

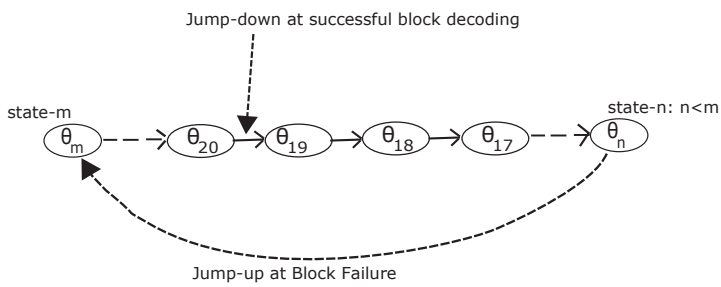


Figure 3.1: Markov Chain Model of PCAA

Let P_n be the probability of frame error in state n. It is also the probability of upward transition to state $n + K$ and the transition probability from state n

to $(n-1)$ is $(1-P_n)$. To compute the block error rate in state n we need the expression of BER in state n for modulation scheme in study. However, in this system BER is influenced by the received SNR as well as θ of state n. These dependencies requires deriving an expression for BER which are hard to compute and are beyond the scope of this project. Having these model terminologies in hand, we will look into an intuitive explanation of the PCAA behaviour in the following section

Intuitive Explanation

Consider X to be the random variable that represents θ_{error} and $X_{\theta_{error}}$ to be the value of θ at which decoding failure occurred. LDPC decoding process begins at predefined initial state θ_{init}. Setting θ_{init} to a very high value will result in high complexity and long convergence time and to a very low value can result into severe performance degradation before PCAA determines optimum θ. At incorrect block decoding, the system experiences an upward jump of $(K\Delta\theta_d)$ from its present state. $K = (1/BLER_t - 1)$ indicates that on average there will be one error out of $(K+1)$ blocks in steady state. Let $\mathbb{E}[X] = \theta_{critical}$ denote the expected value of θ_{error}. With these definitions two approximations are made for the analysis of PCAA

1. There will be no decoding failure and hence no upward jump will happen above $\theta_{critical}$.

2. The probability of reaching a state after $\theta_{critical}$ is almost zero.

The basic intuition behind the approximation is that the system operation is based on jump-up behaviour i.e., on $X_{\theta_{error}}$ and it is mostly restricted to $(\mathbb{E}(X) + K\Delta\theta_d, \mathbb{E}(X))$. If a block failure is considered to be only happening at $\theta_{critical}$, then the upward jump of $(K\Delta\theta_d)$ value at $\theta_{critical}$ state ensures that the system operates at $BLER_t$. One may argue that if $\theta_{critical}$ achieves a fixed value in a system then upon achieving steady state the optimum solution is to fix the θ value slightly above the $\theta_{critical}$ to avoid the decoding failure. In that case, the system will enter into recurrent state in the Markov chain and will stay in that state after entering it. However, it is observed from simulations that due to the random nature of θ_{error}, $\theta_{critical}$ do not achieve a constant value and wanders around $\mathbb{E}[\theta_{critical}]$ in steady state. Fixing the system at $\theta_{critical}$ state may lead to performance loss. Furthermore, as we will see in the simulation results section, $\theta_{critical}$ varies with SNR and fixing the system state can significantly degrade the system performance in presence of fading. Equation (3.4) is the update equation of $\theta_{critical}$ at each decoding failure.

$$\mathbb{E}[X] = \mathbb{E}[X_{\theta_{error}}(old) + X_{\theta_{error}}(new)], \tag{3.4}$$

where $X_{\theta_{error}}(old)$ in equation (3.4) is the set of all the past θ_{error} values since the decoding begins and $X_{\theta_{error}}(new)$ is the most recent value at which decoding failure occurred.

Numerical Example

In this example the PCAA behaviour for $\Delta\theta_d = [1 \quad 0.1 \quad 0.01]$ is explained for a $BLER_t = 10^{-2}$. It is also assumed that a block error only occurred when $\theta \leq \theta_{critical}$ and $\theta_{init} = 30$ for the sake of the algorithm explanation only. Plugging in these values in equation (3.3), we obtain $\Delta\theta_{up} = [99 \quad 9.9 \quad .99]$ for the corresponding $\Delta\theta_d$. The behaviour of the algorithm for this example is shown in figure 3.2, where it can be observed that the algorithm was able to meet $BLER_t$ for all cases of $\Delta\theta_d$.

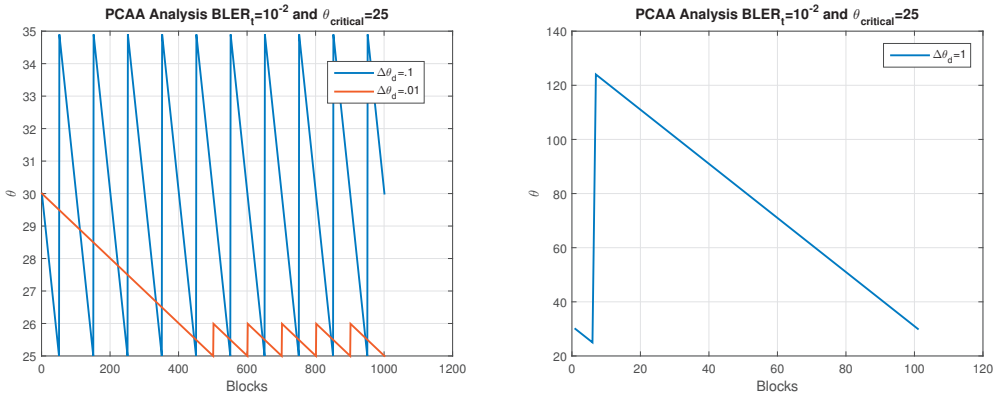


Figure 3.2: PCAA for $\Delta\theta_d = [.1 \quad .01]$ (left) and for $\Delta\theta_d = [1]$ (right)

It can also be observed that with the lower $\Delta\theta_d$, the system operates very close to $\theta_{critical}$. In other words, the variance of $\theta_{critical}$ distribution varies according to the $\Delta\theta_d$. This observation has an important significance to the complexity versus convergence problem of LDPC decoder, which we are aiming to solve using PCAA. The closer the system operates to $\theta_{critical}$, the higher the complexity savings of LDPC decoder will be. Furthermore, steady state for $\Delta\theta_d = [1 \quad .1 \quad .01]$ is achieved at block number 6, 51 and 501 respectively. It can be inferred that for higher $\Delta\theta_d$, the system will enter into the steady state quickly but it will not operate close to the $\theta_{critical}$ state.

Chapter 4

Theoretical Background OFDM

4.1 OFDM Modulation Introduction

Orthogonal Frequency Division Multiplexing (OFDM) is a digital multi-carrier modulation format specially suited to achieve a high-data-rate transmission. The data stream is divided into a number of several low-data-rate subcarriers N. In this fashion, the serial block of N data symbols is transformed into N parallel blocks. Finally, each block is modulated using different orthogonal frequency subcarriers, creating an OFDM symbol.

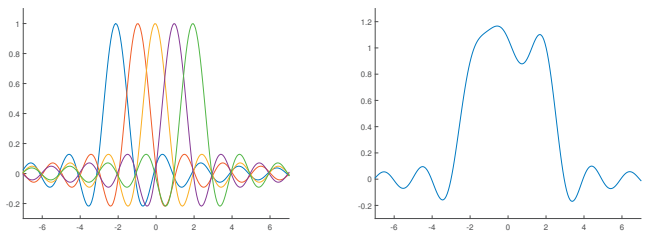


Figure 4.1: Frequency domain individual subcarriers (left) and summation (right)

4.2 OFDM DFT/IDFT

Due to the structure of the OFDM system, where the subcarrier spacing is Δf, i.e. $1/T_s$, the OFDM symbol modulation relies on a mathematical tool, the *Fast Fourier Transform* (FFT). In the digital approach the *Inverse Fast Fourier Transform* (IFFT) is applied to each data symbol block before transmission. Consider the transmit signal in the n_{th} subcarrier, time instant=0, and sampling at $t_k = kT_s/N$

$$s_k = s(t_k) = \frac{1}{\sqrt{T_s}} \sum_{n=0}^{N-1} c_{n,0} \exp\left(j2\pi n \frac{k}{N}\right). \tag{4.1}$$

Equation (4.1) corresponds to the inverse Discrete Fourier Transform (DFT). In most cases N is a power of 2 and the transmitter implements the DFT as an

Inverse Fast Fourier Transform (IFFT) with an N elements output. The nature of this output is in the time domain, and so a conversion parallel to serial is needed to send the signal. The receiver follows a procedure to reverse the steps made in the transmitter. In this manner the signal sampled is packed into a block of N samples, which are later converted from serial to parallel, ready to be subjected by an FFT.

