Erasure Probabilities and Error Correction Assignment

Submitter: Ron Bartov

Background and Introduction

This document provides answers and simulations analysis for the post-interview assignment, which focuses on implementing coding functions and understanding theoretical concepts related to bit-channel erasure probabilities, encoding, channeling, decoding, and frame error rate (FER) analysis.

The assignment is divided into two parts:

- **Part 1**: Focuses on calculating erasure probabilities for bit-channels and analyzing how parameters influence these probabilities.
- **Part 2**: Explores encoding, channel operations, decoding, and their associated error probabilities.

The following sections provide detailed answers, function explanations, and simulations analysis.

Preliminary Knowledge

This section includes all the diagrams and equations from the assignment, serving as the basis for later calculations and analysis.

U-Transform

A transformation from vector $\boldsymbol{x_n}$ to vector $\boldsymbol{u_n}$

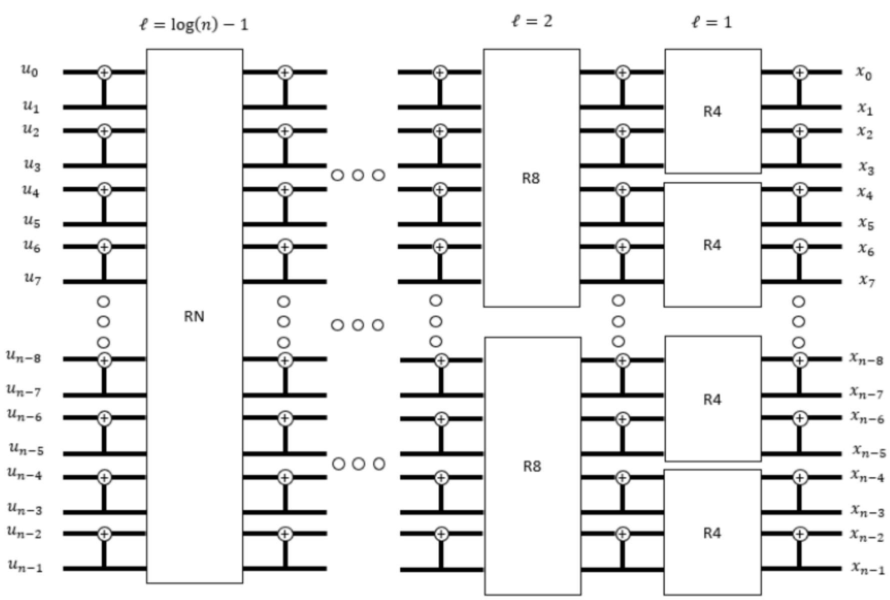


Figure 1: U-transform

Permutation Block Basic Unit for Layer l

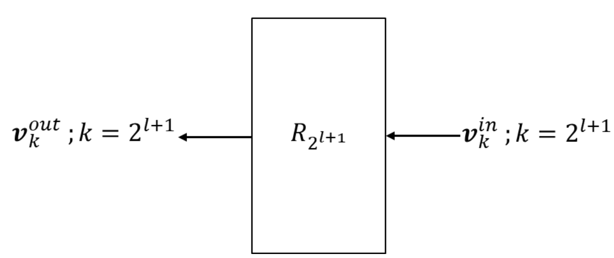


Figure 2: Permutation block basic unit for layer l

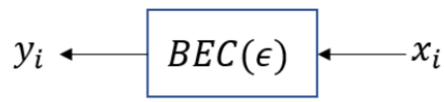
This permutation block performs the following task:

For $j = 0 \dots 2^{\ell+1} - 1$
- If index $j < 2^{\ell}$:
  - $v_{2j}^{\ell} \leftarrow v_j^{\ell}$
- $Else$
  - $v_{2(j-2^{\ell})+1}^{\ell} \leftarrow v_j^{\ell}$

Figure 3: Permutation algorithm for permutation block basic unit

$BEC(\epsilon)$ Channel

A communication channel model where each transmitted bit has a probability ϵ of being erased



$$y_i = \begin{cases} x_i & w.p\ \ 1-\epsilon \\ E & w.p\ \ \epsilon \end{cases}$$

