Dhirubhai Ambani University

CT-216 Introduction to Communication Systems

LDPC for 5G NR Report

Group-25

Supervisor: Professor Yash Vasavada

A report submitted in partial fulfilment of the requirements of
the Dhirubhai Ambani University for the coursework in
CT-216 Introduction to Communication Systems

May 6, 2025

Team Members

Name	Role
Bhagya Majithiya	202301269
Ashka Pathak	202301270
Dhruv Jain	202301272
Jill Chhagnani	202301273
Vansh Vaishnani	202301274
Jevik Rakholiya	202301276
Rujal Jiyani	202301277
Dushyant Varshney	202301278
Kathan Balar	202301279
Lingampalli Venkata	202301280

Honor Code

We declare that:

- The work that we are presenting is our own work.
- We have not copied the work (the code, the results, etc.) that someone else has done.
- Concepts, understanding and insights we will be describing are our own.
- We make this pledge truthfully. We know that violation of this solemn pledge can carry grave consequences

Group-25
April 26, 2025

Abstract

This project presents the simulation and performance analysis of Low-Density Parity-Check (LDPC) codes using MATLAB, focusing on their application in reliable digital communication systems. LDPC codes are powerful error-correcting codes known for their near-capacity performance under iterative decoding. The project involves the complete transmission chain, starting from LDPC encoding of binary input data, followed by digital modulation (e.g., BPSK), transmission over an Additive White Gaussian Noise (AWGN) channel, and decoding using the belief propagation algorithm. Various parameters such as bit error rate (BER) and signal-to-noise ratio (SNR) are analyzed to evaluate the effectiveness of LDPC codes under different noise conditions. The simulation results confirm the robustness of LDPC codes and demonstrate their superiority over traditional error correction techniques in terms of performance and error resilience.

Chapter 1

Introduction

The rapid evolution of wireless communication systems has led to an ever-increasing demand for higher data rates, improved reliability, and lower latency. With the rollout of fifth-generation (5G) networks, these expectations have become even more pronounced. To meet such performance standards, advanced error correction techniques are essential, as they ensure the accurate transmission of data over inherently noisy communication channels.

1.1 Background and Problem Statement

Traditional error-correcting codes, while effective in earlier communication standards, fall short in terms of efficiency and scalability in modern high-throughput environments. This is where **Low-Density Parity-Check (LDPC)** codes come into play. LDPC codes are a class of linear block codes that provide near-Shannon limit performance while maintaining computational efficiency. They are particularly well-suited for the needs of 5G New Radio (NR), where robustness and speed are critical.

The problem at the heart of this project lies in understanding how LDPC codes are implemented within the 5G NR framework and how they enhance the reliability and efficiency of data transmission. Additionally, the complexity of encoding and decoding processes, particularly under varying noise conditions and channel constraints, requires careful study and simulation.

1.2 Aim and Scope of the Project

The primary aim of this project is to explore and demonstrate the implementation of **Low-Density Parity-Check (LDPC) codes** in the context of **5G New Radio (NR)** communication systems. The project focuses on understanding the theoretical foundations of LDPC coding, examining its role in achieving reliable and high-speed data transmission, and evaluating its performance under practical scenarios using simulations.

More specifically, the objectives include:

- Understanding the **mathematical structure** of LDPC codes, particularly the role of sparse parity-check matrices.

- Investigating the **encoding** and **decoding** processes, including both hard and soft decision decoding techniques.

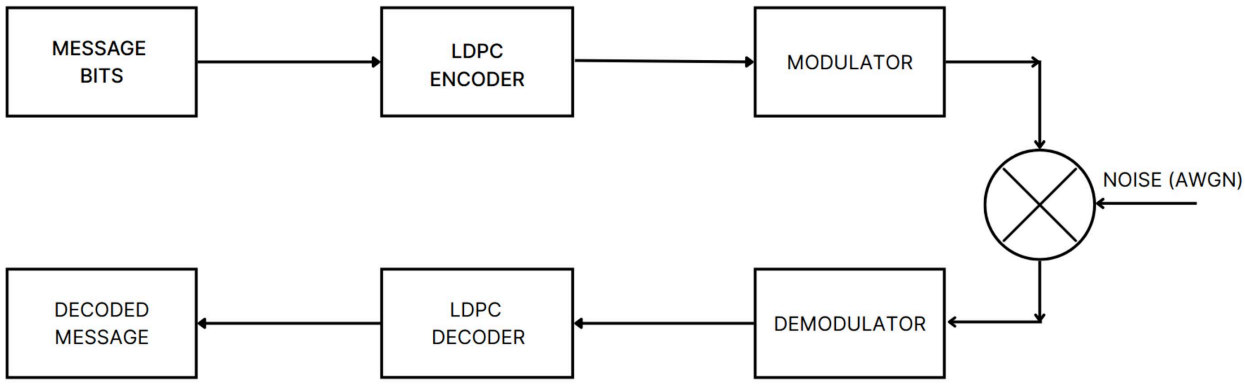
- Simulating the **performance of LDPC codes** using tools like MATLAB under Additive White Gaussian Noise (AWGN) channels.

- Analyzing how different factors such as **base graph selection**, **expansion factor**, **code rate**, and **number of decoding iterations** influence overall performance.

- Comparing the **bit error rate (BER)** and **probability of decoding success** for various signal-to-noise ratio (SNR) levels.

The scope of this project is limited to LDPC code structures and decoding strategies as defined in the 5G NR standard, specifically focusing on base graphs **NR_1_5_352** and **NR_2_6_52**.

1.3 Model of LDPC 5G NR



*Created in Canva

Chapter 2

Encoding

What are LDPC Codes?

LDPC (Low-Density Parity-Check) codes are **linear block codes** defined by **a sparse parity-check matrix H**. The term **"low-density"** refers to the fact that most of the elements in H are **zero**. This sparsity allows for **efficient** encoding and decoding.
A codeword c satisfies:

$$H \cdot c^T = 0 (\text{mod } 2)$$

LDPC codes can be visualised using Tanner graphs, which consist of:

- Variable nodes (representing code bits)
- Check nodes (representing parity checks)

The positions of 1s in the parity-check matrix H determine edges in the graph.

Encoding converts message bits into a codeword by adding redundant parity bits using a parity-check matrix. This allows for reliable error detection and correction at the receiver.

LDPC Encoder receives the message bits as an input vector **m = [m1, m2, ..., mk]**, while the sparse parity check matrix (H) is built using the base graph and lifting factor Z.

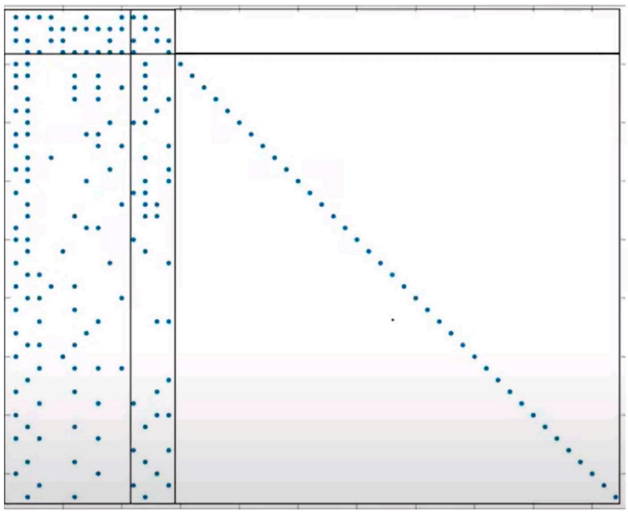
Structure of Base Graph

Block Structure of Base Matrix:

A E O
B C I

A: Connects the first 4 parity bits to systematic bits.Most dense block of Base graph.
E: Smallest block of base graph. Square matrix having double diagonal structure. Use for core parity generation.
O: All zero matrices.It reduces complexity
B: Sparsh block of base graph. Connects remaining parity bits to systematic bits.
C:This block links extended parity to core parity.
I:Identity square matrix use for systematic parity

A, B, C, E define parity relations, while O and I optimize performance.



*taken from slides of Prof. Yash Vasavda

Base Graphs (BG1 and BG2) in 5G NR

5G defines two base graphs to support different block lengths and code rates.

- **BG1**: For large block sizes and high code rates.Dimensions are **46x68**.