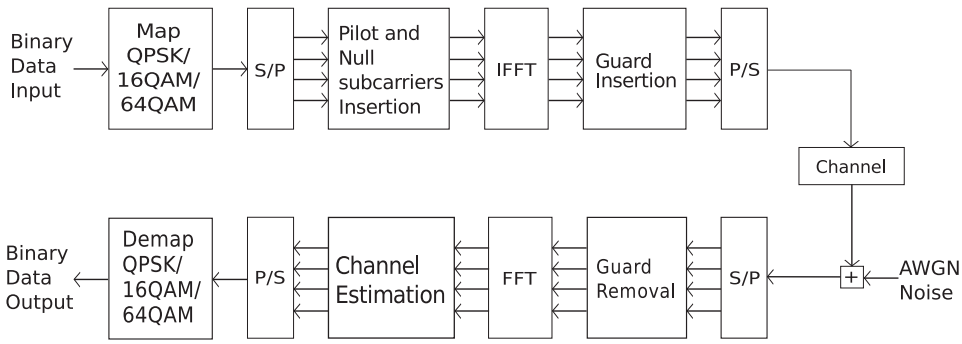


Figure 4.2: OFDM Block Diagram

4.3 OFDM main limitations

There are many advantages of using OFDM, some have been pointed out in the previous section. However, OFDM systems have limitations as well and it is important to mention some of them.

OFDM system relies on the orthogonality of subcarriers and in essentially every wireless communication system, the possibility of a frequency offset is palpable. Local oscillators are not one hundred percent accurate and small errors result from time to time; even with state-of-the-art oscillators, frequency offset still appears. These errors shift carrier frequency, thus risking to eliminate the orthogonality of the system and causing ICI.

One more limitation is the high peak-to-average-power ratio (PAPR). When using the IFFT operation for all subcarriers, a large peak value in the time domain can happen. The power of the OFDM symbol, having N independently modulated subcarriers, is N multiplied by the average power of each of the subcarriers. Therefore, the PAPR is larger in comparison to that in single carrier systems. This is problematic mainly in the uplink, where battery efficiency is important. To reduce the value, clipping is a solution often use.

4.4 Cyclic Prefix

OFDM has been used since the 1960s, followed by the addition of the Cyclic Prefix (CP) few years later. The scheme is still widely used because it is robust against multipath, frequency selective fading and narrowband interference, as they will only affect a certain percentage of the subcarriers.

The inherent orthogonality of the modulation waveforms makes it possible for subcarriers to overlap without interfering with each other, achieving a spectral efficiency higher than other multicarrier communication schemes. This gives place to high-data-rate transmission due to the fact that the channel time delay is probably smaller than the OFDM symbol period.

Moreover, OFDM uses CP as a way of a guard interval to fight multipath, intersymbol interference (ISI) and inter channel interference (ICI), as the CP maintains the orthogonality in the subcarriers. The CP is a copy of the end of the symbol after the IFFT (with a certain period T_{FFT}) added into the guard interval (with duration of T_{GI}). By doing this the linear convolution with the channel impulse response is transformed into a cyclic convolution and the total duration of transmitted OFDM symbol is then T_S.

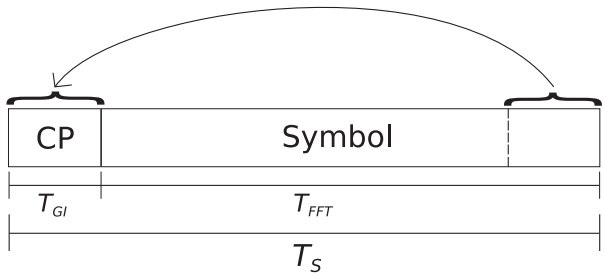


Figure 4.3: CP Representation

4.5 OFDM Simulation

To start the simulation, a random number of bits is encoded and then mapped to a specific modulation. The sequence of bits might not match an integer number of mapped symbols, therefore, bit padding is added. For the whole signal to maintain $\mathbb{E}[P] = 1$ (after scaling it with the normalization factor K_{MOD}) the added bits have a uniform distribution of 0 and 1. For any M-ary QAM, M can be written as $M{=}L^2$ for some L being power of two; and the real and imaginary parts α_I and α_Q of the symbol can be selected from the real L-ary PAM alphabet ($\pm K_{MOD}, \pm 3K_{MOD}, ... \pm (L-1)K_{MOD}$). The n-th element of the alphabet is K_{MOD}(-L+1+2n) for n $\in 0, ..., L-1$. $\mathbb{E}[P]$ can then be calculated as:

$$\begin{aligned}
\mathbb{E}[P] &= \mathbb{E}[|\alpha|^2] \\
&= \mathbb{E}[\alpha_I^2] + E[\alpha_Q^2] \\
&= 2\mathbb{E}[\alpha_I^2] \\
&= 2\sum_{m=0}^{L-1} K_{MOD}^2(-L+1+2m)^2 \\
&= \frac{2}{3}K_{MOD}^2(L^2-1) \\
&= \frac{2}{3}K_{MOD}^2(M-1).
\end{aligned} \tag{4.2}$$

Another key parameter is the SNR. In the simulations performed, the signal SNR before the channel is calculated using equation (4.3)

$$SNR = E_b/N_0 + 10 \cdot \log_{10}\left(\frac{N_{data}}{N_{SC}}\right) + 10 \cdot \log_{10}(N_{BS}) + 10 \cdot \log_{10}(R), \quad (4.3)$$

where N_{data} is the number of data subcarriers, N_{SC} is the number of OFDM total subcarriers, N_{BS} equals the number of bits per mapped symbol, and R is the coding rate.

Finally, to define the performance at the receiver side, the concept Bit Error Rate (BER) must be presented. The bit error rate is the number of errors at the receiver divided by the number of bits transmitted. It is a dimensionless ratio, which for an infinite amount of transmitted bits, becomes a probability. The bit error probability is often described in the literature as BER [16]. The performance of OFDM system was compared against the theoretical performance, where the simulated and the theoretical BER against SNR for modulation schemes QPSK, 16-QAM and 64-QAM over AWGN channel matched. After that the LDPC encoder and decoder was implemented on this OFDM system.

Chapter 5

Simulation Parameters and Methodology

This chapter covers an introduction of the IEEE 802.11n standard, the channel models, parameters, and methodology selected in this project. The aforementioned standard is an amendment published after 802.11a and 802.11g. Part 11 of the standard focuses on two layers of Wireless LAN: the Medium Access Control and the Physical Layer (PHY) specifications.

The parameters used were chosen accordingly to the PHY specifications. The standard supports, naturally, the OFDM technique and the application of LDPC code. The amendment gives the opportunity to achieve higher throughputs and transmission rate compared to previous versions. The maximum data rate being 600 Mbit/s [13]. The frequency bands assigned are 2.4GHz and 5GHz with bandwidths of 20MHz and 40MHz. The standard allows former amendments like 802.11g, 802.11b and 802.11a to be set in the transmission frame, thus assuring coexistence among them.

The IEEE 802.11n PHY make use of 64 OFDM subcarriers, 48 of them are used to carry data and 4 are pilot subcarriers. Pilot subcarriers contain signals to trace the frequency offset and phase noise, and there are spread across the 64 subcarriers. Based on the data rate, modulation and coding rates are selected. Table 5.1 shows the main characteristics of the standard PHY layer.

Table 5.1: IEEE 802.11n Main Parameters

Parameter Name	**Parameter Value**
Carrier Frequency	2.4GHz, 5GHz
Modulation	BPSK, QPSK, 16QAM, 64QAM
Coding Rate (20MHz)	1/2, 2/3, 3/4, 5/6
Bandwidth	20MHz, 40MHz
Number of data subcarriers	48
Number of pilot subcarriers	4
OFDM Symbol Duration T_S (20MHz)	4.0 μs (T_{GI}+T_{FFT})
Guard Duration T_{GI} (20MHz)	0.8μs ($T_{FFT}/4$)
FFT Duration T_{FFT} (20MHz)	3.2μ s ($1/\Delta_F$)
Coding Technique	Convolutional Code, LDPC Code

5.1 Definition of Parameters

A High Throughput (HT) OFDM system is simulated with a 20 MHz channel width and a carrier frequency of 2.4 GHz. The channel is divided into 64 subcarriers, where 48 of them carry data.

Following the standard, three different MCSs were chosen, all of them have 1 spatial stream, as it is a Single Input Single Output (SISO) approach.

Table 5.2: Chosen modulation parameters

MCS	Modulation	N_{BPSCS}	R	N
2	QPSK	2	3/4	1296
3	16QAM	4	1/2	1296
5	64QAM	6	2/3	648

In table 5.2, R represents the coding rate, N refers to the codeword length for the LDPC decoder, and N_{BPSCS} is the number of coded bits per single carrier. Although the standard mentions that each OFDM subcarriers can have a different constellation size, in the project the same constellation size is kept during each simulation. The mapping constellations are implemented using Gray rectangular coding.

Gray code is a manner of coding bits in a binary system in such a way that two neighbouring constellation points have only one different bit. Due to the nature of the coding, the maximum error found in Gray-encoded data is much less than the one occurring using weight binary encoding. Figure 5.1 shows the constellation for each modulation mapping applied in the simulations. The data is later multiplied by a normalization factor K_{MOD} specified in table 5.3 to produce an $\mathbb{E}[P] = 1$ in every mapping (see equation (4.2)).

