CT216: Introduction to Communication Systems

End of the Semester Project

Project Report (LaTeX)

LDPC Decoding for 5G NR

Group - 06

Professor: Yash Vasavada
Mentor: Madhav Jivani

This report is made by only three students: Maulik Khoyani (202301062), Mahek Vaghera (202301066) and Dhruvil Mehta (202301061)

Team Members

202301057	Jenish Vasani
202301058	Bhanderi Raj Sureshbhai
202301059	Shraddha Rathod
202301060	Neeraj Vania
202301061	Dhruvil Mehta
202301062	Maulik Khoyani
202301063	Vrajesh Dabhi
202301065	Vansh Padaliya
202301064	Urvesh Dabhi
202301066	Mahek Vaghera
202101132	Poojan Shah

1 BPSK Modulation and Error Analysis

A message bits sequence undergoes the following process:

$$\text{bits} \xrightarrow{\text{BPSK mapper}} s \xrightarrow{\text{adding noise}} r = s + n \xrightarrow{\text{Decision Device}} \hat{s} \xrightarrow{\text{BPSK demapper}} \hat{a}$$

Given:

- a: 0 0 1 0 1 1 0 1
- s: +1 +1 -1 +1 -1 -1 +1 -1
- r: 0.8 0.2 -0.8 1.9 -0.6 0.2 1.3 0.1
- $\hat{s}$: +1 +1 -1 +1 -1 +1 +1 +1
- $\hat{a}$: 0 0 1 0 1 0 0 0

Decoding errors are observed at the 6th and 8th bits.

1.1 Signal-to-Noise Ratio (SNR)

$$\text{SNR} = \frac{\text{signal power}}{\text{noise power}}$$

SNR is unitless and typically reported in decibels:

$$\text{SNR (dB)} = 10 \log_{10}(\text{SNR})$$

1.2 Error Rate

Error rate is the probability that $a \neq \hat{a}$. The Monte Carlo simulation method is used to compute the Bit Error Rate (BER):

$$\text{BER} = \frac{n_e}{N}$$

where n_e is the number of errors, and N is the number of transmitted bits.

1.3 Power Spectral Density (PSD)

$$\text{power} = \text{PSD} \times \text{bandwidth}$$

Time per symbol:

$$T = \frac{1}{\text{bandwidth}}$$

Energy per symbol:

$$E_s = P \cdot T = \frac{P}{\text{bandwidth}}$$

Noise energy:

$$\sigma^2 = \text{noise PSD}$$

SNR in discrete time:

$$\text{SNR} = \frac{E_s}{\sigma^2}$$

In continuous time:

$$\text{SNR} = \frac{\text{signal power}}{\text{noise power}}$$

1.4 BPSK Mapping

$$c_1 = 0 \implies \text{symbol} = -1, \quad c_1 = 1 \implies \text{symbol} = +1$$

Energy per symbol:

$$E_s = \frac{(-1)^2 + (+1)^2}{2} = 1$$

Noise power:

$$\sigma^2$$

SNR:

$$\text{SNR} = \frac{1}{\sigma^2}$$

BER:

$$\text{BER} = P(s = +1) \cdot P(n < -1) + P(s = -1) \cdot P(n > 1)$$

Probability density function of noise:

$$p(n = x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2}{2\sigma^2}}$$

1.5 Uncoded and Coded BPSK

Uncoded BPSK rate:

$$R = 1\,\text{bit/symbol}$$

Coded BPSK rate:

$$R = \frac{k}{n}\,\text{bits/symbol}$$

Coded BPSK achieves the same BER at a lower SNR. BER for uncoded BPSK:

$$\text{BER} = Q\left(\frac{1}{\sigma}\right) = Q\left(\sqrt{\text{SNR}}\right) = Q\left(\sqrt{2 \cdot \frac{E_b}{N_0}}\right)$$