- **BG2**:For smaller block sizes and low code rates.Dimensions are **42x52**.

BG1:
A: 4 x 22 E: 4 x 4 O: 4 x 42 all zero
B: 42 x 22 C: 42 x 4 I: 42 x 42 all identity

BG2:
A: 4 x 10 E: 4 x 4 O: 4 x 38 all zero
B: 38 x 10 C: 38 x 4 I: 38 x 38 all identity

Base graphs are sparse matrices with integers or -1 indicating shifts or zero blocks.

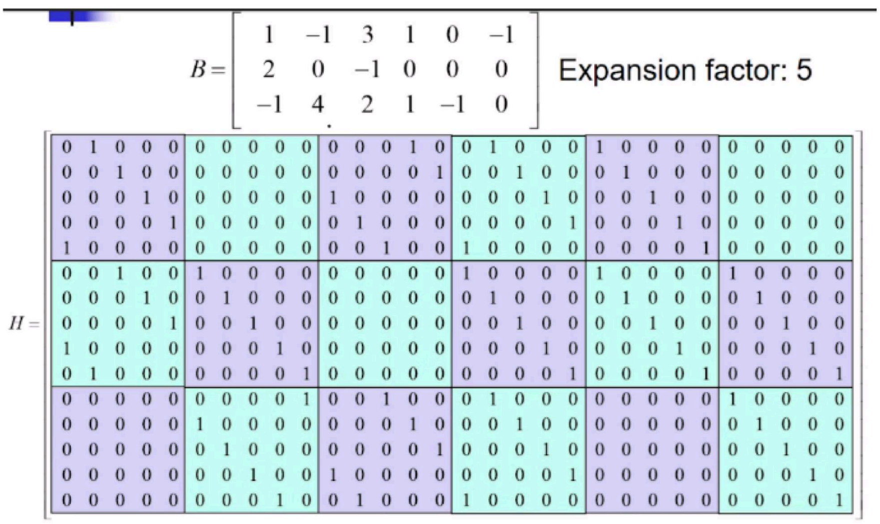
Expansion Factor : Base matrix has elements like 0, -1 and any number between 0 to Z-1, where Z is an expansion factor

Every base graph is blown up into a big parity-check matrix H by expansion factor (Z).

Each element Bi,j in the base graph defines a z × z submatrix:

- **Bi,j = −1 ⇒ Bi,j = zero submatrix.**
- **Bi,j = 0 ⇒ Bi,j = identity submatrix.**
- **Bi,j = p ⇒ Bi,j = circularly shifted identity matrix by p positions.**
-

*taken from slides of Prof. Yash Vasavda



Encoding using Base Matrix

In order to do encoding, we utilize the structure inherent in the base matrix. The important point is that because of this particular structure, most of the parity bits (numbered 5 and greater) can be found by solving easy linear equations.
But for the first four parity bits, we have to solve a system of four linear equations at once. The procedure is easier since the matrix is of special double-diagonal form.

Let's understand this with an example:

$$H = \begin{bmatrix} I_1 & 0 & I_3 & I_1 & I_2 & I & 0 & 0 \\ I_2 & I & 0 & I_3 & 0 & I & I & 0 \\ 0 & I_4 & I_2 & I & I_1 & 0 & I & I \\ I_4 & I_1 & I & 0 & I_2 & 0 & 0 & I \end{bmatrix}$$

We begin with message vector: m = [m_1, m_2, m_3, m_4]
We add 4 parity bits: [p_1, p_2, p_3, p_4]
So, the full codeword becomes: [m_1, m_2, m_3, m_4, p_1, p_2, p_3, p_4]

From the H matrix we can derive the following equations:

1. Equation 1: $I_1m_1 + I_3m_3 + Im_4 + I_2p_1 + Ip_2 = 0$
2. Equation 2: $I_2m_1 + Im_2 + I_3m_3 + Ip_2 + Ip_3 = 0$
3. Equation 3: $I_4m_2 + I_2m_3 + Im_4 + I_1p_1 + Ip_3 + Ip_4 = 0$
4. Equation 4: $I_4m_1 + I_1m_2 + Im_3 + I_2p_1 + Ip_4 = 0$

Now,let's add all four equations:
(NOTE:We are performing addition and doing mod 2 so if two same matrix say x are added, it will become 2x which is 0 after doing mod 2).
$I_1p_1 = I_1m_1 + I_3m_3 + I_1m_4 + I_2m_1 + I_2m_2 + I_3m_3 + I_4m_2 + I_2m_3 + I_4m_1 + I_1m_2 + Im_3$

By solving this we will get p_1 , and by substituting p_1 on above equations we will also get p_2 , p_3 and p_4.

Encoding Procedure in 5G

1. **Choose base graph (BG1 or BG2) depending upon payload size.**
   The choice between Base Graph 1 (BG1) and Base Graph 2 (BG2) is largely dependent on the payload size (A) and the target code rate (R). BG1 is generally preferred for large payloads (A ≥ 384 bits) and high code rates since it has a more compact parity-check matrix appropriate for high-throughput applications. BG2, however, is designed for low code rates and small payloads and provides a more compact and versatile structure for low-latency use. The selection is made based on rules specified in the 3GPP TS 38.212 specification to provide efficient decoding and encoding while satisfying performance requirements.

2. **Calculate lifting size (Z) from standard tables.**
   Once a base graph has been chosen, the second task is to specify the lifting size, Z, which stretches the base graph up to the needed codeword length. The lifting size must be taken from a predefined range of values (e.g., Z = 2, 3, 4,., 384) so that the total amount of information bits after lifting is greater than or equal to the payload size A.

3. **Extend the base graph to create a full H matrix.**
   Having settled the base graph and the lifting size, every entry of the base graph is replaced with a larger matrix to form the complete parity-check matrix H. Precisely, every non-negative value in the base graph is replaced with a Z × Z circulant permutation matrix right-shifted by the given value, and -1 entries are replaced by Z × Z zero matrices. This organized expansion forms a big but thin matrix H, maintaining the low-density characteristic necessary for effective LDPC decoding. Circulant matrices guarantee that the structure is hardware-friendly and amenable to rapid parallel implementation.

4. **Apply structured encoding to create parity bits.**
   The encoding process involves computing the parity bits required to complete the codeword. This is done by solving the equation $H \cdot cT$=0 where c is the codeword vector. The matrix H is partitioned into submatrices, typically as H=[H1|H2], where H1 corresponds to the message portion and H2 to the parity part. Using structured encoding techniques, such as back-substitution or triangularization, the parity bits are calculated as a linear combination of the message bits. This structured approach avoids the need to invert large matrices, making the encoding process computationally efficient.

5. **Append message and parity bits together to create codeword**
   Once the parity bits are calculated, the next step is to construct the complete codeword. This is accomplished by combining the original message bits with the newly calculated parity bits into a single vector: c=[m | p] where m is the message and p the parity. The resulting codeword fulfills all parity-check equations stipulated by the matrix H and is therefore transmission-ready over the communication channel. The codeword will be utilized at the receiver for decoding with iterative algorithms, e.g., belief propagation, to extract the original message even with transmission errors.

Chapter 3

Rate Matching

Code rate matching is a technique used to adjust the effective transmission rate without altering the fundamental structure of the LDPC code. This is essential in communication systems where varying data rates are required without designing new codes.

Understanding the Code Structure:

In LDPC codes, a base matrix of size m×n is used, and it is expanded by a factor z to form the full parity-check matrix H. This expansion defines a code of length:

$Codeword\ length\ =\ n.z$
Out of this total length:
$(n\ -\ m).z\ are\ information\ bits.$
$m.z\ are\ parity\ bits.$
Puncturing:

Puncturing means not transmitting certain bits of the codeword to increase the code rate. These bits are known at the encoder but are treated as erasures (unknown) at the decoder.

To adapt a fixed LDPC code to various code rates without redesigning the code, we use puncturing.

We want to find the value of puncturing required to achieve a desired code rate x/y .

From the base matrix:

$Total\ bits\ before\ puncturing\ =\ n.z$

$Total\ information\ bits\ before\ puncturing\ =\ (n\ -\ m).z$

$Punctured\ bits\ =\ a.z$

$Remaining\ bits\ after\ initial\ 2.z\ punctured\ =\ (n\ -\ 2).z$

$Out\ of\ this,\ a{\cdot}z\ bits\ are\ additionally\ punctured \rightarrow Transmitted\ bits\ =\ (n\ -\ 2).z\ -\ a.z$

Initially, 2 columns are punctured in most cases because of their lower reliability and to avoid redundancy.