Table 5.3: Normalization factor for different modulations

Modulation	K_{MOD}
BPSK	1
QPSK	$1/\sqrt{2}$
16-QAM	$1/\sqrt{10}$
64-QAM	$1/\sqrt{42}$

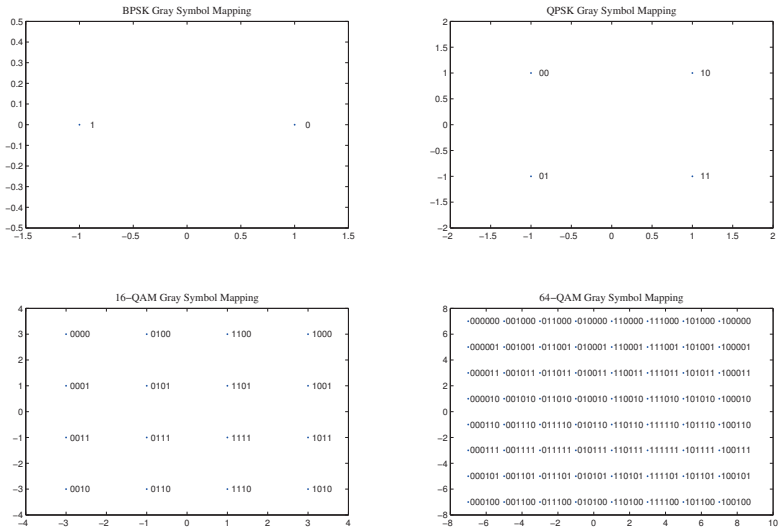


Figure 5.1: Constellation Mapping

5.2 Channel Models

The effects of a propagation channel are an attenuation of the signal, delay and frequency dispersion, in conjunction with addition of AWGN and co-channel interference.

5.2.1 AWGN Channel

The first stage of this work was implemented using an AWGN Channel. The stated channel mitigates the transmitted signal and incorporates Gaussian distributed noise, which accurately models the main sort of noise. The characteristic of being "white" comes from light properties due to white light having same quantity of all frequencies contained in the visible band of the electromagnetic spectrum. Moreover, the WGN properties state that two given samples are uncorrelated, in spite of being near each other in the time domain.

Noise is present in every wireless communication system. The main sorts being thermal noise, electrical noise from active electronics, and inter-cellular interference. The impact is seen in a degradation of the SNR, degrading the spectral efficiency of the communication system.

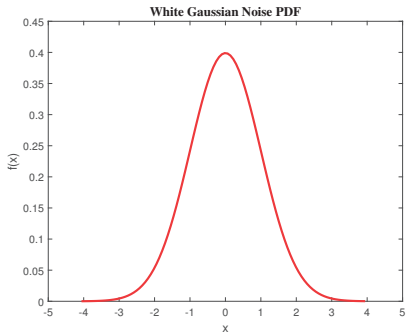


Figure 5.2: White Gaussian Noise probability density function (pdf)

5.2.2 Rayleigh Flat Fading, Time Varying Channel with AWGN

The second stage of the project was to simulate a time varying channel. This approach takes into account the natural environment of a wireless system, where multipath appears due to the signal propagation through different paths. This causes the signal to bounce off several surfaces before being received.

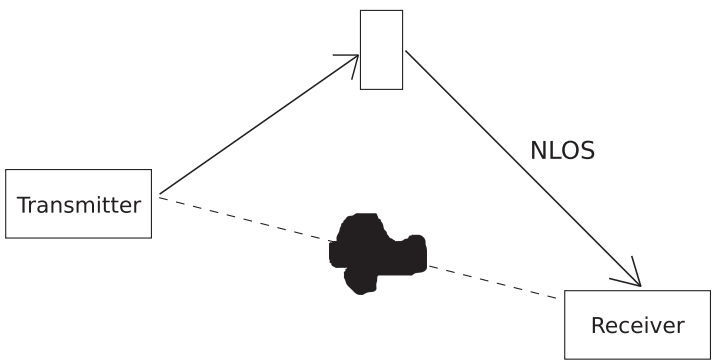


Figure 5.3: Non Line-of-Sight (NLOS) Representation

The behaviour of travelling paths changes unpredictably, however, it can be described using statistical models. When there exists a large amount of reflected paths, and these comes from a NLOS environment, the central limit theorem is properly used to state that the channel has statistical characteristics of a Rayleigh distribution.

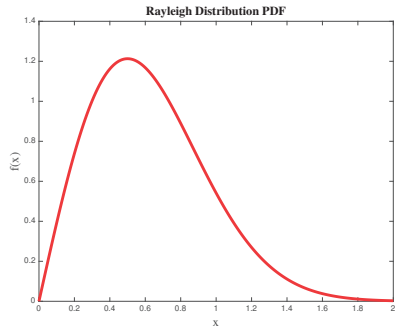


Figure 5.4: Rayleigh pdf

Besides paths reflection, Doppler shift is also considered in the second stage. Due to this, the Rayleigh-distributed channel gain fluctuates with time. Doppler shift is the name of the effect in the signal when the transmitter is moving in relation to the receiver, the relative speed between them is directly proportional to the Doppler shift. In the same manner, Doppler shift is proportional to Doppler spread (D_s), and $T_C \approx 1/D_s$, where T_C represents the coherence time (time domain dual of D_s).

D_s and T_C help describe the time varying nature of the channel, how fast it varies. For a slow fading: $T_C > T_S$, where T_S represents the symbol period [17].

The type of fading selected for the second part is flat fading. Based on the comparison between multipath RMS delay spread σ_τ and the OFDM symbol length, either flat fading or frequency selective fading can take place. This effect appears when the transmitted signal is affected in the whole spectrum by the same path gains and phase shifts. Hence, the following stands for flat fading, where B_C is the coherence bandwidth and B represents the bandwidth of signal:

Frequency Domain	*Time Domain*
$B_C > B$	$T_S > \sigma_\tau$

Implementation Note

SNR of the fading signal continuously varies due to slowly varying nature of fading channel. In simulation, blocks that experiences deep fading dips result in $\mathbb{E}[SNR_{faded}]$ much lower than $\mathbb{E}[SNR_{nonfaded}]$. Blocks with $\mathbb{E}[SNR_{faded}]$ lower than $SNR_{operating}$ are discarded from the analysis for Rayleigh flat fading case.

5.2.3 Rayleigh Frequency Selective Channel with AWGN

In the third and final stage, performance of the the system is evaluated on frequency selective channel in an indoor environment.

Compared to second stage of the project, in this stage along with amplitude fluctuations in time domain, channel will be frequency selective as well. Frequency selective fading occurs when

- BW of signal > BW of channel
- Delay spread > Symbol period

Frequency selective channel Model

Tapped delay line (TDL) is a popular model for discrete multipath frequency selective channel. The model uses multiple number of frequency flat fading generators corresponding to Multi path Components (MPC), where MPC are independent from each other with the expected power of one per MPC. In TDL model, low-pass impulse response of the channel is modelled as [18]:

$$\widetilde{c}(\tau(t),t) = \sum_{k=1}^{K(t)} \widetilde{a_k}(\tau_k(t),t)\delta(\tau - \tau_k(t)). \tag{5.1}$$

In equation (5.1), $K(t)$ is the time varying number of MPC, $\widetilde{a_k}(\tau_k(t),t)$ are the low pass time-varying complex channel coefficients that includes both amplitude and phase effects, $\tau_k(t)$ are the time-varying time delays, τ represents delay axis and $\tau(t)$ in $\widetilde{c}(\tau(t),t)$ represents that resultant MPCs are different at any observation time t.

For IEEE802.11n given channel models [13], the PDP delays τ_k are not an integer multiple of channel sampling period T_s of the system. Before implementing the TDL FIR filter for discrete time channel simulator, given channel taps must be adjusted to the integer multiple of T_s by oversampling. This process will make the number of taps too large for the equivalent FIR filter for TDL model. Band limited discrete multipath channel model with tap adjustment by FIR filter implementation is briefly discussed in [18].

In this project, indoor exponential model of 2.4GHz indoor channel, used by the IEEE802.11b task group [19], is used in simulation due to its implementation simplicity. The major advantage of this channel model is its ability to produce channel taps at integer multiple of T_s. Each channel tap is modelled by an independent complex Gaussian random variable with its average power that follows exponential PDP [20].