The Q-function is defined as:

$$Q(u) = \frac{1}{\sqrt{2\pi}} \int_u^\infty \exp\left(-\frac{1}{2}x^2\right)\,dx$$

where Q is the Q-function. Signal energy per information bit:

$$E_b = \frac{E_s}{R} = \frac{nE_s}{k}$$

Noise power:

$$\frac{N_0}{2} = \sigma^2$$

Where N_0 (Noise spectral density): The amount of noise power per 1 Hz of bandwidth.
SNR:

$$\text{SNR} = \frac{E_s}{\sigma^2} = 2R \cdot \frac{E_b}{N_0}$$

$$\frac{E_b}{N_0} = \frac{\text{SNR}}{2R}$$

For BPSK, since SNR $= \frac{1}{\sigma^2}$:

$$\frac{E_b}{N_0} = \frac{1}{2R\sigma^2}$$

2 Repetition Code (1:3)

For a (1:3) repetition code:

$$[m_1] \xrightarrow{\text{ENCODER}} [c_1, c_2, c_3] \xrightarrow{\text{BPSK}} S \xrightarrow{\text{adding noise}} S{+}\text{Noise} \xrightarrow{\text{DECODER}} [L_1, L_2, L_3]$$

Goal: Find L_1, L_2, L_3 from m_1.

L_i represents the "belief" that bit $c_i = 1$, computed as a function of r_1, r_2, r_3:

$$L_i = f(r_1, r_2, r_3)$$

where r_i is the intrinsic part, and the rest is extrinsic. The codeword is generated by:

$$[c_1\, c_2\, c_3]^T = G[m_1]^T$$

where G is a 3×1 generator matrix.

2.1 Log-Likelihood Ratio (LLR)

$$P(c_1 = 0|r_1) = \frac{f(r_1|c_1 = 0) \cdot P(c_1 = 0)}{f(r_1)}$$

$$P(c_1 = 1|r_1) = \frac{f(r_1|c_1 = 1) \cdot P(c_1 = 1)}{f(r_1)}$$

Likelihood ratio:

$$\frac{P(c_1 = 1|r_1)}{P(c_1 = 0|r_1)} = \frac{f(r_1|c_1 = 1)}{f(r_1|c_1 = 0)}$$

For BPSK with $a = 1$:

$$c_1 = 0 \implies \text{symbol} = -1, \quad c_1 = 1 \implies \text{symbol} = +1$$

$$r_1 = 1 + N(0, \sigma^2) \quad \text{or} \quad r_1 = -1 + N(0, \sigma^2)$$

$$\frac{P(c_1 = 1|r_1)}{P(c_1 = 0|r_1)} = \frac{\frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(r_1-1)^2}{2\sigma^2}}}{\frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(r_1+1)^2}{2\sigma^2}}} = \exp\left(\frac{2r_1}{\sigma^2}\right)$$

Log-likelihood ratio:

$$l_1 = \log\left(\exp\left(\frac{2r_1}{\sigma^2}\right)\right) = \frac{2r_1}{\sigma^2}$$

Similarly:

$$l_2 = \frac{2r_2}{\sigma^2}, \quad l_3 = \frac{2r_3}{\sigma^2}$$

2.2 Output LLR (SISO Decoder)

$$L_i = \log\left(\frac{P(c_i = 1|r_1, r_2, r_3)}{P(c_i = 0|r_1, r_2, r_3)}\right)$$

$$P(c_1 = 0|r_1, r_2, r_3) = \frac{f(r_1, r_2, r_3|c_1 = 0) \cdot P(c_1 = 0)}{f(r_1, r_2, r_3)}$$

$$P(c_1 = 1|r_1, r_2, r_3) = \frac{f(r_1, r_2, r_3|c_1 = 1) \cdot P(c_1 = 1)}{f(r_1, r_2, r_3)}$$