Figure 4: BEC channel

Bit Channel Decoder (BCD) Basic Unit

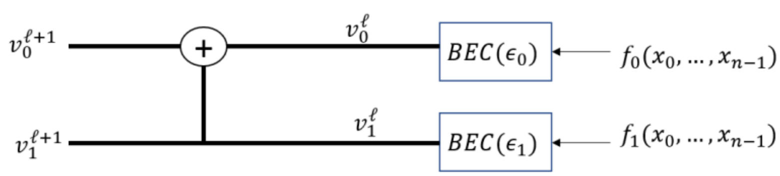


Figure 5: Bit channel decoder basic unit

Satisfies the following effective channel equations for v_j^{l+1}:

For $j = 0$, assuming $\boldsymbol{x}$ is given:

$$\epsilon_0^{\ell+1} = p\left(v_0^{\ell+1} = E \middle| \boldsymbol{x}\right) = \mathbf{1} - (1-\epsilon_0^{\ell})(1-\epsilon_1^{\ell})$$

Equation 1: Erasure probability for upper output of BCD basic unit

For $j = 1$, assuming v_0^{l+1} and $\boldsymbol{x}$ are given:

$$\epsilon_1^{\ell+1} = p\left(v_1^{\ell+1} = E \middle| \boldsymbol{x}, v_0^{\ell+1} given\right) = \epsilon_0^{\ell}\epsilon_1^{\ell}$$

Equation 2: Erasure probability for lower output of BCD basic unit

For $j > 1$, assuming all $v_{j'}^{l+1}, j' < j$ and $\boldsymbol{x}$ are given:

$$\epsilon_j^{\ell+1} = p\left(v_j^{\ell+1} = E \middle| \boldsymbol{x}, v_0^{\ell+1}, \dots, v_{j-1}^{\ell+1}\ given\right)$$

Equation 3: Definition of erasure probability for array elements, that are the outputs of concatenate BCD units, for index greater than one

Encoder-Channel-Decoder (ECD) Model for Input Vector of Length 4

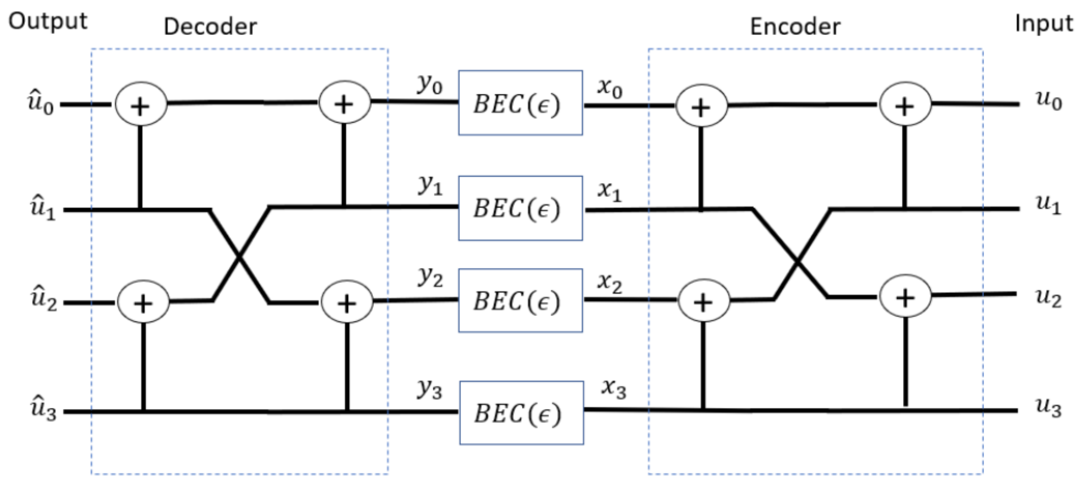


Figure 6: Encoder-Channel-Decoder Model for part 2

Satisfies the following output vector equations:

$$\hat{u}_0 = y_0 + y_1 + y_2 + y_3$$

Equation 4: ECD output 0

$$\hat{u}_1 | \hat{u}_0 = y_2 + y_3 \; \boldsymbol{or} \; \hat{u}_0 + y_0 + y_1$$

Equation 5: ECD output 1

$$\hat{u}_2 | \hat{u}_1, \hat{u}_0 = y_1 + y_3 \; \boldsymbol{or} \; (\hat{u}_0 + \hat{u}_1) + y_0 + y_3 \; \boldsymbol{or} \; \hat{u}_0 + y_0 + y_2 \; \boldsymbol{or} \; y_1 + y_2 + \hat{u}_1$$