First step in exponential model is to determine the maximum number of paths given RMS delay spread σ_τ and channel sampling period T_s by using

$$k_{max} = \lceil 10\sigma_\tau / T_s \rceil. \tag{5.2}$$

The power of the k^{th} tap is zero mean and $\sigma_k^2/2$ distributed. k^{th} channel tap samples are generated by using equation (5.3)

$$h_k = Z_1 + jZ_2, k = 0, ..., k_{max}. \tag{5.3}$$

The power of each channel tap is calculated in terms of first tap power by using

$$\sigma_k^2 = \sigma_0^2 e^{-kT_s/\sigma_\tau}, \tag{5.4}$$

σ_0^2 is the power of first tap which is calculated in a way to make the average received power of 1, which is as follows

$$\sigma_0^2 = \frac{1 - e^{-T_s/\sigma_\tau}}{1 - e^{-k_{max+1}T_s/\sigma_\tau}}. \tag{5.5}$$

Implementation Note

For simulation, the frequency response of the channel is constant for all OFDM symbols in one block, however, frequency response changes from block to block. Block length determines the number of samples for a channel which are created using equation 5.3. After transmission through this channel the received data symbols on the i-th sub-carrier of the j-th OFDM symbol R_{ij} can be expressed as

$$R_{ij} = S_{ij}H_{ij} + n_{ij}. \tag{5.6}$$

In equation (5.6), S_{ij} is the transmitted symbol, H_{ij} is channel fading in frequency domain and n_{ij} is the AWGN noise where $\mathbb{E}[n_{ij}] = \sigma^2$, SNR_{ij} is obtained by using equation (5.7).

$$SNR_{ij} = \frac{|S_{ij}H_{ij}|^2}{\sigma^2}. \tag{5.7}$$

Since equalization is performed in frequency domain before LDPC decoding, it will change the noise per subcarrier. In order to accommodate the equalization, the modified noise variance , $\sigma^2/|H_{ij}|^2$, for each individual subcarrier in OFDM symbol is used to calculate the channel LLRs for LDPC decoder.

Chapter 6

Simulation and Results

This chapter provides detailed analysis on the BLER, Mean Complexity and Convergence performance of simulations performed for different configurations of MCSs, $\Delta\theta_d$, and $BLER_t$ as described in tables 6.1, 6.2, and 6.3. OFDM system implementation and results analysis were carried out in MATLAB© while LDPC decoder was implemented in C to take advantage of its faster computations compared to MATLAB©. The simulations are performed over AWGN, time varying and frequency selective channels and performance results of each channel configuration is discussed in subsequent sections.

6.1 Simulation Results AWGN Channel

The following parameters were defined for this first stage:

Table 6.1: Parameters for AWGN Channel

Stage	**MCS**	$\Delta\theta_d$	$BLER_t$	θ_{init}
1	2,3,5	.1,.01,.001	.1,.01,.001	20

6.1.1 BLER vs E_b/N_0 performance

- BLER vs E_b/N_0 simulation performed on $BLER_t = 10^{-2}$ and $\theta_{init} = 20$.
- For Lower $\Delta\theta_d$ lower BLER was obtained while keeping $BLER_t$ fixed.
- BLER for $\Delta\theta_d = .1$ and $\Delta\theta_d = .01$ meets $BLER_t = 10^{-2}$. However, for $\Delta\theta_d = .001$ BLER was slightly below the $BLER_t$.

This first section of results shows that desired $BLER_t$ is achieved for the parameters specified in table 6.1, which validates the expected behaviour of PCAA mentioned in section 3.2.2. The performance is below or equal to the $BLER_t$ when $\Delta\theta_d$ is selected at or above $BLER_t$ for all MCSs.

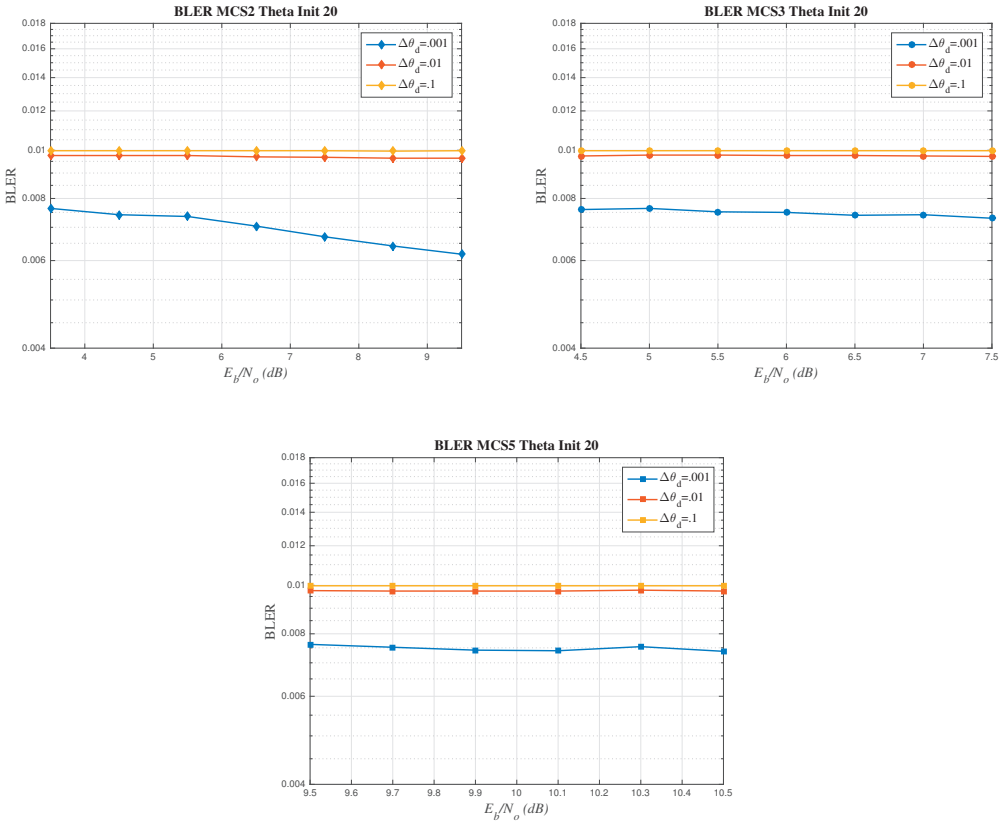


Figure 6.1: MCS2, MCS3 and MCS5 BLER vs E_b/N_0 for $\Delta\theta_d = 0.001$, $\Delta\theta_d = 0.01$ and $\Delta\theta_d = 0.1$

6.1.2 Convergence at extreme SNRs

Convergence is achieved once the system operates in $\mathbb{E}[\theta_{critical}]$ region. In order to study the convergence, simulations are performed over 100 realizations for 4000 blocks each per test configuration. Mean complexity is calculated by taking ensemble average across all realizations.

- MCS2 $E_b/N_0 = 3.5$dB and 9.5dB for $\Delta\theta_d = .1$ converges within 5-10 ms and 10-15 ms, for $\Delta\theta_d = .01$ MCS2 converges within 60-70 ms and 100 ms respectively for stated SNRs. For $\Delta\theta_d = .001$ it doesn't converge in 200 ms.
- MCS3, on both $E_b/N_0 = 4.5$dB and $E_b/N_0 = 7.5$dB, for $\Delta\theta_d = .1$ converges within 2-5 ms, for $\Delta\theta_d = .01$ it converges within 35-45 ms and for $\Delta\theta_d = .001$ it doesn't converge in 100 ms.
- MCS5 $E_b/N_0 = 9.5$dB for $\Delta\theta_d = .1$ converges within 2-5 ms, for $\Delta\theta_d = .01$ it converges within 20-23 ms and $\Delta\theta_d = .001$ doesn't converge in 50 ms. The convergence time for $E_b/N_0 = 10.5$dB is same as $E_b/N_0 = 9.5$dB except for $\Delta\theta_d = .01$ in which system converges in 16-19 ms.

Convergence rate is proportional to $\Delta\theta_d$. Time to reach convergence or steady state will be lower at lower SNRs and higher at higher SNRs. Low SNRs have high $\mathbb{E}[\theta_{critical}]$ resulting in small time to achieve steady state. Similarly high SNR has low $\mathbb{E}[\theta_{critical}]$ and therefore it will take more time to reach steady state.