Assuming $P(c_1 = 0) = P(c_1 = 1)$:

$$\frac{P(c_1 = 1|r_1, r_2, r_3)}{P(c_1 = 0|r_1, r_2, r_3)} = \frac{f(r_1, r_2, r_3|c_1 = 1)}{f(r_1, r_2, r_3|c_1 = 0)}$$

For repetition code:

$$c_1 = 0 \implies \text{symbol vector} = [-1, -1, -1]$$

$$r_1 = -1 + N_1(0, \sigma^2), \quad r_2 = -1 + N_2(0, \sigma^2), \quad r_3 = -1 + N_3(0, \sigma^2)$$

$$c_1 = 1 \implies \text{symbol vector} = [1, 1, 1]$$

$$r_1 = 1 + N_1(0, \sigma^2), \quad r_2 = 1 + N_2(0, \sigma^2), \quad r_3 = 1 + N_3(0, \sigma^2)$$

where N_1, N_2, N_3 are independent Gaussian random variables. Likelihood ratio:

$$\frac{P(c_1 = 1|r_1, r_2, r_3)}{P(c_1 = 0|r_1, r_2, r_3)} = \exp\left(\frac{2}{\sigma^2}(r_1 + r_2 + r_3)\right)$$

Log-likelihood ratio:

$$L_1 = \frac{2}{\sigma^2}(r_1 + r_2 + r_3)$$

Here, r_1 is intrinsic, and r_2, r_3 are extrinsic.

3 (2:3) SPC Code

For a (2:3) Single Parity Check (SPC) code:

$$[m_1, m_2] \xrightarrow{\text{ENCODER}} [c_1, c_2, c_3] \xrightarrow{\text{BPSK}} S \xrightarrow{\text{adding noise}} S\text{+Noise} \xrightarrow{\text{SISO DECODER}} [L_1, L_2, L_3]$$

Goal: Find L_1, L_2, L_3 from $[m_1, m_2]$.

Log-likelihood ratios:

$$l_1 = \log\left(\frac{P(c_1 = 1|r_1)}{P(c_1 = 0|r_1)}\right), \quad l_2 = \log\left(\frac{P(c_2 = 1|r_2)}{P(c_2 = 0|r_2)}\right), \quad l_3 = \log\left(\frac{P(c_3 = 1|r_3)}{P(c_3 = 0|r_3)}\right)$$

Given p_2, p_3, define $p_1 = P(c_1 = 0|r_2, r_3)$. Possible codewords:

c_1	c_2	c_3
0	0	0
0	1	1
1	0	1
1	1	0

Define:

$$l_{\text{ext},1} = \log\left(\frac{1 - p_1}{p_1}\right), \quad l_2 = \log\left(\frac{1 - p_2}{p_2}\right), \quad l_3 = \log\left(\frac{1 - p_3}{p_3}\right)$$

$$p_1 = p_2 p_3 + (1 - p_2)(1 - p_3), \quad 1 - p_1 = p_2(1 - p_3) + (1 - p_2)p_3$$

$$(1 - p_1) - p_1 = p_2((1 - p_3) - p_3) + (1 - p_2)((1 - p_3) - p_3)$$

$$(1 - p_1) - p_1 = -((1 - p_2) - p_2)((1 - p_3) - p_3)$$

$$\frac{(1 - p_1) - p_1}{(1 - p_1) + p_1} = -\frac{(1 - p_2) - p_2}{(1 - p_2) + p_2} \cdot \frac{(1 - p_3) - p_3}{(1 - p_3) + p_3}$$

$$\frac{1 - \frac{p_1}{1-p_1}}{1 + \frac{p_1}{1-p_1}} = -\frac{1 - \frac{p_2}{1-p_2}}{1 + \frac{p_2}{1-p_2}} \cdot \frac{1 - \frac{p_3}{1-p_3}}{1 + \frac{p_3}{1-p_3}}$$

$$\frac{1 - e^{-l_{\text{ext},1}}}{1 + e^{-l_{\text{ext},1}}} = -\frac{1 - e^{-l_2}}{1 + e^{-l_2}} \cdot \frac{1 - e^{-l_3}}{1 + e^{-l_3}}$$

$$\tanh\left(\frac{l_{\text{ext},1}}{2}\right) = -\tanh\left(\frac{l_2}{2}\right) \cdot \tanh\left(\frac{l_3}{2}\right)$$