The code rate is then:

$$\frac{message\ bits}{trasmitted\ bits} = \frac{(n-m).z}{(n-2).z - a.z} = \frac{x}{y}$$

This equation helps us compute the number of punctured bits $a.z$ required to achieve the desired rate $\frac{x}{y}$.

Steps to Achieve Target Code Rate $\frac{x}{y}$

1. **Determine Code Structure**:
    - Base matrix size $m\ x\ n$
    - Expansion factor z
2. **Compute**:
    - Total codeword bits: $n.z$
    - Information bits: $(n - m).z$
    - Punctured bits: Start with $2.z$, and compute additional $a.z$
3. **Apply Rate Matching Formula**:
$$\frac{(n-m).z}{(n-2).z - a.z} = \frac{x}{y}$$
4. **Solve for a**:
This gives the amount of additional puncturing needed to achieve the desired rate.

Chapter 4

BPSK Modulation

BPSK is a digital modulation technique used to transmit binary data over a communication channel. It is the simplest form of phase shift keying and is widely used in systems where robustness and simplicity are key.

Working Principle:

- In BPSK, binary bits are mapped to symbols:
    - Bit '0' is represented by 1 (Phase shift of 0°).
    - Bit '1' is represented by -1 (Phase shift of 180°).

Advantages:

- **High Noise Immunity** – Due to the 180° phase difference, BPSK is less susceptible to noise.
- **Simple Implementation** – Requires only two phase states, making modulation and demodulation straightforward.
- **Better BER Performance** – Lower Bit Error Rate (BER) in noisy environments compared to other binary modulation schemes.

Disadvantages:

- **Low Spectral Efficiency –** Transmits only 1 bit per symbol, limiting data rates.
- **Bandwidth Requirement** – Higher bandwidth compared to multi-level modulation schemes like QPSK.

Chapter 5

Additive White Gaussian Noise (AWGN)

The AWGN channel is a fundamental model in digital communication systems, used to mimic random noise in a communication channel.

- **A**dditive: Noise is added to the signal, not multiplied or altered structurally.
- **W**hite: Power is uniformly distributed across all frequencies — like white light contains all colours.
- **G**aussian: The noise amplitude follows a Gaussian (normal) distribution with Mean = 0 and Variance = σ²

In the AWGN model, the received signal $y(t)$ is the sum of the transmitted $x(t)$ and Gaussian noise $n(t)$:

$$y(t) = x(t) + n(t)$$

Where:

- $n(t) \sim N(0, N_0/2)$ represents white Gaussian noise with a constant power spectral density $N_0/2$
- The channel does not introduce fading or interference, allowing isolated study of noise impact

This model is extensively used for analyzing the performance of LDPC codes in clean, idealized conditions.

Chapter 6

Shannon Capacity and Theoretical Limit

The Shannon theorem defines the maximum possible data rate C of a channel with bandwidth B and signal-to-noise ratio $\frac{S}{N}$:

$$C = B.log_2(1 + \frac{S}{N}) \text{ [bits/sec]}$$

For a unit bandwidth AWGN channel (normalized to $B = 1\ Hz$), the capacity simplifies to:

$$C = log_2(1 + SNR) \text{ [bits/sec]}$$

Where SNR is the signal-to-noise ratio $\frac{E_b}{N_0}$, often expressed in decibels (dB). The Shannon Limit represents the theoretical minimum SNR per bit $\frac{E_b}{N_0}$ required for reliable communication at a given code rate R. This is derived by rearranging the capacity formula:

$$R = \frac{C}{B} = log_2(1 + \frac{E_b}{N_0}.R) \Rightarrow \frac{2^R - 1}{R}$$

At low rates (e.g., $R \to 0$), the minimum $\frac{E_b}{N_0}$ approaches:

$$(\frac{E_b}{N_0})_{min} = ln(2) \approx -1.59\ dB$$

This is known as the Shannon limit — the absolute lower bound SNR for any coding scheme to achieve arbitrarily low error probability.

LDPC codes are known for their near-shannon-limit performance, particularly under iterative decoding in AWGN channels. The decoding algorithms, such as the Sum-Product Algorithm (SPA), leverage probabilistic message-passing to iteratively converge on likely transmitted bits, achieving Bit Error Rates (BER) very close to theoretical limits. This makes LDPC a powerful choice in noise-affected environments like 5G NR.

Chapter 7

Decoding

Decoding is a critical phase in any error-correcting code system, as it determines whether the received message can be correctly interpreted despite transmission errors. In LDPC coding, decoding plays an even more significant role due to the sparse and structured nature of the parity-check matrix, which allows for efficient iterative decoding algorithms.

LDPC decoding is typically carried out using **graph-based approaches** that operate on the **Tanner graph** representation of the code. In this bipartite graph, variable nodes (representing codeword bits) and check nodes (representing parity-check constraints) exchange messages iteratively to converge toward a valid codeword that satisfies all parity conditions.

This project explores two major decoding strategies: **hard decision decoding**, which uses binary decisions without considering signal confidence, and **soft decision decoding**, which incorporates reliability information through techniques like the **Min-Sum algorithm**.

Tanner Graph:

A Tanner graph is a bipartite graph representation of Low-Density Parity-Check (LDPC) codes, which are powerful error-correcting codes used in modern communication systems. Tanner graphs provide a visual and mathematical framework for understanding the structure of LDPC codes and implementing iterative decoding algorithms like belief propagation (message passing).

Key Features of Tanner Graphs:

- **Bipartite Structure:** Consists of two disjoint sets of nodes:
- **Variable Nodes (VNs):** Represent codeword bits.
- **Check Nodes (CNs):** Represent parity-check constraints.
- **Edges:** Connections between VNs and CNs based on the parity-check matrix (H). An edge indicates that a particular bit is involved in a specific parity check.

The Tanner graph is fully defined by the parity-check matrix (H) of the LDPC code. For an LDPC code with:

- n variable nodes (codeword length)
- m check nodes (number of parity constraints)

The H matrix is an m × n binary matrix where:

- H(i,j) = 1: Indicates an edge between Check Node (CN) i and Variable Node (VN) j.
- H(i,j) = 0: No connection exists.
- Each row of H is a Check Node (CN).
- Each column of H is a Variable Node (VN).

Message Passing in Tanner Graph:

We have implemented the Tanner graph using a message-passing matrix between CNs and VNs. The equation used is:

$$L = r .* H$$

r: Received bits (after demodulation).
H: Parity-check matrix.
L: Message-passing matrix between CNs and VNs.

7.1 Hard Decision Decoding

Hard Decision Decoding (HDD) is a fundamental decoding technique in digital communication systems where the receiver makes a binary decision (0 or 1) on each received bit before error correction. Unlike soft decoding, which utilizes probabilistic information, HDD discards signal reliability metrics, prioritizing speed and simplicity over optimal performance.

Suppose we receive this analog signal:
[0.8, −0.3, 0.9, −0.1]
You apply a threshold (e.g., 0):

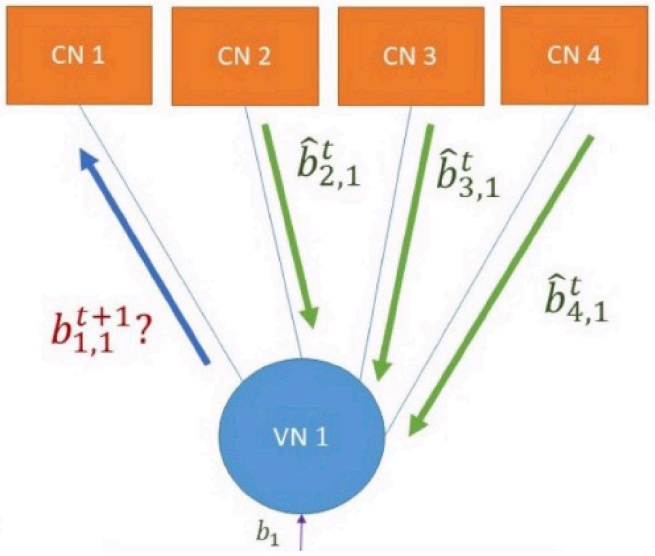
- Values > 0 → 0
- Values ≤ 0 → 1

So you get:
[0, 1, 0, 1]

LDPC codes use Tanner graphs with Variable Nodes (VNs) and Check Nodes (CNs). In HDD, messages are binary values (0 or 1).