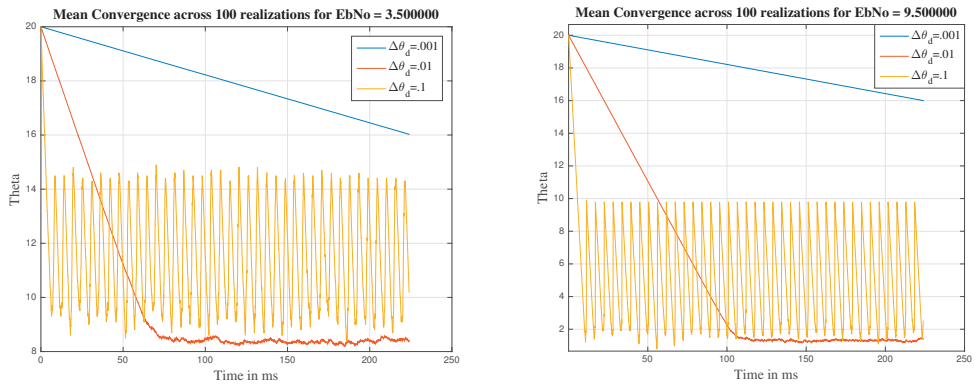


Figure 6.2: MCS2 Convergence at E_b/N_0=3.5dB (left) and E_b/N_0=9.5dB (right)

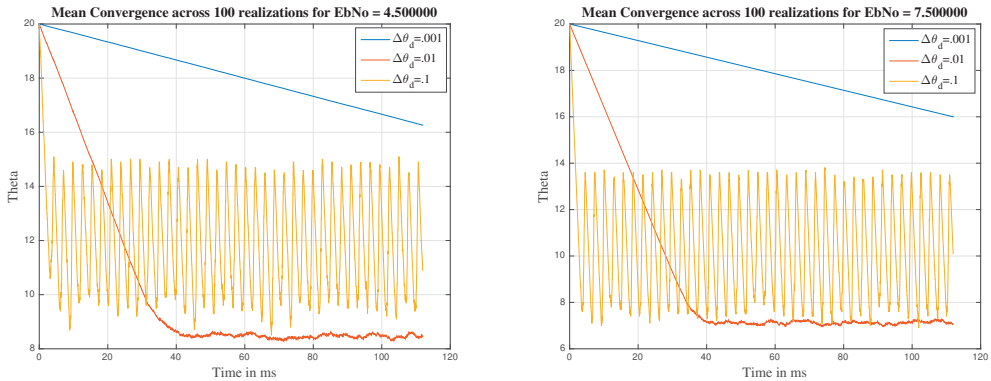


Figure 6.3: MCS3 Convergence at E_b/N_0=4.5dB (left) and E_b/N_0=7.5dB (right)

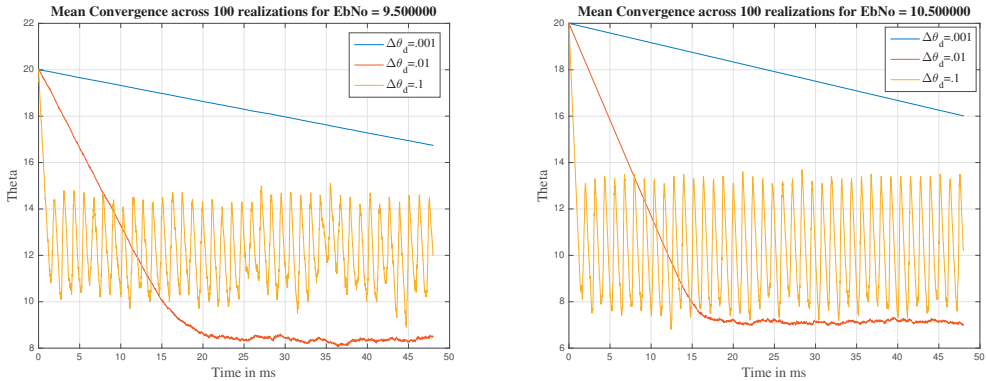


Figure 6.4: MCS5 Convergence at E_b/N_0=9.5dB (left) and E_b/N_0=10.5dB (right)

In MCS3 at $E_b/N_0 = 4.5$dB and in MCS5 at $E_b/N_0 = 9.5$dB, $\mathbb{E}[\theta_{critical}]$ is around 8.5. However, convergence time of MCS3 is approximately 40ms, whereas, for MCS5, it is half as much. Major reason for this observation is differences in codeword length and modulation order of MCS5 and MCS3 that influences the block duration and hence convergence time for similar $\theta_{critical}$. The block duration is proportional to the constellation size, and the codeword length. For higher-order modulation, blocks of length N will have less OFDM symbols compared to the number of OFDM symbols with lower order modulation.

6.1.3 Steady State Mean Complexity

Steady state consists of all the blocks following the block n at which PCAA brings down θ of FC from θ_{init} to $\mathbb{E}[\theta_{critical}]$, region where decoding failure is highly probable.

Mean complexity in steady state for subjected MCSs was obtained at $BLER_t = 10^{-2}$, $\theta_{init} = 20$ and for each of the $\Delta\theta_d = 0.001$, $\Delta\theta_d = 0.01$, $\Delta\theta_d = 0.1$ separately. Results were then plotted for each MCS individually including mean complexity of Non-FC LDPC decoding. For very small values of $\Delta\theta_d$ the system spends more time in transient state leading to high complexity. Transient state duration has significant impact on the overall complexity performance. However, once the system is in steady state, lower $\Delta\theta_d$ makes the system oscillates around $\mathbb{E}[\theta_{critical}]$ with small variance and hence it will results in improved system performance and lesser complexity.

- For all MCSs, mean complexity in steady state reduces with reduction in $\Delta\theta_d$.
- MCS3 has the highest mean complexity followed by MCS5 and MCS2.

Lowest step size results in lowest complexity because in steady state, the system with lowest step $\Delta\theta_d$ operates around $\mathbb{E}[\theta_{critical}]$ with smaller jumps compared to higher values of $\Delta\theta_d$.

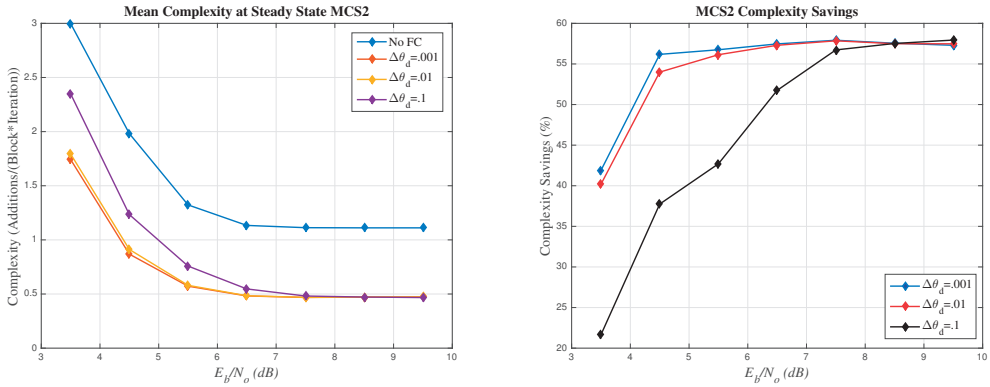


Figure 6.5: MCS2 Complexity Comparison

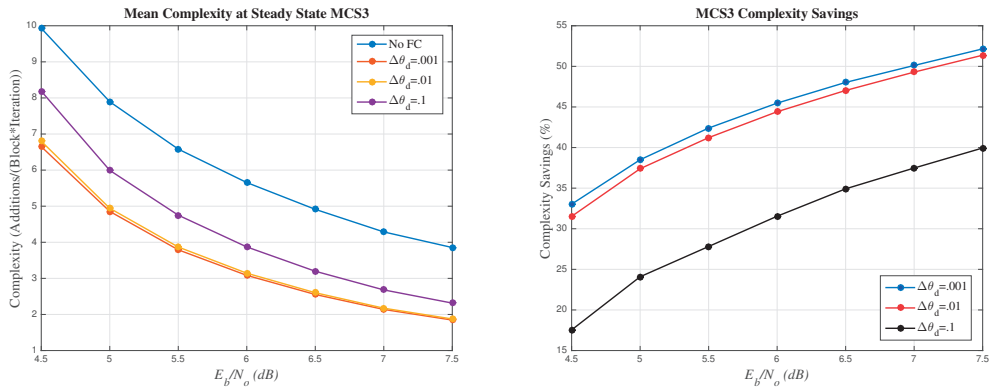


Figure 6.6: MCS3 Complexity Comparison

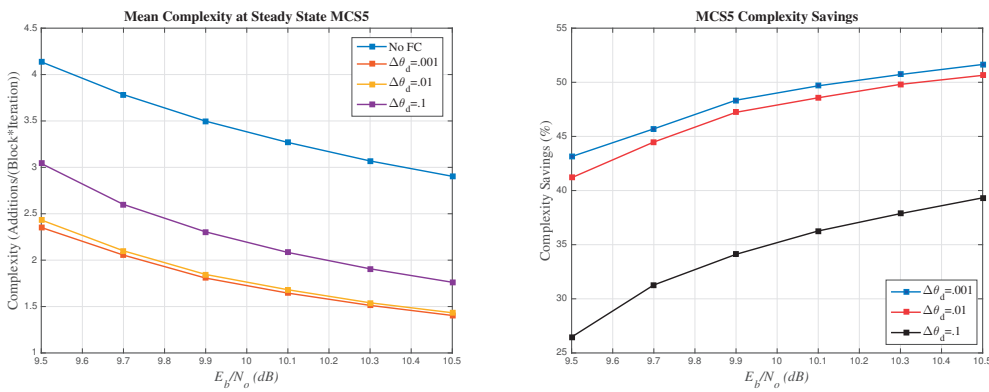


Figure 6.7: MCS5 Complexity Comparison

$\Delta\theta_d$ is a good parameter to measure the tradeoff between convergence rate and complexity reduction. As $\Delta\theta_d$ increases, convergence time decreases and vice versa. However, the reduction in convergence time comes at the cost of increased complexity due to the high upward jump in θ.