Comment: If $x > 0$, then $\tanh(x) > 0$, and vice versa for negative x. Thus:

$$\text{sign}(l_{\text{ext},1}) = -\text{sign}(l_2) \cdot \text{sign}(l_3)$$

Taking the absolute value and logarithm:

$$\log\left(\tanh\left(\frac{|l_{\text{ext},1}|}{2}\right)\right) = \log\left(\tanh\left(\frac{|l_2|}{2}\right)\right) + \log\left(\tanh\left(\frac{|l_3|}{2}\right)\right)$$

Define:

$$f(x) = \log\left(\tanh\left(\frac{|x|}{2}\right)\right)$$

$$f(l_{\text{ext},1}) = f(l_2) + f(l_3)$$

This function has the property $f(x) = f^{-1}(x)$. Thus:

$$|l_{\text{ext},1}| = f(f(l_2) + f(l_3))$$

The function $f(x)$ is very high at low values of x and decreases as x increases. Since:

$$f(l_2) + f(l_3) \approx f(\min(|l_2|, |l_3|))$$

the smaller value dominates the sum. Therefore:

$$|l_{\text{ext},1}| = f(f(\min(|l_2|, |l_3|))) = \min(|l_2|, |l_3|)$$

Including signs:

$$l_{\text{ext},1} = -\text{sign}(l_2) \cdot \text{sign}(l_3) \cdot \min(|l_2|, |l_3|)$$

For total LLR:

$$L = l + [l_{\text{ext},1}, l_{\text{ext},2}, l_{\text{ext},3}]$$

For a longer (N-1:N) SPC code:

$$|l_{\text{ext},1}| = -\text{sign}(l_2) \cdot \text{sign}(l_3) \cdot \ldots \cdot \text{sign}(l_N) \cdot \min(|l_2|, |l_3|, \ldots, |l_N|)$$

$$|l_{\text{ext},2}| = -\text{sign}(l_1) \cdot \text{sign}(l_3) \cdot \ldots \cdot \text{sign}(l_N) \cdot \min(|l_1|, |l_3|, \ldots, |l_N|)$$

$$\vdots$$

$$|l_{\text{ext},N}| = -\text{sign}(l_1) \cdot \text{sign}(l_2) \cdot \ldots \cdot \text{sign}(l_{N-1}) \cdot \min(|l_1|, |l_2|, \ldots, |l_{N-1}|)$$

Define:

$$S = -\text{sign}(l_1) \cdot \text{sign}(l_2) \cdot \ldots \cdot \text{sign}(l_N)$$

$$m_1 = \min(|l_1|, |l_2|, \ldots, |l_N|)$$

$$\text{pos} = \arg\min(|l_1|, |l_2|, \ldots, |l_N|)$$

$$m_2 = \min(|l_1|, \ldots, |l_{\text{pos}-1}|, |l_{\text{pos}+1}|, \ldots, |l_N|)$$

$$m_1 = |l_{\text{pos}}|$$

$$|l_{\text{ext,pos}}| = m_2, \quad |l_{\text{ext},i}| = m_1 \quad \text{for} \quad i \neq \text{pos}$$

Total LLR:

$$L = l + l_{\text{ext}}$$

4 Decoding Algorithms

4.1 Hard Decoding

Input: Received Symbols r_i
Output: Final Codeword

1. **Initialize estimated codeword**:
   For each symbol r_i:
   $$\hat{b}_i = \begin{cases} -1 & \text{if } r_i > 0 \\ +1 & \text{otherwise} \end{cases}$$

2. **Construct Tanner Graph**:
   Create Variable Nodes (VN) for each bit in the estimated codeword.
   Create Check Nodes (CN) for each parity-check equation.