Steps for Hard Decision Decoding:

Step 1: Variable Node to Check Node (VN → CN)

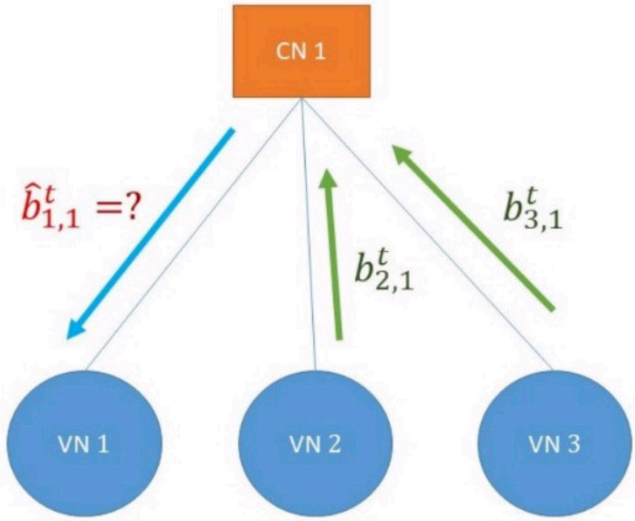


- **Initialization:**

Each VN j is assigned a hard-decoded bit b_j from the received signal.

- **First Iteration:**
  Each VN sends the bit it currently holds to all the Check Nodes (CNs) it is connected to.

Step 2: Check Node to Variable Node (CN → VN)

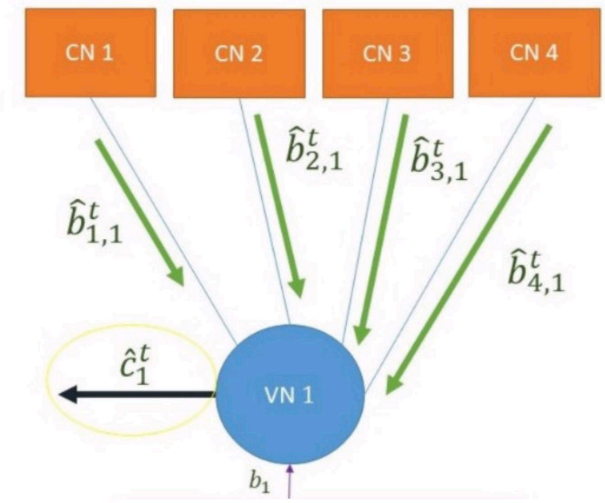


At this point, every check node receives bits from all its connected variable nodes.
Each CN i computes a parity-check message for connected VNs:

- Each CN enforces a single parity-check (SPC) constraint, for every VN connected to it. For each VN j, CN i calculates XOR of all incoming bits except $b_{j,i}$.

- The result is then sent back to VN j as a correction suggestion

Step 3: Iterative Decoding Process



- Each VN gathers responses from all the checknodes it's linked to.
- Now, each VN performs a majority vote among the bits received from CNs, along with the original bit it was initialized with.
- The result of this vote becomes the VN's new value. Also, in the case of a tie (equal number of 1's and 0's),we are assuming that the variable node decides its value as 0.

This iterative process continues either until a maximum number of iterations is reached, or until no further changes occur in the current decoded codeword.

7.2 Soft Decision Decoding

Soft decoding is a technique used in digital communications to improve the accuracy of received data by considering the confidence level of each bit, rather than making a simple binary decision.

In soft scheme, the messages are the conditional probability that in the given received vector received bit is a 1 or a 0. The Min-Sum algorithm is a simplified version of the Belief Propagation (BP) or Sum-Product algorithm used for decoding LDPC codes. It approximates the computations in the sum-product algorithm to reduce complexity, especially for hardware implementations.

Priori probabilities for the received bits is the input probabilities as here they were known in advance before running the LDPC decoder. The bit probabilities returned by the decoder are called the a posterior probabilities

Sum-Product Algorithm:

1. Initialization of Priori probabilities:

$$\lambda_i = \log\left(\frac{P(x_i = 0 \mid y_i)}{P(x_i = 1 \mid y_i)}\right)$$ where for AWGN channel $$\lambda_i = \frac{2y_i}{\sigma^2}$$

$$v_{i\to j}^{(0)} = \mathbf{L}_i$$ This is the message sent from variable node i to check node j in the first iteration.

2. Message Passing: The algorithm proceeds iteratively with two main steps

   Check Node Update:
   Each check node j computes a message to each variable node i∈N(j) using the messages received from all other VNs i′∈N(j)\i

$$L = \log\left(\frac{P(x=0)}{P(x=1)}\right)$$ this can be written as $$\frac{P(x=0)}{P(x=1)} = e^L$$

Therefore $P(x=0) = \frac{e^L}{1+e^L}, \quad P(x=1) = \frac{1}{1+e^L}$

We know,

$$\tanh\left(\frac{x}{2}\right) = \frac{e^x - 1}{e^x + 1}$$

Now replace 'x' with 'L':

$$\tanh\left(\frac{L}{2}\right) = \frac{e^{L/2} - e^{-L/2}}{e^{L/2} + e^{-L/2}} = \frac{e^L - 1}{e^L + 1}$$

Which in terms of probability can be written as:

$$\tanh\left(\frac{L}{2}\right) = \frac{P(x=0) - P(x=1)}{P(x=0) + P(x=1)} = P(x=0) - P(x=1)$$

Since P(x = 0) + P(x = 1) = 1, we get:

$$\tanh\left(\frac{L}{2}\right) = 1 - 2P(x=1)$$

Also we know,

$$\tanh\left(\frac{x}{2}\right) = \frac{e^x - 1}{e^x + 1} \quad \Rightarrow \quad \tanh^{-1}(x) = \frac{1}{2}\log\left(\frac{1+x}{1-x}\right)$$

Simplifying we get:

$$c_{j\to i} = 2\tanh^{-1}\left(\prod_{i'\in N(j)\setminus i} \tanh\left(\frac{v_{i'\to j}}{2}\right)\right)$$

Variable node update:

Each variable node i updates the message to each check node j∈N(i)

$$v_{i\to j}^{(t)} = L_i + \sum_{j'\in N(i)\setminus j} c_{j'\to i}^{(t)}$$

Min-Sum Approximation of Check Node Update values:

Sum-Product Algorithm exact computation involves multiple hyperbolic tangent and inverse hyperbolic tangent evaluations, which are computationally expensive.

Therefore we approximate it's value using Min-Sum:

For small values of x, we can expand $\tanh^{-1}(x)$ using its Taylor series:

$$\tanh^{-1}(x) = x + \frac{x^3}{3} + \frac{x^5}{5} + \cdots \approx x \quad \text{for small } x$$

More precisely,

$$\tanh^{-1}(x) \approx \text{sign}(x) \cdot \min|x|$$

In LDPC decoding, we encounter expressions like:

$$\tanh^{-1}(\tanh(a) \cdot \tanh(b))$$

This expression behaves similarly to:

$$\min(|a|, |b|)$$

More precisely, we have the identity:

$$\tanh^{-1}(\tanh(a)\tanh(b)) = \operatorname{sgn}(a) \cdot \operatorname{sgn}(b) \cdot \min(|a|, |b|) + \text{correction}$$

Therefore are approximation Check Node Update becomes:

$$c_{j\to i}^{(t)} \approx \left(\prod_{i' \in N(j)\setminus i} \operatorname{sign}\left(v_{i'\to j}^{(t)}\right) \right) \cdot \min_{i' \in N(j)\setminus i} \left| v_{i'\to j}^{(t)} \right|$$

3. Posterior probabilities:
   At each variable node i, compute the updated LLR estimate for the bit:

$$L_i^{(t)} = L_i + \sum_{j \in N(i)} c_{j\to i}^{(t)}$$

   This value is used for making a hard decoding:

$$\hat{x}_i = \begin{cases} 0, & \text{if } L_i^{(t)} \geq 0 \\ 1, & \text{if } L_i^{(t)} < 0 \end{cases}$$