6.1.4 $\mathbb{E}[\theta_{critical}]$ Results

It is the expected value of all the θ_{error} values at which LDPC decoding failed and PCAA increases θ to a higher value of $\theta + \Delta\theta_{up}$. Higher value of system $\mathbb{E}[\theta_{critical}]$ indicate that system is operating at higher complexity. It is observed that $\mathbb{E}[\theta_{critical}]$ is high at low SNR and vice versa for all MCSs.

Since each MCS operates at different E_b/N_0 therefore $\mathbb{E}[\theta_{critical}]$ comparison among various MCS is not a suitable choice. For all MCS, as the SNR decreases PCAA increases the θ value to meet the $BLER_t$. Because at lower SNR, with lower θ, the system performance will be worse than the desired $BLER_t$. LDPC decoder can not produce reliable enough decision for low θ in this case. In order to meet the desired performance at low SNR, higher values of θ are required at FC algorithm subjected to $SNR \geq SNR_{operating}$. Because of this reason, LDPC decoder reported high $\theta_{critical}$ for each MCS at low SNR.

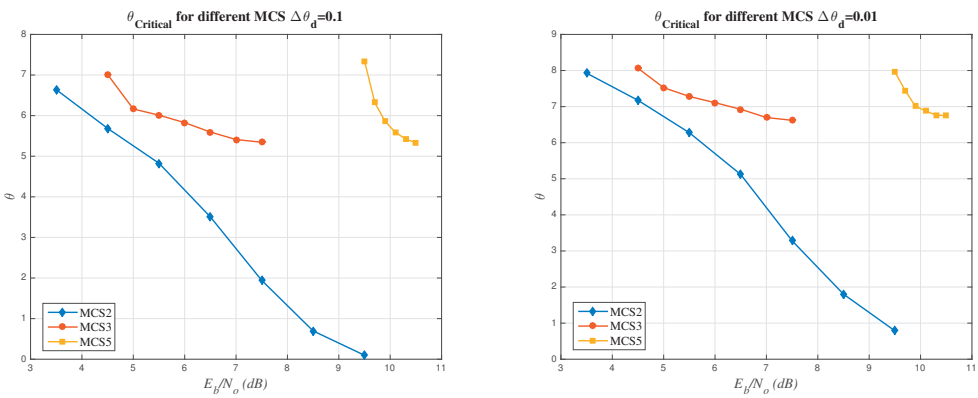


Figure 6.8: $\mathbb{E}[\theta_{critical}]$ at $\Delta\theta_d = .1$ (left) and $\mathbb{E}[\theta_{critical}]$ at $\Delta\theta_d = .01$ (right)

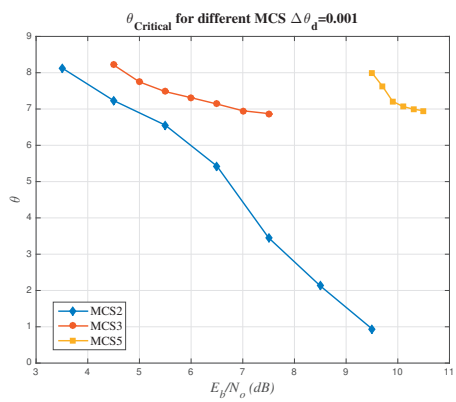


Figure 6.9: $\mathbb{E}[\theta_{critical}]$ at $\Delta\theta_d = .001$

6.1.5 Complexity for all MCSs together

A general conclusion can be made for all MCSs tested: ***Complexity of the system implemented using LDPC decoding with FC, with parameters of FC tuned by PCAA, is lower than the case where no FC is used.*** Also, $\Delta\theta_d = BLER_t$ gives a good trade-off between convergence time and complexity reduction.

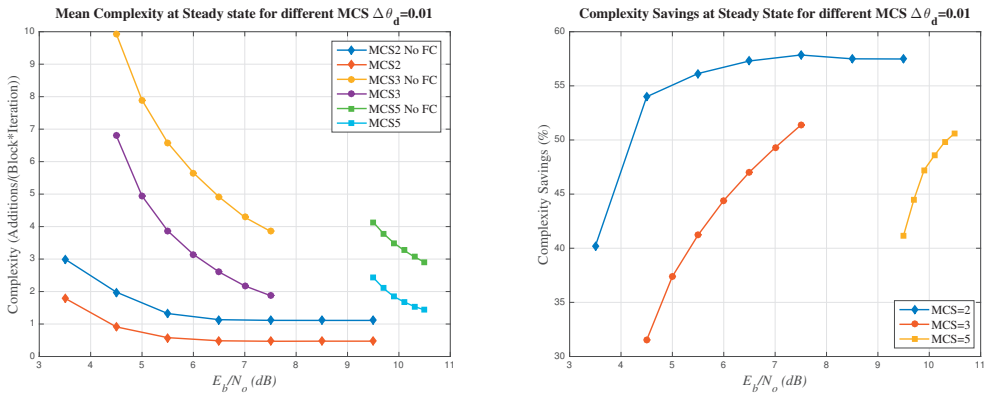


Figure 6.10: MCSs Complexity Comparison

6.1.6 Mean Complexity for different $BLER_t$

In figure 6.11 it is easy to see how complexity of the MCSs increases as $BLER_t$ decreases. This is because it takes higher effort to achieve lower $BLER_t$ (better performance of the system). It is important to state that $\Delta\theta_d$ remains 0.01 for the 9 graphs.

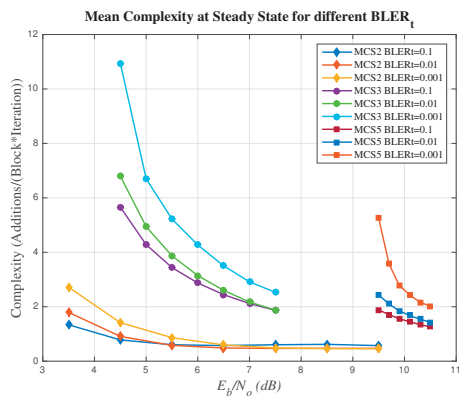


Figure 6.11: Mean Complexity of MCSs for different $BLER_t$

6.2 Results Rayleigh Flat Fading, Time Varying Channel

For this second stage, simulations only take place under $SNR_{operating}$ in order for the system to achieve $BLER_t$. The blocks where the SNR is below $SNR_{operating}$

are not taken into account for the analysis. The PCAA performance is noticeable in the graphs, as θ adapts to the environment.

The following parameters are defined:

Table 6.2: Parameters Rayleigh Flat Fading Time Varying Channel

Stage	**MCS**	$\Delta\theta_d$	$BLER_t$	θ_{init}	**Doppler Shift**
2	2,3	0.01	0.01	20	20

From section 6.1 Convergence time for MCS3, $\Delta\theta_d$=0.01 is between 35∼40 ms.The coherence time T_C can be obtained by:

$$T_C = \frac{9}{16\pi f_d}, \tag{6.1}$$

where f_d is the maximum Doppler shift. For the implemented channel, T_C = 9ms.

From results it can be observed that for $\Delta\theta_d$ = .1 PCAA makes the system converge faster before the channel properties changes. For $\Delta\theta_d$ = .01, .001 system takes longer time to converge than T_C.

6.2.1 BLER vs E_b/N_0 performance

During the second stage, both MCSs chosen are able to stay below the $BLER_t$.

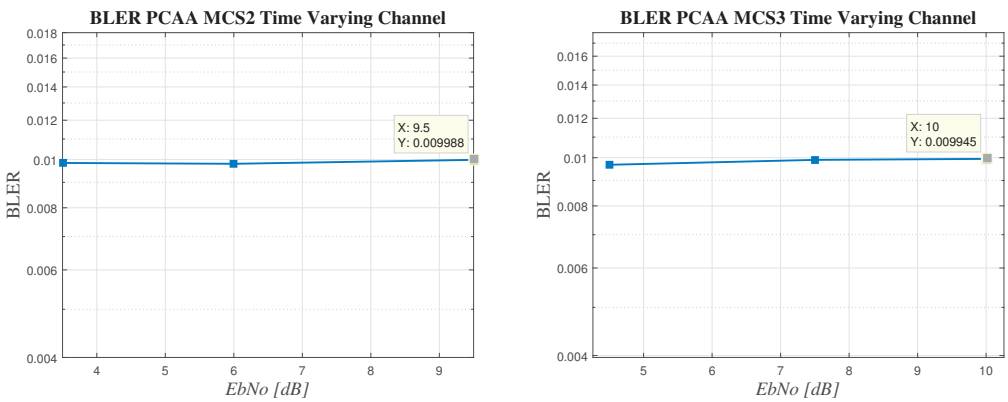


Figure 6.12: MCS2 BLER vs E_b/N_0 Time Varying Channel (left) and MCS3 (right)

6.2.2 Steady State Mean Complexity Time Varying Channel

The same procedure performed in part one is used to calculate mean complexity in a time varying channel. Mean complexity seen in figure 6.13 (left) for the case of MCS2 PCAA FC LDPC decoding shows to be higher at low SNR in comparison with the Non-FC LDPC decoding. This could take place if the channel experiences

deep fading for the first case. However, at higher SNRs, mean complexity drops and complexity savings of above 30% are accomplished at E_b/N_0=9.5. Likewise, MCS3 shows high complexity savings.