3. **Error Detection Loop**:
   Do:

   - For each Check Node CN_j:
     - Compute $\hat{b}_{j,1}^{(t)}$ from connected VNs.
     - Identify potential error positions.
   - For each detected error:
     - Flip the identified erroneous bit.

   While (not converged) and (iteration count $<$ $\max_i iterations$).

4. **Output Final Codeword**:
   Return the reconstructed codeword based on corrected bits.

4.2 Soft Decoding (Min-Sum)

L is a sparse matrix with the same dimensions as the parity-check matrix H.
Initialize:
$$L[N[j,i],j] = r_j \quad \text{for } j = 1, 2, \ldots, n$$

Every nonzero entry of column i in L is replaced by r_i.
Received bits vector: $r = [r_1, r_2, \ldots, r_n]$.

4.2.1 Row Operations

Purpose: To process the rows of the storage matrix L, computing values related to each non-zero entry.
Steps:

- Calculate Min_1 (minimum absolute value) and Min_2 (next higher absolute value) for non-zero entries.

- Update the magnitude of all values, setting the smallest to Min_2 and others to Min_1.
- Determine the sign based on the parity of entries in the row.

4.2.2 Column Operations

Purpose: To process the columns of the storage matrix L, ensuring the beliefs are accurately updated.

Steps:

- Compute a new value for each column.
- Calculate Sum_j as the sum of the corresponding row entry r_j plus the entries in column j.
- Update each entry by subtracting the old entry from the total sum.

4.2.3 Final Decision

After completing the row and column operations, make a final decision on each bit:

$$\text{Decision on bit } j = \begin{cases} 0 & \text{if } Sum_j > 0 \\ 1 & \text{if } Sum_j < 0 \end{cases}$$

More iterations enhance performance, typically requiring about 5 to 8 iterations for optimal results.

5 Performance Analysis with Visualizations

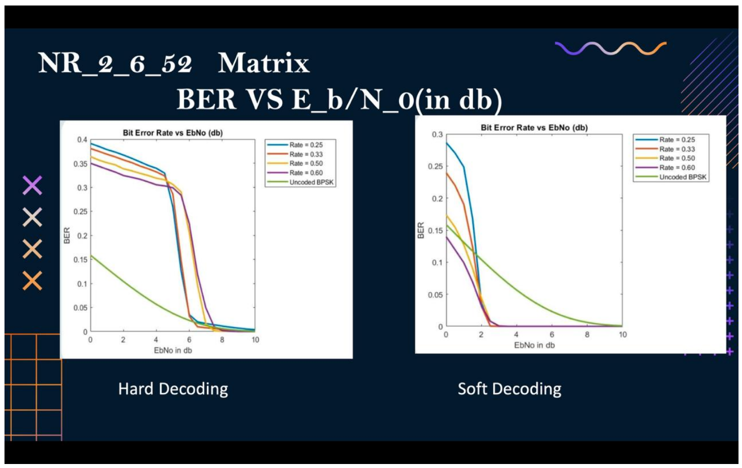

Figure 1: Relationship between E_b/N_0 and BER

Explanation: Higher E_b/N_0 means a cleaner signal, leading to a lower Bit Error Rate (BER). Conversely, lower E_b/N_0 indicates a noisier signal, resulting in a higher BER.