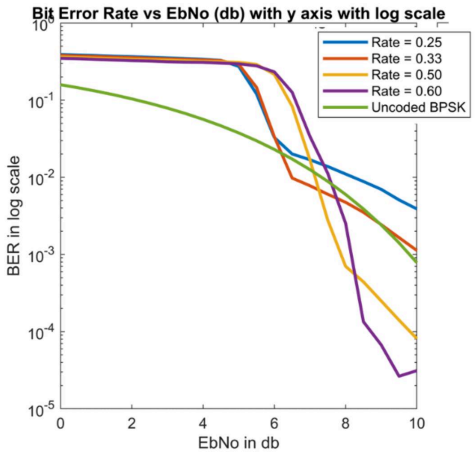
Chapter 8

Results

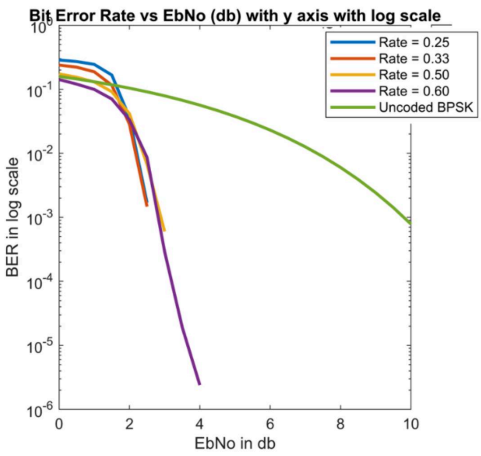
8.1 Graphs of Matrix NR_2_6_52

Bit Error Rate Vs Signal to Noise Ratio(dB) In Log Scale

HARD DECISION DECODING

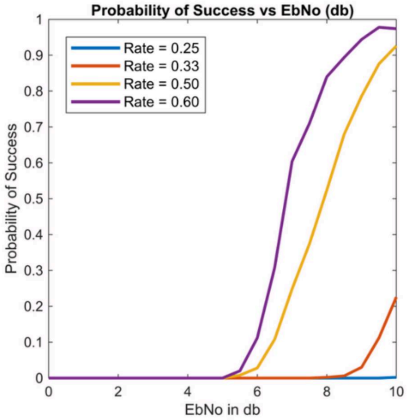


SOFT DECISION DECODING



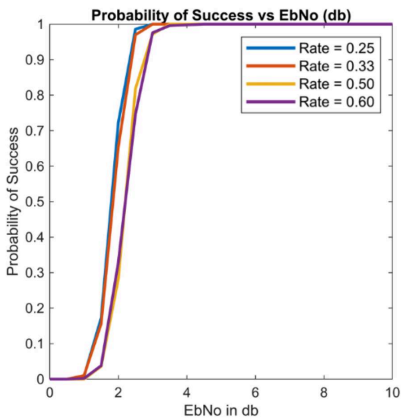
Probability of Success Vs Signal to Noise Ratio(dB)

HARD DECISION DECODING



As the Code Rate decreases, higher value of Eb/N0 is required to achieve greater probability of Success for Hard Decision Decoding.

SOFT DECISION DECODING

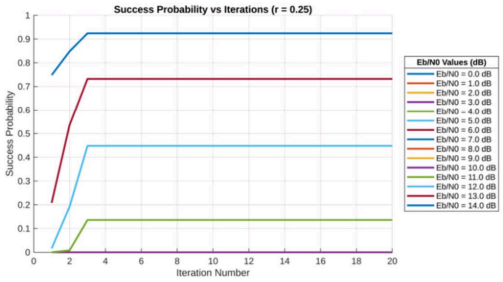


In Soft Decision Decoding, for any Eb/N0 above 3, Probability of Success is almost 100%.

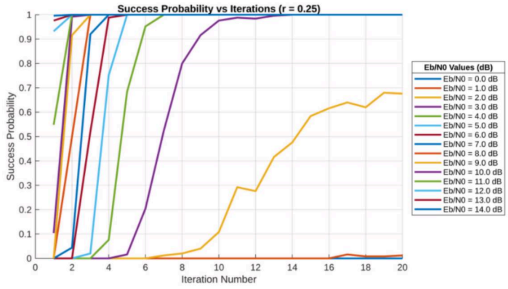
Probability of Success Vs No. of Iterations