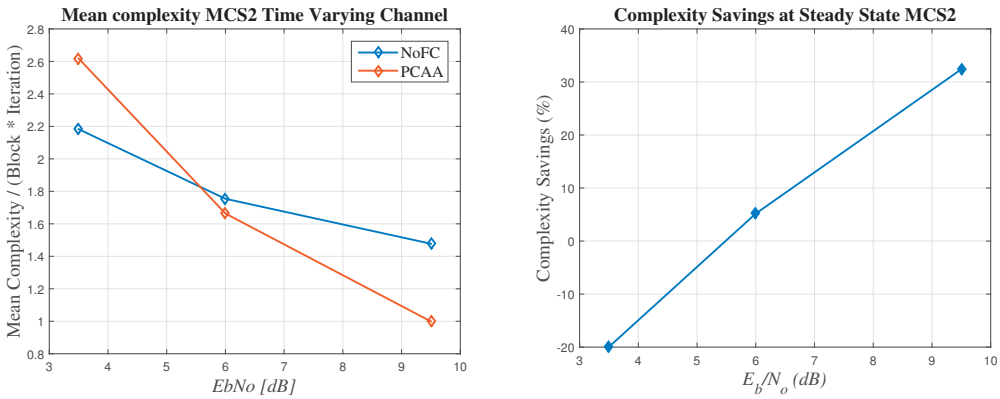


Figure 6.13: Mean Complexity of MCS2 Time Varying Channel (left) and Complexity Savings (right)

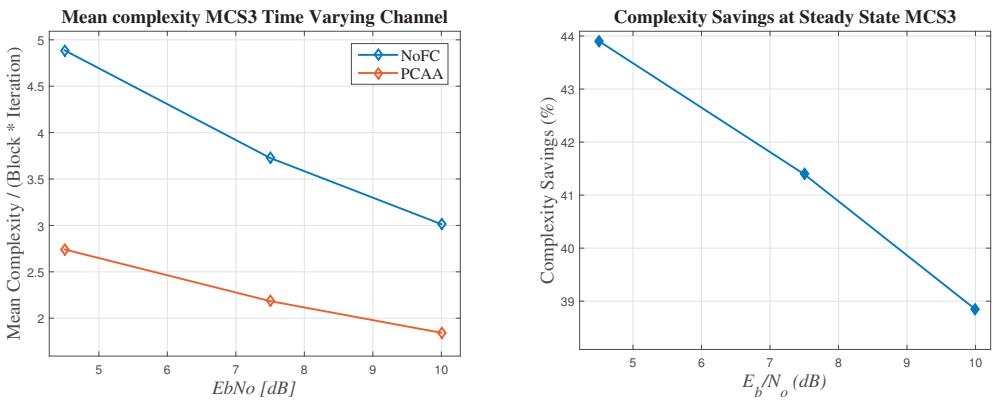


Figure 6.14: Mean Complexity of MCS3 Time Varying Channel (left) and Complexity Savings (right)

6.2.3 PCAA Tracking capability for Flat Fading, Time Varying Channel

An important part of the project is to present the behaviour of θ. In the following plots, it is easy to see how the channel affects the signal and how θ changes as well. When the channel is fading the PCAA increases θ. On the contrary, if the channel does not degrade the system, PCAA decreases θ.

This clearly shows that for given parameter in table 6.2 PCAA is capable of tracking the slowly time varying frequency flat channel.

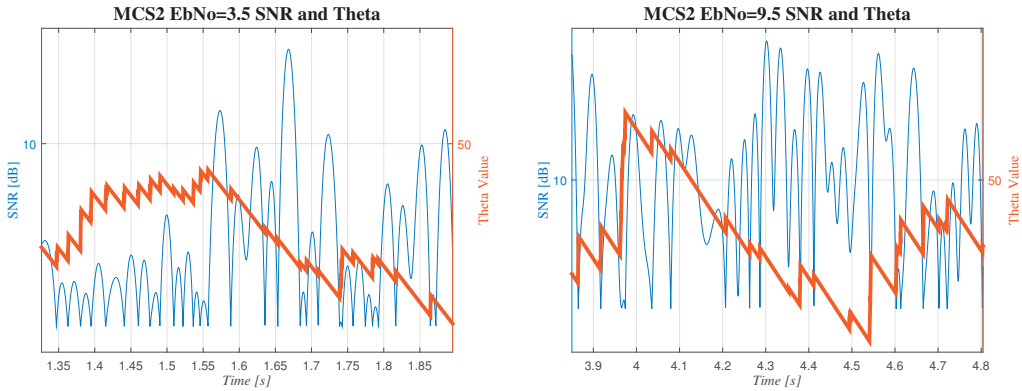


Figure 6.15: MCS2 E_b/N_0=3.5dB Signal SNR vs Theta Value Time Varying Channel (left) and E_b/N_0=9.5dB (right)

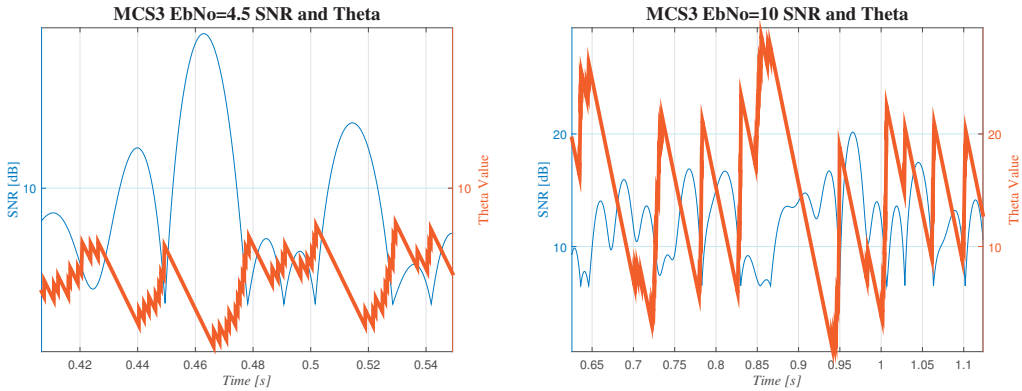


Figure 6.16: MCS3 E_b/N_0=4.5dB Signal SNR vs Theta Value Time Varying Channel (left) and E_b/N_0=10dB (right)

6.3 Results Part 3 Frequency Selective Channel

The main parameters for the last stage are presented below:

Table 6.3: Parameters Frequency Selective Channel

Stage	**MCS**	$\Delta\theta_d$	$BLER_t$	θ_{init}	T_s	σ_τ	**MPC**
3	2	0.01	0.01	20	50ns	25ns	6

During the third stage of the project, only one MCS is chosen, at E_b/N_0=20 dB. It is important to mention that the channel, while being frequency selective, each OFDM sub-carrier experiences flat fading channel with independent Rayleigh fading. Also, from table 5.2 and 6.3, it can be seen that

$$\sigma_\tau < T_{GI}. \tag{6.2}$$

The equation (6.2) ensures that no ISI is experienced. The system is able to deliver $BLER_t$ for this last stage as can be the seen from the simulation result given in table 6.4.

Table 6.4: MCS2 BLER vs E_b/N_0 Frequency Selective Channel

MCS	E_b/N_0	**BLER**
2	20dB	.001

6.3.1 Steady State Mean Complexity Frequency Selective Channel

The same channel for both case (PCAA FC LDPC decoding and Non-FC LDPC decoding) is introduced. For the first case, Mean complexity shows to be less than for the second case, giving a high complexity savings of 66.69%, simulation results are given in table 6.5.

Table 6.5: Mean Complexity (C) of MCS2 Frequency Selective Channel

MCS	E_b/N_0	C_{FC}	C_{NOFC}
2	20dB	.46	1.6

6.3.2 PCAA Tracking capability for Frequency Selective Channel

In the following graphs, the behaviour of θ is presented. Two different observations from figure 6.17 (left), are selected. At the first observation, t=.9587s, the system experiences a decoding failure and at the second observation, t=.9833s, the block was successfully decoded. The next step is to investigate the channel frequency response at these two observation times.