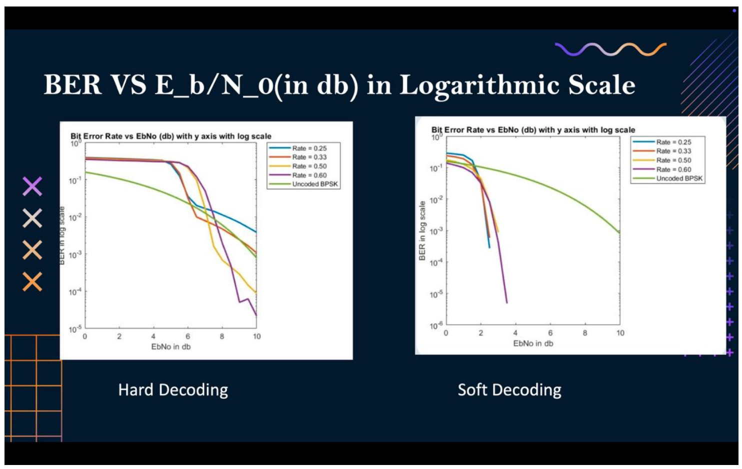

Figure 2: BER on a Logarithmic Scale

Explanation: By converting to a logarithmic scale, we:

- Spread out the BER values over multiple orders of magnitude.
- Make it easier to compare different coding schemes and observe improvements.
- Can clearly see coding gains (e.g., how much lower E_b/N_0 is needed to achieve a target BER).

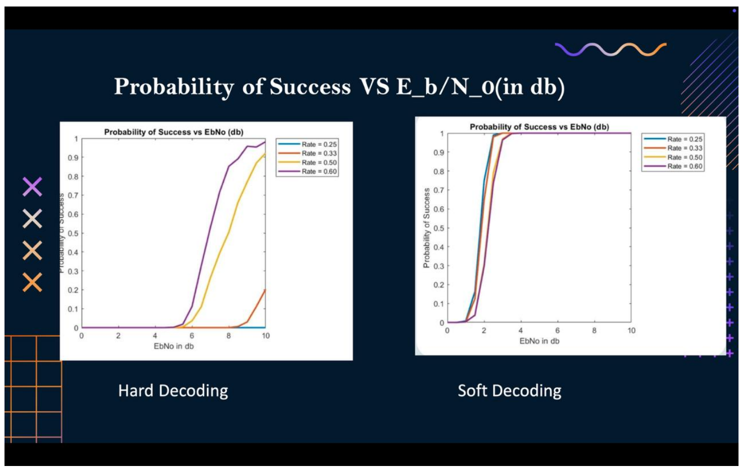

Figure 3: Impact of E_b/N_0 on SNR and Error Probability

Explanation: As the E_b/N_0 (energy per bit to noise power spectral density ratio) increases, the Signal-to-Noise Ratio (SNR) also increases. An increase in SNR leads to a reduction in the error power relative to the signal power, which in turn decreases the probability of error. Consequently, the probability of successful transmission increases. Hard decoding struggles at low SNRs. Higher-rate decoding surprisingly performs better here because hard decoding fails to exploit the extra redundancy effectively.

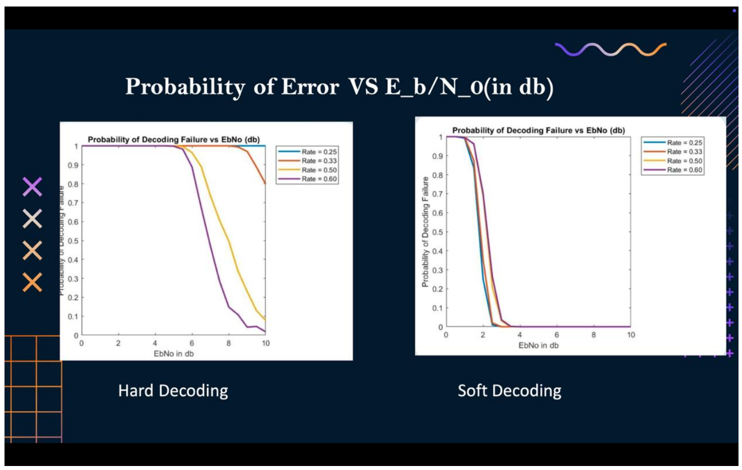

Figure 4: Performance Comparison of Hard and Soft Decoding

Explanation:

Hard Decoding: At low E_b/N_0, the failure rate is 100% (lines start at the top). It only starts to succeed (go down to 0) at higher E_b/N_0, around: So, very strong signals are needed for hard decoding to work well.