CODE RATE = 1/4

HARD DECISION DECODING

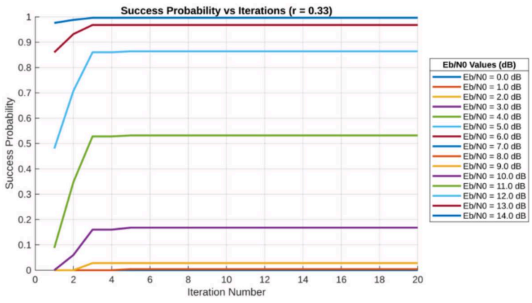


SOFT DECISION DECODING



CODE RATE = 1/3

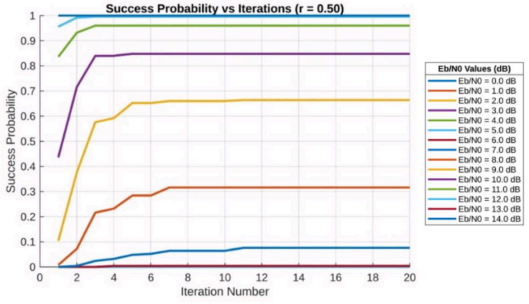
HARD DECISION DECODING



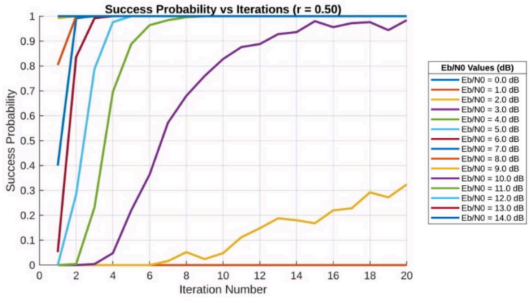
SOFT DECISION DECODING

CODE RATE = 1/2

HARD DECISION DECODING

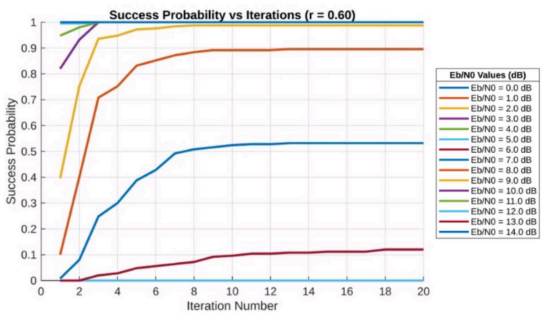


SOFT DECISION DECODING

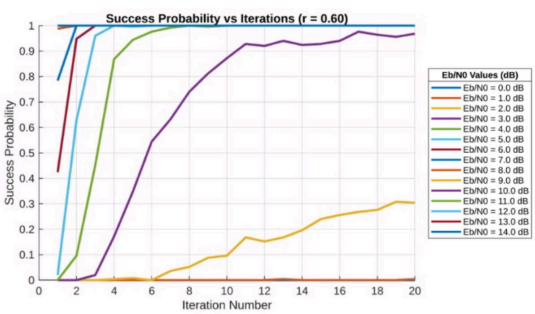


CODE RATE = 3/5

HARD DECISION DECODING



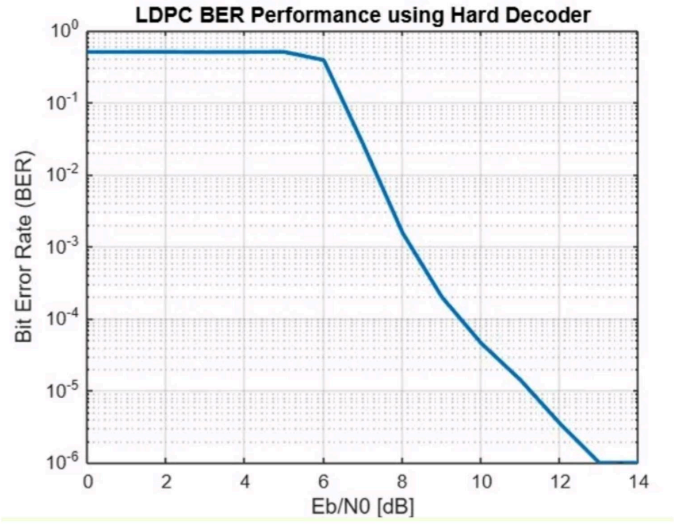
SOFT DECISION DECODING



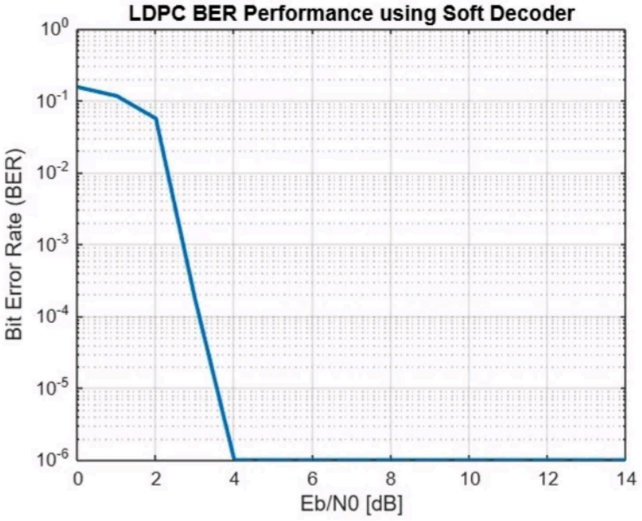
8.2 Graphs of Matrix NR_1_5_352

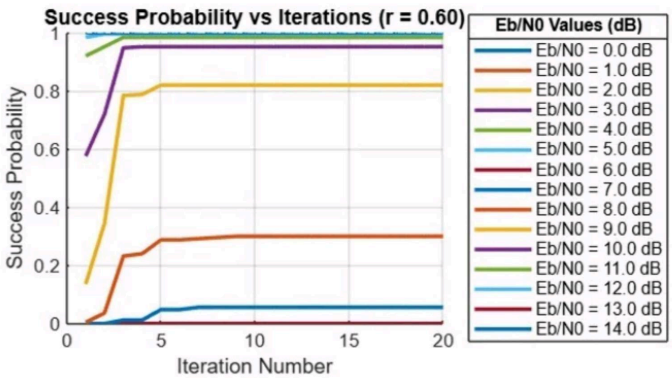
i) Code Rate = ⅓

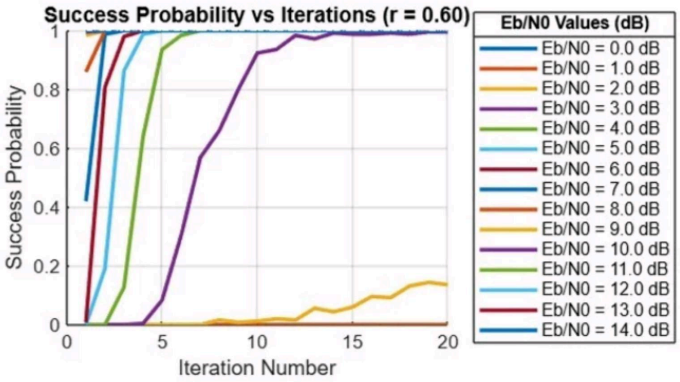
Hard Decoder

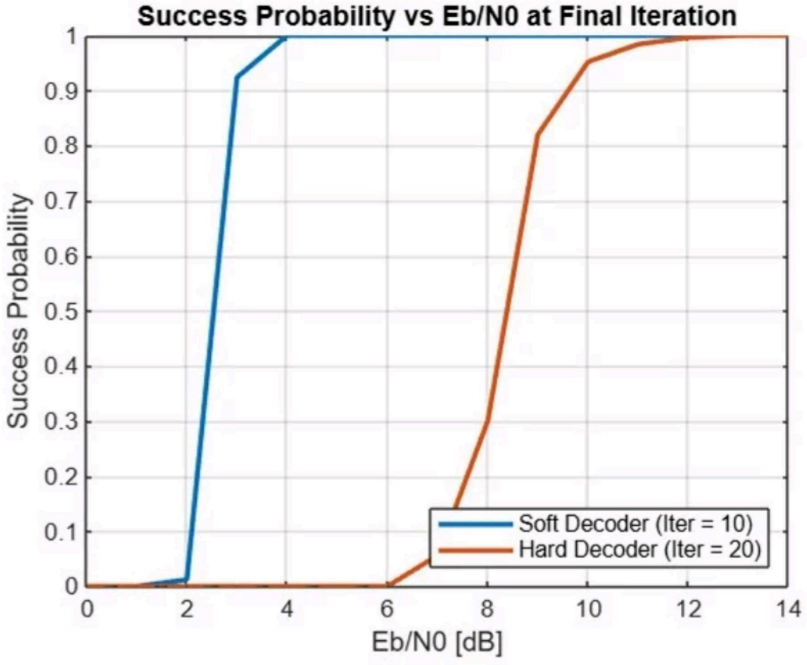


Soft Decoder



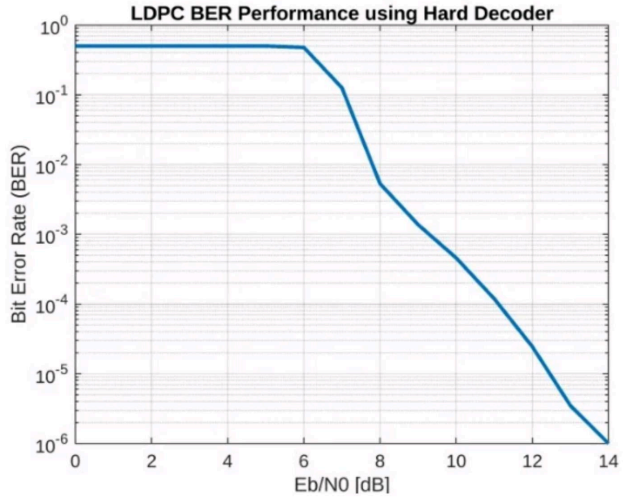




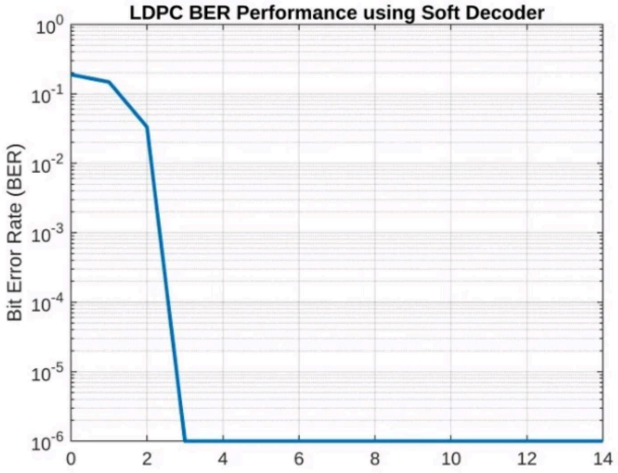


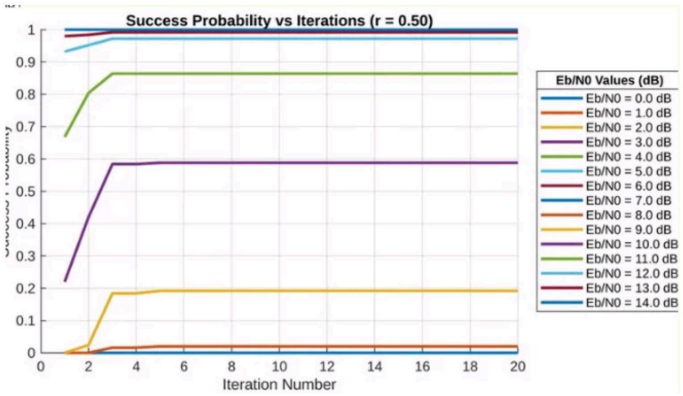
ii) Code Rate = ½

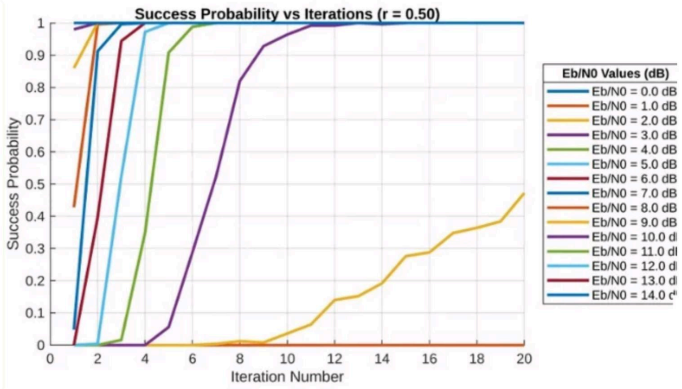
Hard Decoder

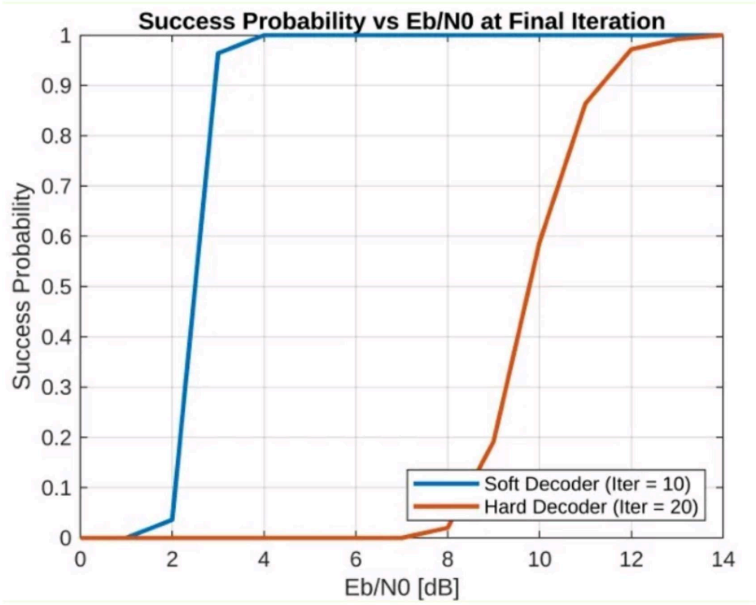


Soft Decoder



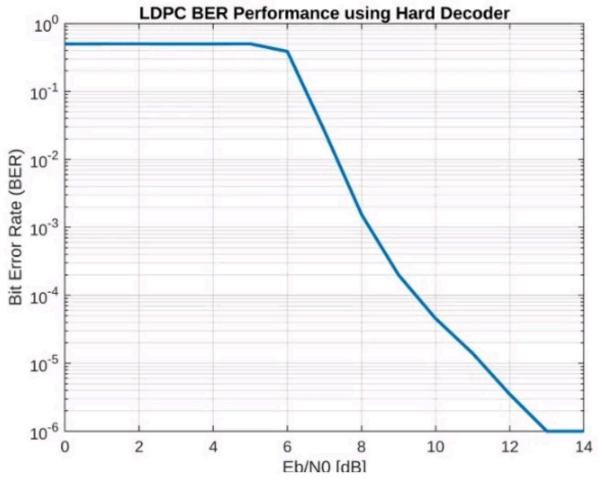


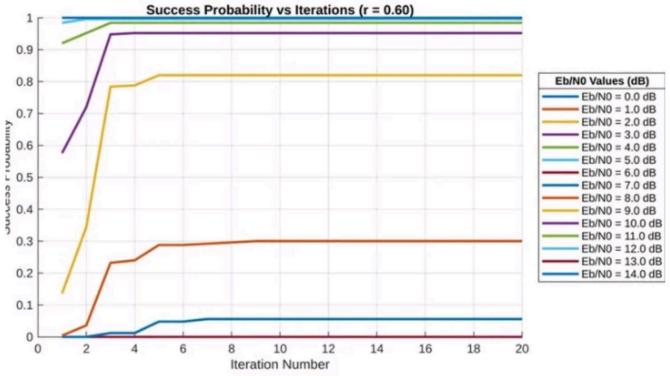




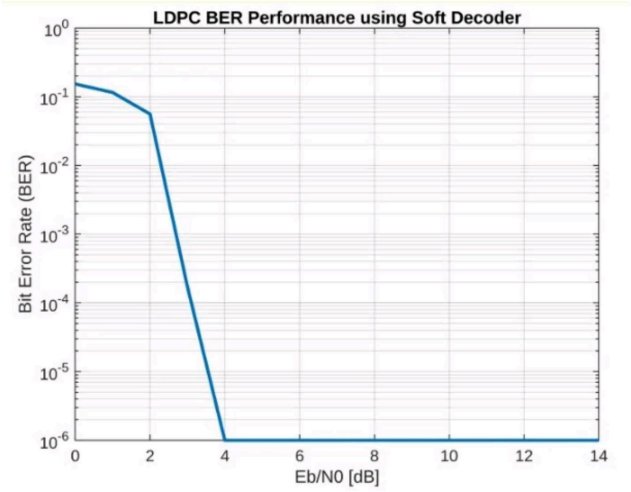
iii) Code Rate = ⅗

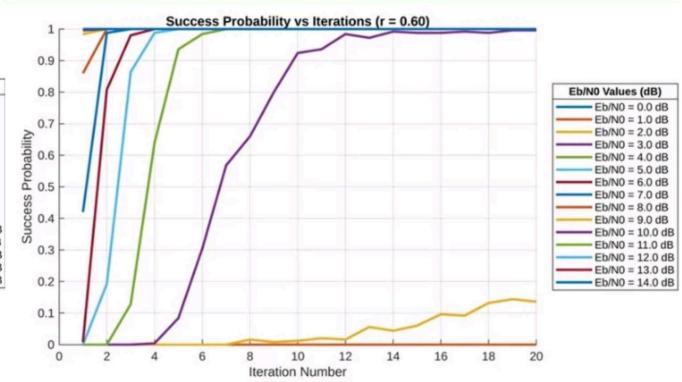
Hard Decoder

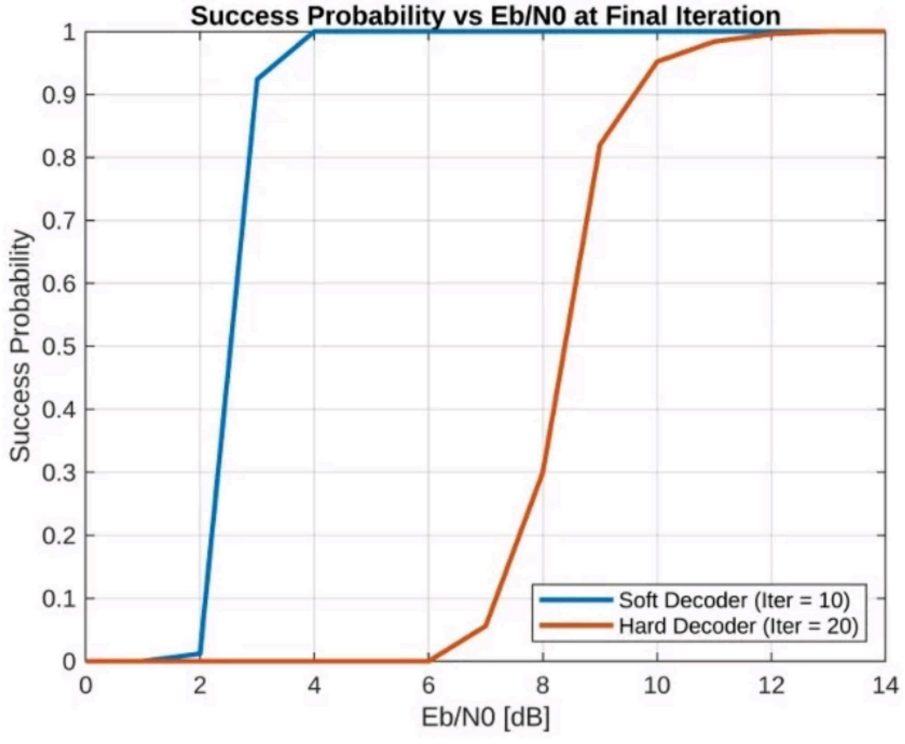




Soft Decoder

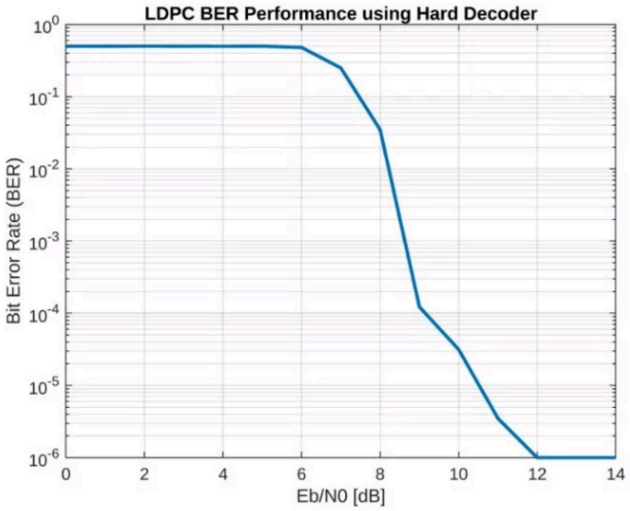


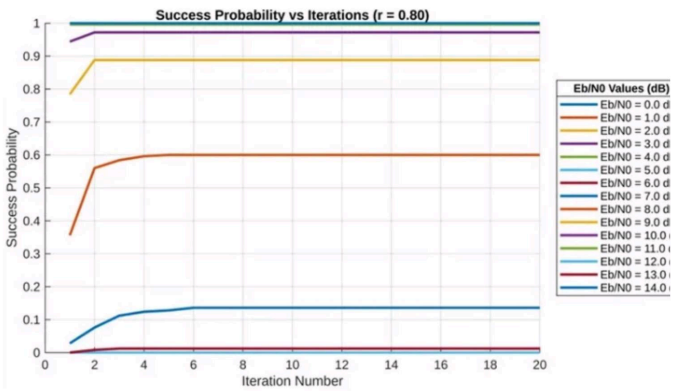




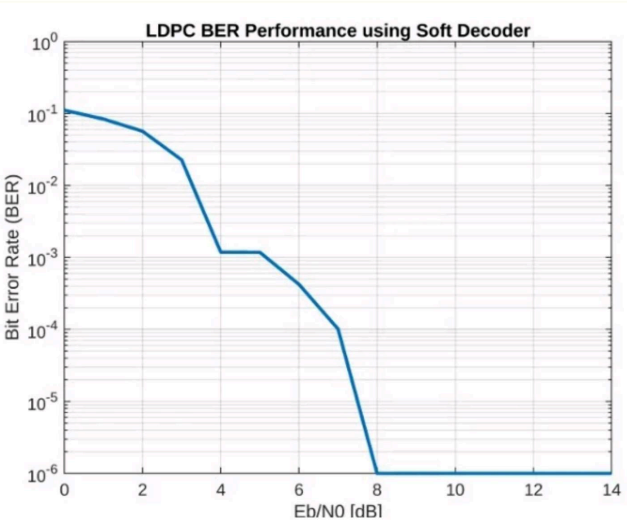
iv) Code Rate = ⅘

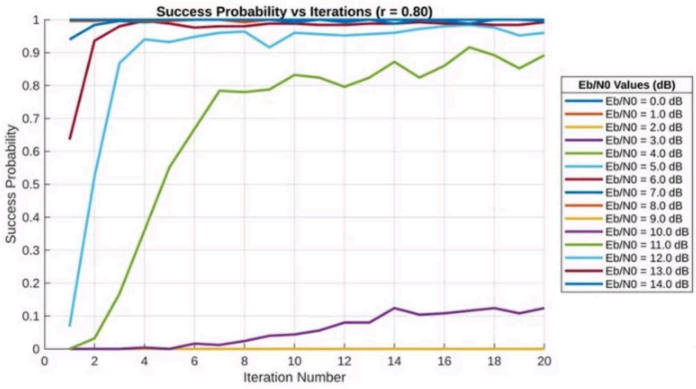
Hard Decoder

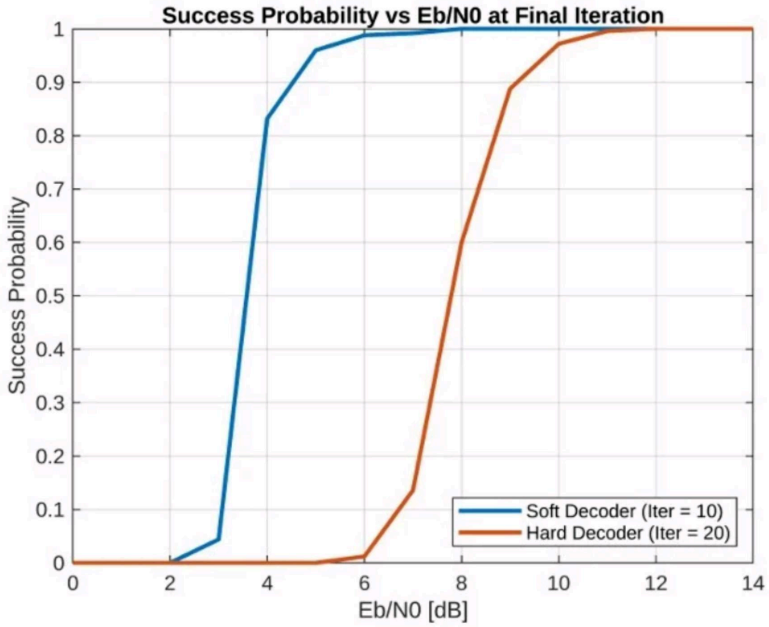




Soft Decoder