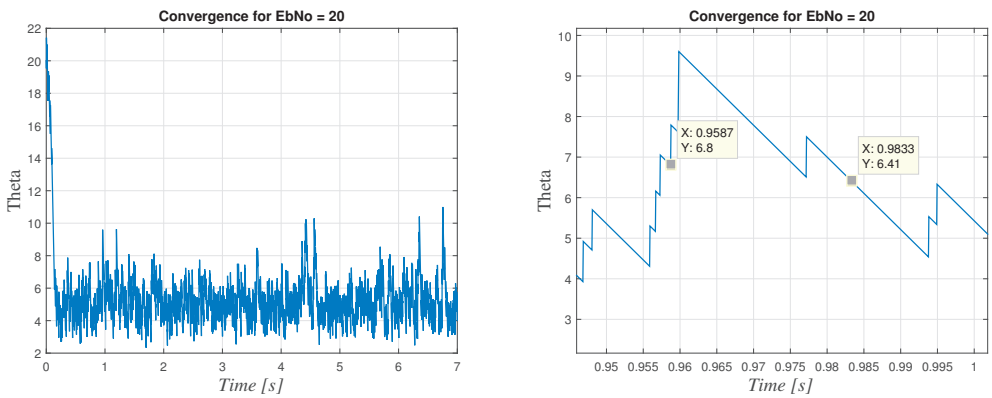


Figure 6.17: Convergence of θ (left) Comparison among θ values (right)

Figure 6.18 shows the two different channel frequency response for the blocks mentioned above. It is easy to see that at the first block, the overall channel power

is much lower than the channel power at the second block. Thus explaining why the PCAA increases θ. On the other hand, when the system experiences a better channel, PCAA keeps decreasing θ. By this manner, the tracking capability of PCAA is demonstrated for a frequency selective channel.

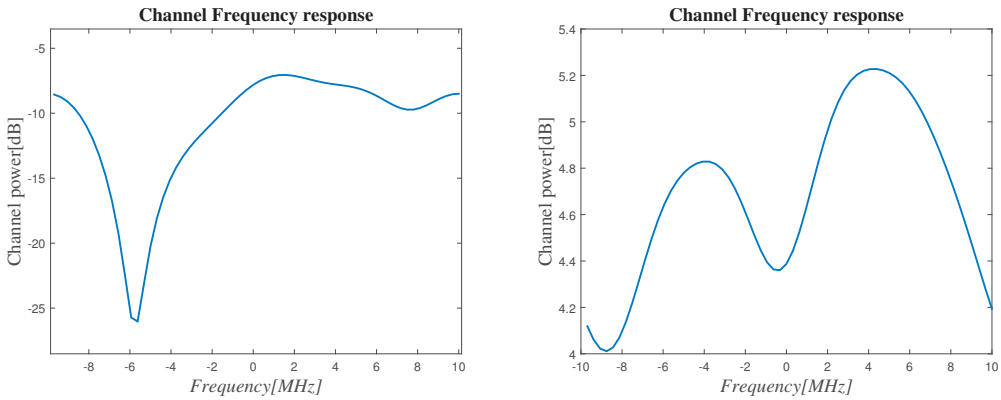


Figure 6.18: Channel Frequency Response Comparison

Chapter 7
Conclusion

In this project, the performance of reduced complexity version of the LDPC decoding is investigated using OMS FC algorithm and PCAA algorithm for dynamically adjusting the FC θ threshold. Convergence and complexity performance metrics for various MCSs has been simulated under IEEE802.11n OFDM communication system. The results obtained from this research project are important in achieving more energy efficient communication system. It is observed that $\Delta\theta_d = BLER_t$ yields best performance in terms of both complexity savings and convergence time.

Firstly, simulations are performed on AWGN channel. A large number of LDPC decoding parameters combinations were tested using three different MCSs, $\Delta\theta_d$'s and $BLER_t$ in order to determine the most optimum configuration. In all test configurations system meets the desired $BLER_t$ indicating that PCAA adjust θ of LDPC FC decoding effectively. It is found that as $\Delta\theta_d$ increases convergence time decreases but it leads to increased complexity. Moreover, the constellation size and the codeword length play important roles in improving the convergence time of system given that $\mathbb{E}[\theta_{critical}]$ is same among different configurations.

Secondly, system simulations performed on Rayleigh flat fading slow time varying channel with optimum configuration found from first stage of the project. Even though the channel coherence time was smaller than the convergence time of chosen configuration, PCAA was able to track the channel changes with complexity savings relative to non-forced convergence LDPC decoding.

Lastly, the system undergoes a frequency selective channel. The result section shows how PCAA still delivers good tracking capabilities and significant complexity savings of 66.69% relative to LDPC Non-FC decoding case.

7.1 Future Work

This work uses irregular LDPC codes for encoding/decoding purposes and next stage would be to research the behaviour of how regular LDPC codes influences the system performance. Current work uses limited test configurations due to long simulation time, and future simulations can be run on wider range of decoding parameters to validate the results developed in current thesis project.

Performance of frequency selective fading channel implementation by TDL model and Doppler filters is another area to be explored. Additionally impact of

frequency selective channel with RMS delay spread greater than CP duration on system performance is another open area for investigation.

Markov model of a PCAA algorithm in section 3.2.2 requires determining exact analytical expression of $BLER(\theta)$ function. Determining $BLER(\theta)$ and $C(\theta)$ analytic expression in estimating the system state and performance of system as function of time will lead to more efficient system design.

References

[1] M. Sarajlic, L. Liu and O. Edfors, "Modified forced convergence decoding of LDPC codes with optimized decoder parameters," *Personal, Indoor, and Mobile Radio Communications (PIMRC), 2015 IEEE 26th Annual International Symposium on*, Hong Kong, 2015, pp. 440-445.

[2] R. G. Gallager, *Low-Density Parity-Check Codes*. Cambridge, MA: MIT Press, 1963.

[3] D. J. C. MacKay and R. M. Neal, "Near Shannon limit performance of low density parity-check codes," in *Electronics Letters*, vol. 33, no. 6, pp. 457-458, 13 Mar 1997.

[4] R. Tanner, "A recursive approach to low complexity codes," in *IEEE Transactions on Information Theory*, vol. 27, no. 5, pp. 533-547, Sep 1981.

[5] S. J. Johnson, Introducing Low-Density Parity-Check Codes. [Online]. Available: http://sigpromu.org/sarah/SJohnsonLDPCintro.pdf

[6] M. Lentmaier, "Error Control Coding" Lecture Notes. EIT, Lund University.

[7] J. Chen, A. Dholakia, E. Eleftheriou, M. P. C. Fossorier and Xiao-Yu Hu, "Reduced-Complexity Decoding of LDPC Codes," *in IEEE Transactions on Communications*, vol. 53, no. 8, pp. 1288-1299, Aug. 2005.

[8] D. E. Hocevar, "A reduced complexity decoder architecture via layered decoding of LDPC codes," *Signal Processing Systems, 2004. SIPS 2004. IEEE Workshop on*, 2004, pp. 107-112.

[9] E. Sharon, S. Litsyn and J. Goldberger, "An efficient message-passing schedule for LDPC decoding," *Electrical and Electronics Engineers in Israel, 2004. Proceedings. 2004 23rd IEEE Convention of*, 2004, pp. 223-226.

[10] M. Sarajlic, L. Liu and O. Edfors, "Reducing the complexity of LDPC decoding algorithms: An optimization-oriented approach," *2014 IEEE 25th Annual International Symposium on Personal, Indoor, and Mobile Radio Communication (PIMRC)*, Washington DC, 2014, pp. 861-866.

[11] A. Sampath, P. Sarath Kumar and J. M. Holtzman, "On setting reverse link target SIR in a CDMA system," *Vehicular Technology Conference, 1997, IEEE 47th*, Phoenix, AZ, 1997, pp. 929-933 vol.2.

[12] K. I. Pedersen et al., "Frequency domain scheduling for OFDMA with Limited and Noisy Channel Feedback," *2007 IEEE 66th Vehicular Technology Conference*, Baltimore, MD, 2007, pp. 1792-1796.

[13] IEEE 802.11n-2012, IEEE Standard for Information TechnologyTelecommunications and information exchange between systems, Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications [Online]. Available: http://standards.ieee.org/getieee802/download/802.11-2012.pdf

[14] M. Jiang, C. Zhao, L. Zhang and E. Xu, "Adaptive offset min-sum algorithm for low-density parity-check codes," in *IEEE Communications Letters*, vol. 10, no. 6, pp. 483-485, June 2006.

[15] Z. Si, "Structured LDPC Convolutional Codes," Doctoral Thesis in Telecommunications *KTH Stockholm*, Sweden 2012 [Online]. Available: http://www.diva-portal.org/smash/get/diva2:576694/FULLTEXT01.pdf

[16] A.F. Molisch, *Wireless Communications.* United Kingdom: Wiley-IEEE press, pp. 417-422, 2011.

[17] T. S. Rappaport, *Wireless Communications: Principles and Practice.* Upper Saddle River, NJ: Prentice-Hall, 1996.

[18] C. D. Iskanderp, "A Matlab-based object-oriented approach to multipath fading channel simulation," Mathworks, Natick, MA, Tech. Rep. 18869, Feb. 2008.

[19] N. Chayat, "Tentative Criteria for Comparison of Modulation Methods," Document: IEEE P802.11-97/96, Sept. 1997.

[20] Y. S. Cho, J. Kim, W. Y. Yang, and C. G. Kang, *MIMO-OFDM Wireless Communications With MATLAB.* Hoboken, NJ: Wiley, p.29 Oct. 2010.

Appendix A

Test Appendix

LUND
UNIVERSITY

Series of Master's theses
Department of Electrical and Information Technology
LU/LTH-EIT 2016-529

http://www.eit.lth.se