Soft Decoding: Same message protection levels (same rates), but the lines drop much earlier. Around 3–4 dB, success is already observed. This means soft decoding needs much less signal strength to work properly.

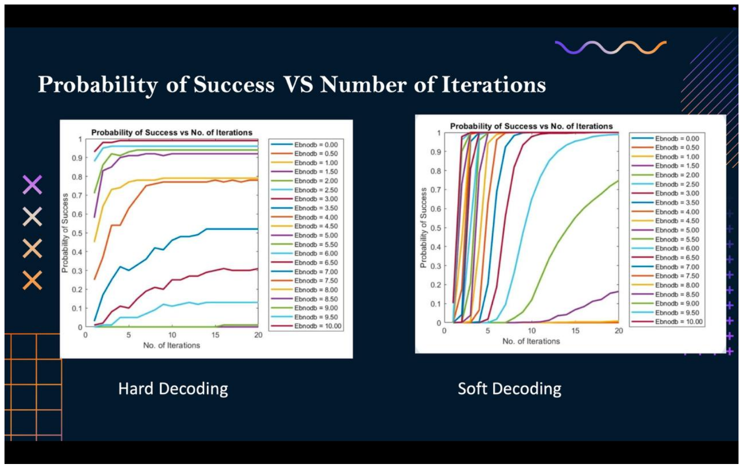

Figure 5: Iteration Impact on Hard and Soft Decoding

Explanation:
Hard Decoding:

- For low E_b/N_0 (bad signal), success stays low even with more iterations.
- For mid E_b/N_0, success gradually improves but is still not great.
- For high E_b/N_0, it reaches near 1 (perfect decoding) after only a few iterations.
- Overall: Hard decoding needs both a good signal and enough iterations to work well.

Soft Decoding:

- Even for lower E_b/N_0 values (bad signal), success improves quickly with just a few iterations.
- For medium to high E_b/N_0, it almost hits 1 (perfect success) within 5–10 iterations.
- It performs much better and faster than hard decoding.

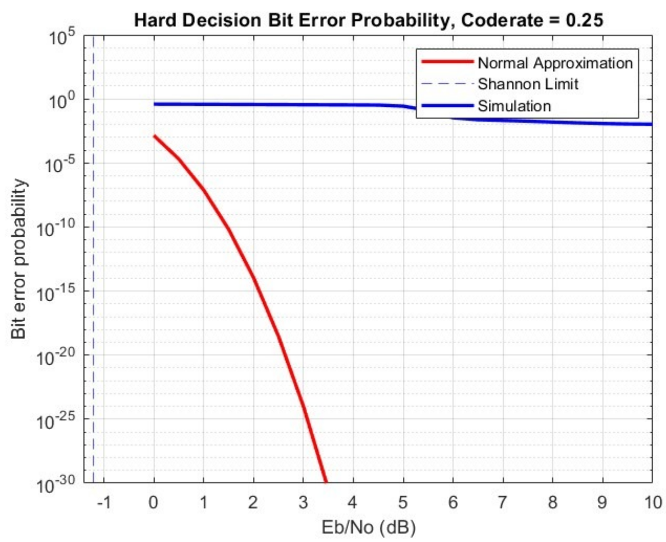

Figure 6: Comparision with Shannon Limit

Explanation: Code rate = 0.25: Low error rate even at low Eb/No. Simulation is close to Shannon Limit. Good error correction but low data rate.