8.3 Observations from the Graphs

- Sum-Product Algorithm achieved the best BER performance across all SNR values. BER decreased rapidly with increasing SNR.
- Min-Sum Algorithm closely followed SPA, with slightly higher BER at low SNR.
- Hard Decision Decoding lagged behind both SPA and Min-Sum, especially in noisy conditions. Its BER dropped much more slowly with increasing SNR.

SNR (dB)	SPA BER	Min-Sum BER	Hard Decision BER
Low	High	High	Very High
Medium	Low	Slightly higher	Much higher
High	Very Low	Low	Moderate

Key Characteristics of Soft Decision Decoding:

- **Reliability Awareness:** Instead of converting received signals directly into bits, soft decoding considers the confidence level (likelihood) of each bit being a 0 or 1 using Log-Likelihood Ratios (LLRs).
- **Improved Accuracy:** By utilizing probabilistic information, soft decoding significantly reduces the Bit Error Rate (BER), especially in low signal-to-noise ratio (SNR) environments.
- **Iterative Message Passing:** Operates on the Tanner graph by exchanging real-valued messages between variable and check nodes, refining bit estimates with each iteration.
- **Higher Computational Complexity:** Requires more resources and time compared to hard decoding due to continuous value processing and multiple iterations.
- **Algorithmic Variants:** Includes approaches like the Min-Sum and Sum-Product algorithms, which trade off between performance and computational efficiency.

Key Characteristics of Hard Decoding:

- **Binary Quantization:** Converts analog received signals into discrete bits using a threshold (e.g., 0 for BPSK).