Equation 6: ECD output 2

$$\hat{u}_3 | \hat{u}_2, \hat{u}_1, \hat{u}_0 = y_3 \; \boldsymbol{or} \; \hat{u}_1 + y_2 \; \boldsymbol{or} \; \hat{u}_2 + y_1 \; \boldsymbol{or} \; \hat{u}_0 + \hat{u}_1 + y_0 + \hat{u}_2$$

Equation 7: ECD output 3

Part 1: Bit-Channel Erasure Probabilities and Transform Analysis

This part investigates the behavior of bit-channels in a $BEC(\epsilon)$. It examines how parameters such as the erasure probability ϵ and the size of the transform n affect the reliability of bit-channels. Insights are drawn through computations, sorting, and plotting of erasure probabilities.

Question 1: Write a function that computes an array of erasure probabilities

Function Signature:

```
def erasure_probability_recursive_u_transform(n: int, epsilon: int) -> np.ndarray:
```

Functions Used:

- **effective_bit_channels_in_bcd:**
  Implements the transition of an array through a series of blocks of "Bit Channel Decoder Basic Unit".
- **permutation_block:**
  Implements the "Permutation Block".
- **verify_power_of_two:**
  Verify that n is a power of two (used for validation through the simulations

Algorithm:

- **Base Case:**
  If $n = 2$:
  - Initialize the input erasure probabilities as $[\epsilon, \epsilon]$.
  - Apply the "Bit Channel Decoder Basic Unit" to compute the erasure probabilities.
  - Return the computed probabilities.
- **Recursive Case:**
  If $n > 2$:
  - Compute the layer index by $l = \log(n) - 1$.
  - Recursively calculate the erasure probabilities for two vectors with size $\frac{n}{2}$.
  - Concatenate the computed probabilities for the half-size blocks to create the input probabilities vector for the current block size.
  - Apply the permutation operation based on the current layer l to reorder the input probabilities.
  - Apply the "effective_bit_channels_in_bcd" function on the reordered probabilities.
  - Return the computed probabilities.

Example for $N = 8$:

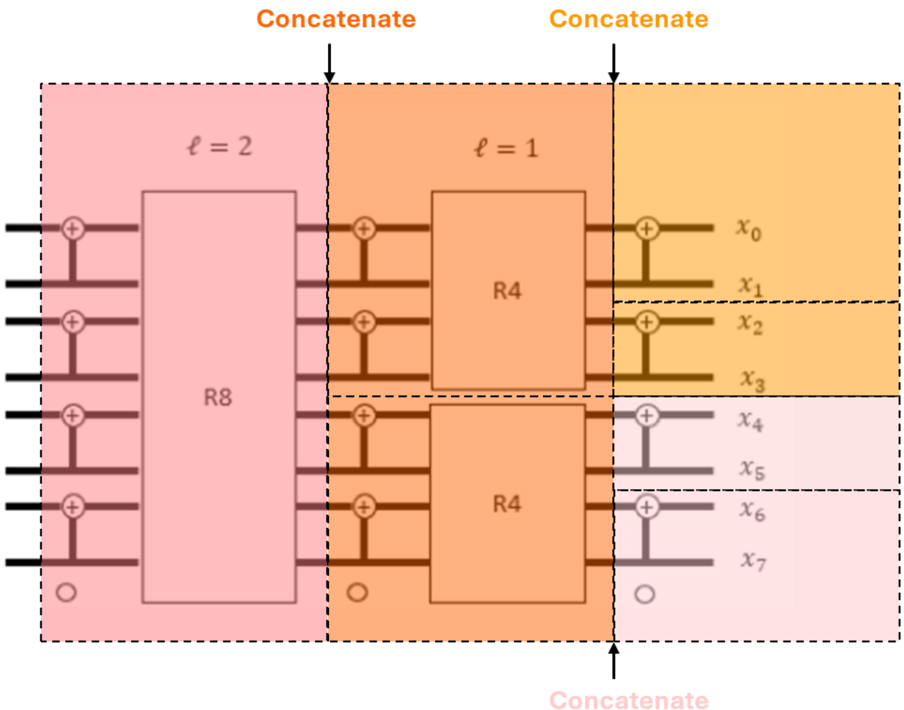


Figure 7: Recursive U-transform example for N=8

Question 2: Simulate erasure probabilities for different input lengths

Simulation Results:

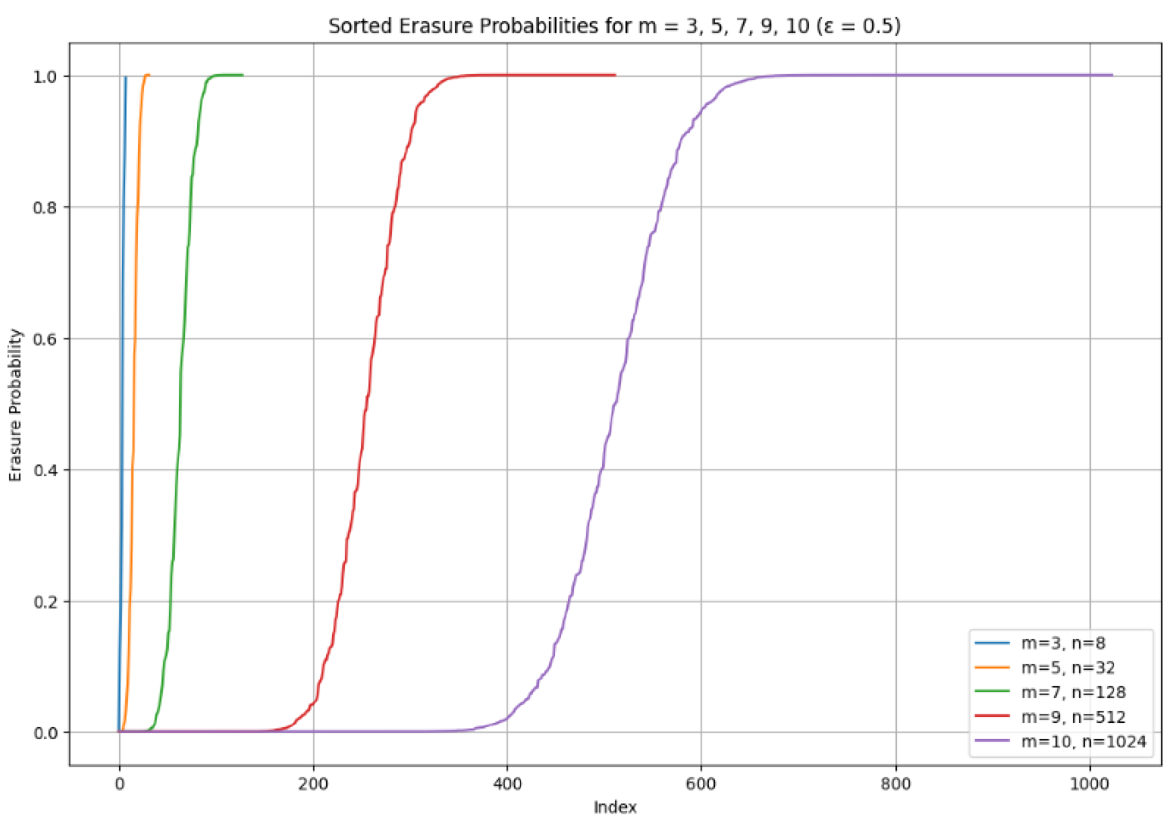


Figure 8: Erasure probabilities simulation for different input lengths and for ϵ = 0.5

Simulation Analysis:

- From the plot, we observe that for a given n the erasure probabilities span the entire range between 0 and 1. This indicates that some output indices exhibit low erasure probabilities, while others have high erasure probabilities.
- The graph's shape (based on the selected axes) resembles a shifted sigmoid curve. Notably, the number of output indices with erasure probabilities less than 0.5 is approximately equal to those with erasure probabilities greater than 0.5.
- As n increases, the graph retains its overall shape, and the erasure probability values are still distributed evenly around 0.5.
- The U-Transform is composed of multiple layers, where each layer concatenates the outputs of Bit Channel Decoder Basic Units and passes them through Permutation Block Basic Units. Within a given layer, based on the equations provided for the Bit Channel Decoder Basic Unit, an input erasure probability vector of length 2 yields two distinct erasure probability outputs: ϵ^2 and $1-(1-\epsilon)^2$. Since these values represent probabilities, the range is constrained to $\epsilon \in [0,1]$.
By plotting these output functions (Figure 9), it becomes evident that for one output, the erasure probability decreases, while for the other, it increases.