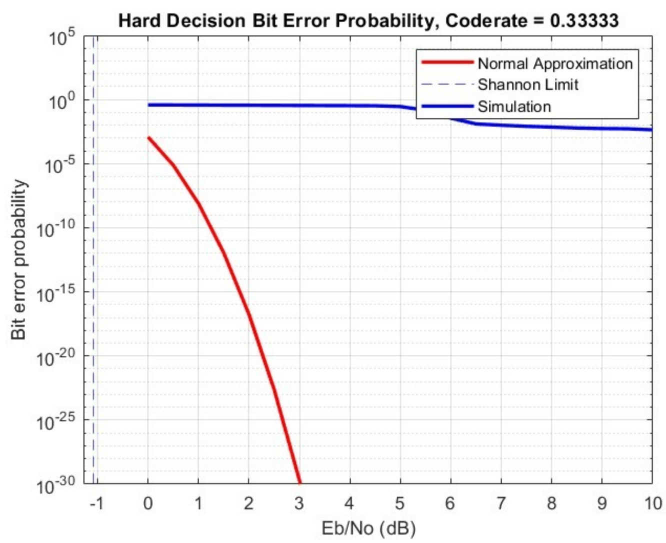

Figure 7: Comparision with Shannon Limit

Explanation: Code rate = 0.3333: Needs slightly higher Eb/No than 0.25. Simulation drifts a bit from Shannon Limit. Balances data rate and error correction.

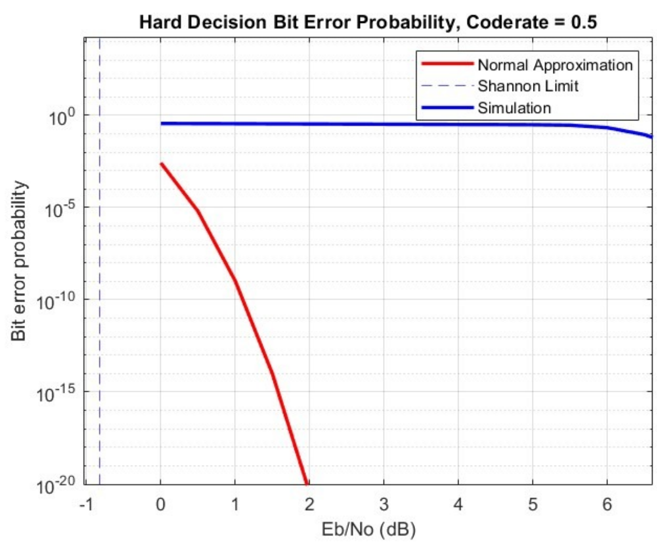

Figure 8: Comparision with Shannon Limit

Explanation: Code rate = 0.5: Higher data rate but more errors. Needs higher Eb/No. Simulation is farther from Shannon Limit.

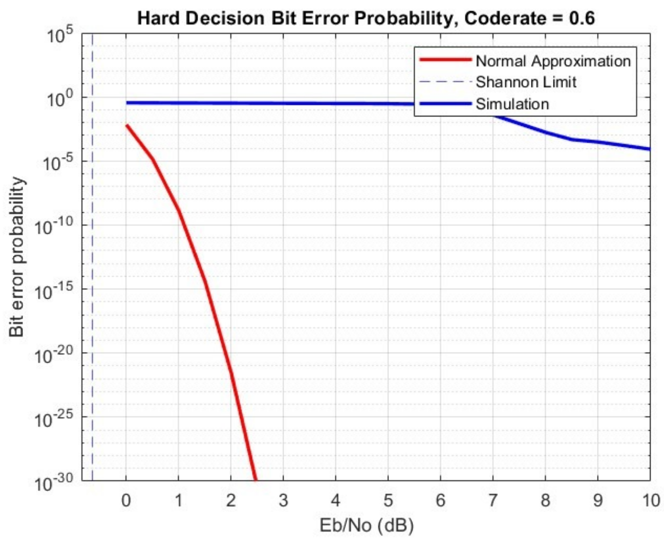

Figure 9: Comparision with Shannon Limit

Explanation: Code rate = 0.6: Sends the most data but worst error performance. Needs high Eb/No. Simulation far from Shannon Limit due to low redundancy.