- **Low Complexity:** Eliminates the need for probability calculations, enabling faster decoding.
- **Suboptimal Performance:** Higher Bit Error Rate (BER) compared to soft decoding, especially in low-SNR scenarios.

8.4 Conclusion

LDPC codes offer a powerful way to correct errors in digital communication. The structure of their parity-check matrices allows for efficient decoding using iterative methods. In this project, we studied three decoding strategies — Hard Decision, Min-Sum, and Sum-Product — and evaluated their performance on a small LDPC code. The results confirm that while Sum-Product is the most accurate, Min-Sum provides a good balance of performance and efficiency, and Hard Decision decoding, while simple, is significantly less effective.
As LDPC codes continue to play a central role in systems like 5G NR, understanding their decoding behaviour is critical for building reliable and efficient communication systems.

Bibliography

1. NPTEL-NOC IITM Course : **LDPC and Polar Codes in 5G Standard by Prof. Andrew Thangraj**
2. **An Introduction to Low-Density Parity-Check Codes** by Paul H. Siegel, Electrical and Computer Engineering University of California, San Diego
3. **Introduction to Low-Density Parity Check Codes** by Brian Kurkoski
4. **CT-216 Introduction to Communication Systems Lecture Slides** by Prof. Yash Vasavada, DAIICT Winter 2025
5. Different references from ShareTechNote Handbook on 5G
6. Generalized Simplified Variable-Scaled Min Sum LDPC decoder for irregular LDPC Codes