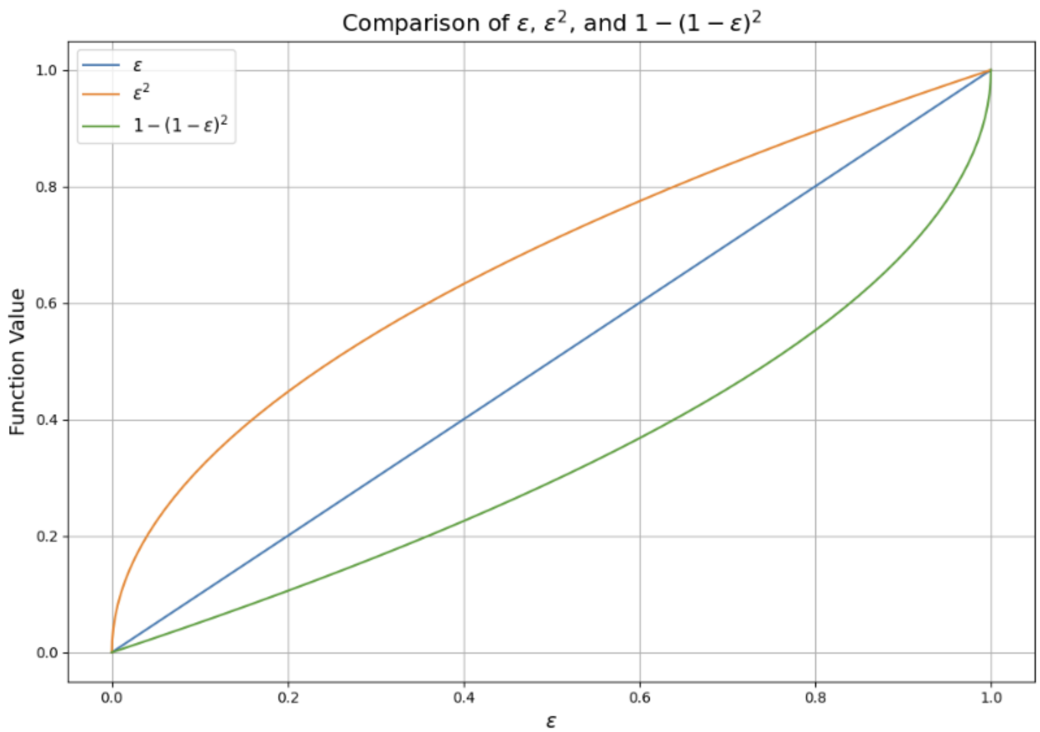


Figure 9: BCD outputs erasure probabilities comparison

In each layer, after passing through the Bit Channel Decoder Basic Units, the new output probabilities are further processed by the Permutation Block Basic Units. These blocks rearrange the probabilities such that the inputs to the next layer's Bit Channel Decoder Basic Units are grouped into pairs of identical values. This arrangement ensures that, in each layer, every Bit Channel Decoder Basic Unit produces one output probability higher than its input and one lower. Over successive layers, the probabilities become more polarized, clustering near 0 and 1. Around the midpoint $\frac{n}{2}$ there is a distinct transition zone. In this region, probabilities gradually shift from being predominantly below 0.5 to predominantly above 0.5, marking the boundary between the two clusters.

Question 3: Simulate erasure probabilities for different ϵ values

Simulation Results:

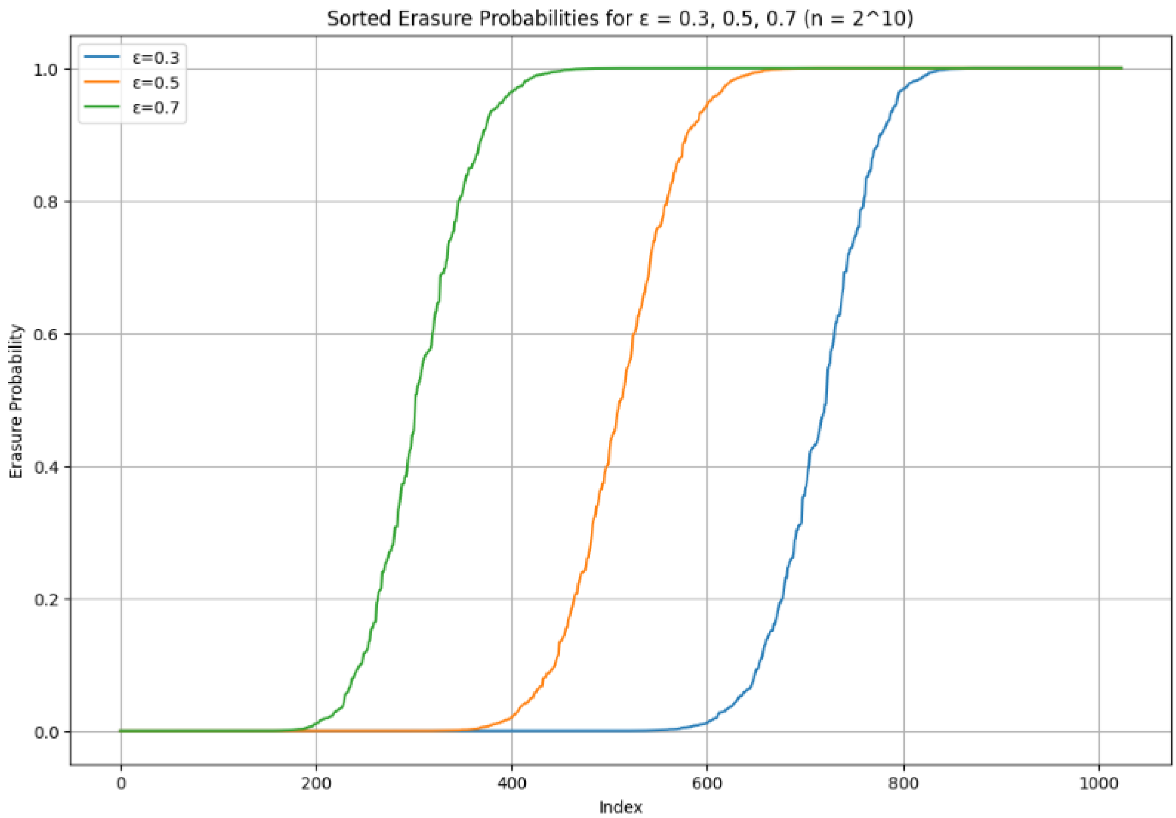


Figure 10: Erasure probabilities simulation for different ϵ values and for n = 1024

Simulation Analysis:

- As observed in Question 2, we again see dominant clusters for the erasure probability values, concentrated near 0 and 1, with a transition zone whose center depends on the chosen initial value of ϵ.
- For a given epsilon value (the initial erasure probability for the input), the boundary point between the clusters (the transition zone) shifts to approximately $n \cdot (1 - \epsilon)$.
- Based on the explanation from Question 2 regarding the U-Transform, it makes sense that the initial erasure probability (either 0.3 or 0.7) influences the output. A starting point of 0.3, for example, results in a greater concentration of output samples with lower erasure probabilities.

Part 2: Encoding, Channeling, and Decoding Analysis

Question 1: Compute the erasure probabilities for all bit channels

- Let's take the Channel-Decoder part from the given Encoder-Channel-Decoder Model and annotate it by marking specific points:

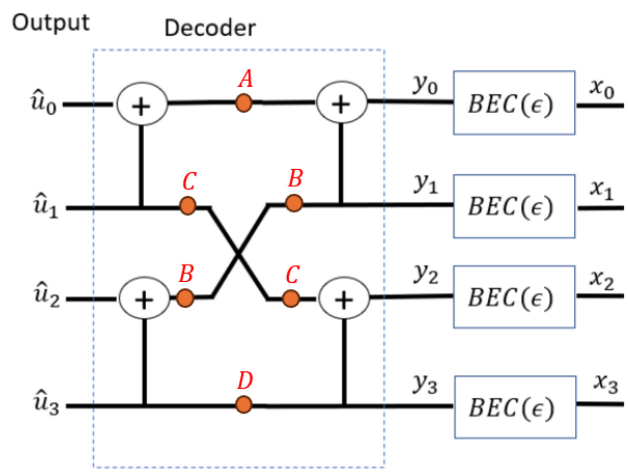


Figure 11: Channel and encoder block from ECD model with additional marks

- From the $BEC(\epsilon)$ channel definition:
  $P_e(y_0) = P_e(y_1) = P_e(y_2) = P_e(y_3) = \epsilon$

- From the Bit Channel Decoder Basic Unit equations:
  $\epsilon_{AC} = P_e(A \mid x) = P_e(C \mid x) = 1 - (1-\epsilon)^2$
  $\epsilon_{BD} = P_e(B \mid x, A) = P_e(D \mid x, C) = \epsilon^2$

- Applying again the same equation for the decoder output:
  $P_e(u_0 \mid x) = 1 - (1-\epsilon_{AC})^2 = 1 - (1-\epsilon)^4$
  $P_e(u_1 \mid x, u_0) = \epsilon_{AC}^2 = (1-(1-\epsilon)^2)^2$
  $P_e(u_2 \mid x, u_0, u_1) = 1 - (1-\epsilon_{BD})^2 = 1 - (1-\epsilon^2)^2$
  $P_e(u_3 \mid x, u_0, u_1, u_2) = \epsilon_{BD}^2 = \epsilon^4$

Question 2: Writing an encoder function

Function Signature:

```
def encode(i: list, info_locs: list, n: int = 4) -> np.ndarray:
```

Functions Used:

- **transform_bits_array:**
  Transforms an input array to an output array using encoder and permutation rules

Algorithm:

- Verify if the length of the input information bits vector i matches the length of the list "info_locs".
- Initializing a vector u with zeros.
- Place the information bits in vector u.
- Transform u to the codeword x using "transform_bits_array"

Question 3: Writing a channel function

Function Signature:

```
def channel(x: np.ndarray, eps: float) -> np.ndarray:
```

Algorithm:

- Generate an array called "random_values" with same length as x and populate it with random samples from a uniform distribution over $[0,1)$.
- Generate a noisy vector y based on the following rule:

$$y_i = \begin{cases} -1, & random\_value_i < \epsilon \\ x_i, & else \end{cases}$$

Note:

The condition in the generation of the noisy vector is based on the fact that by taking a sample s from a uniform distribution over $[0,1)$, the probability of the event $\{s < \epsilon\}$ is $P(\{s < \epsilon\}) = \int_{-\infty}^{\epsilon} f_{s\sim u[0,1]} ds = \int_{0}^{\epsilon} ds = \epsilon$.

Question 4: Writing a decoder function

Function Signature:

```
def decode(y: np.ndarray, info_locs: list) -> bool:
```

Functions Used:

- **transform_by_equations:**
  Applies the set of equations from the "Encode-Channel-Decoder Model" on an input array, only for indexes exist in "info_locs"

Algorithm:

- Initialize a decoder estimated output vector u with zeros
- Apply "transform_by_equations", while passing the noisy vector y, the initialized output vector u, and the indexes locations "info_locs"
- Check if any of the elements in the estimated vector u are erased (represented by -1)
- Returns the following output:

$$Output = \begin{cases} True, & if\ no\ erased\ bits\ are\ found \\ False, & else \end{cases}$$

Question 5: Concatenating encoder-channel-decoder

Concatenation Implementation:

The concatenation is implemented through a function called "encode_channel_decode", that has the following function signature:

```
def encode_channel_decode(i:list, info_locs:list, eps:float, n:int=4) -> bool:
```

Simulation:

The simulation for this question can be executed from the attached python file under the title "Question 5: Concatenating encoder-channel-decoder"

Question 6: Comparison between computed and simulated erasure probability

Note:

For this question and the subsequent ones, the following functions are utilized:

- **run_monte_carlo_simulation:**
  Perform a Monte Carlo simulation to calculate either the simulated erasure probability or FER for given channel combinations and epsilon values.
- **plot_simulation_results:**
  Plot the results of the Monte Carlo simulation and the computed probabilities

Simulation Results:

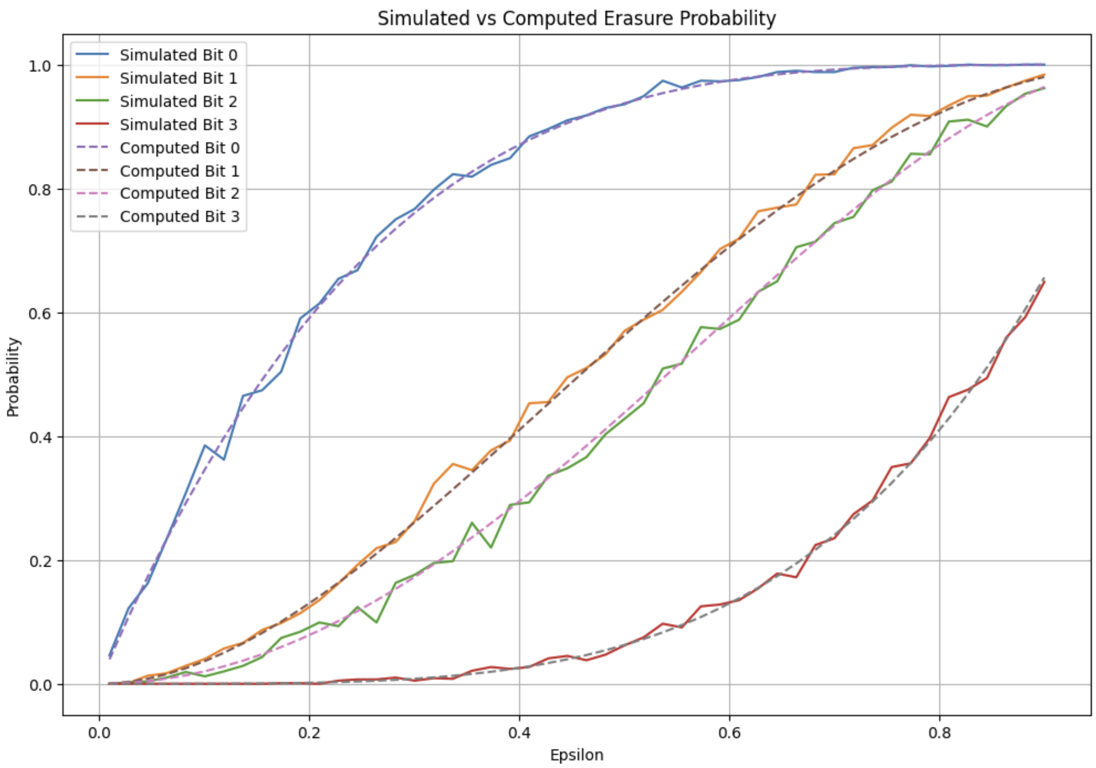


Figure 12: Comparison between computed and simulated erasure probability

- The computed erasure probabilities were calculated using the "computed_erasure_probability" function, which is using the equations from Question 1.
- The simulated erasure probabilities were calculated using the "run_monte_carlo_simulation" function, which performs the following steps:
  - For each channel and epsilon value
    - For each Monte Carlo simulation:
      - Generates a random information bit for the current channel
      - Run the "encode_channel_decode" function
      - Updating the number of False results from the decoder
    - Calculating the simulated erasure probability by $P_e = \frac{\#False\ Results}{\#MC\ Simulations}$
- As can be seen from the Plot, the simulated erasure probabilities are aligned with the computed erasure probabilities.

Question 7: Increasing FER for transmitting in bit channel zero

- Let's define the following sets:
  - $\Omega \equiv \{\omega_j = \{u_j = E\} | \, j = 0,1,2,3\}$
  - $S \equiv \{\bigcap_{j \in J} \omega_j \, \big| \, J \subseteq \{0,1,2,3\}, 0 \in J\}$

- Given a system that uses only bit-channel 0 to transmit information, we get $FER_0 = P(\omega_0)$
- From the dependent of the rest of the u's in u_0, we know that $s \subseteq \omega_0$ for each $s \in S$, and because also ω_0 belong to S, we get
  $\bigcup s \in S = \omega_0$
- Using the definition of FER, and the fact that bit-channel 0 is always in use, we get
  $FER_0 = P(\omega_0) = P\left(\bigcup s \in S\right) = FER_{transmit-in-all-channels}$
- To conclude, we can use all channels without increasing the FER.
- The explanation above was confirmed through a simulation of the described case.

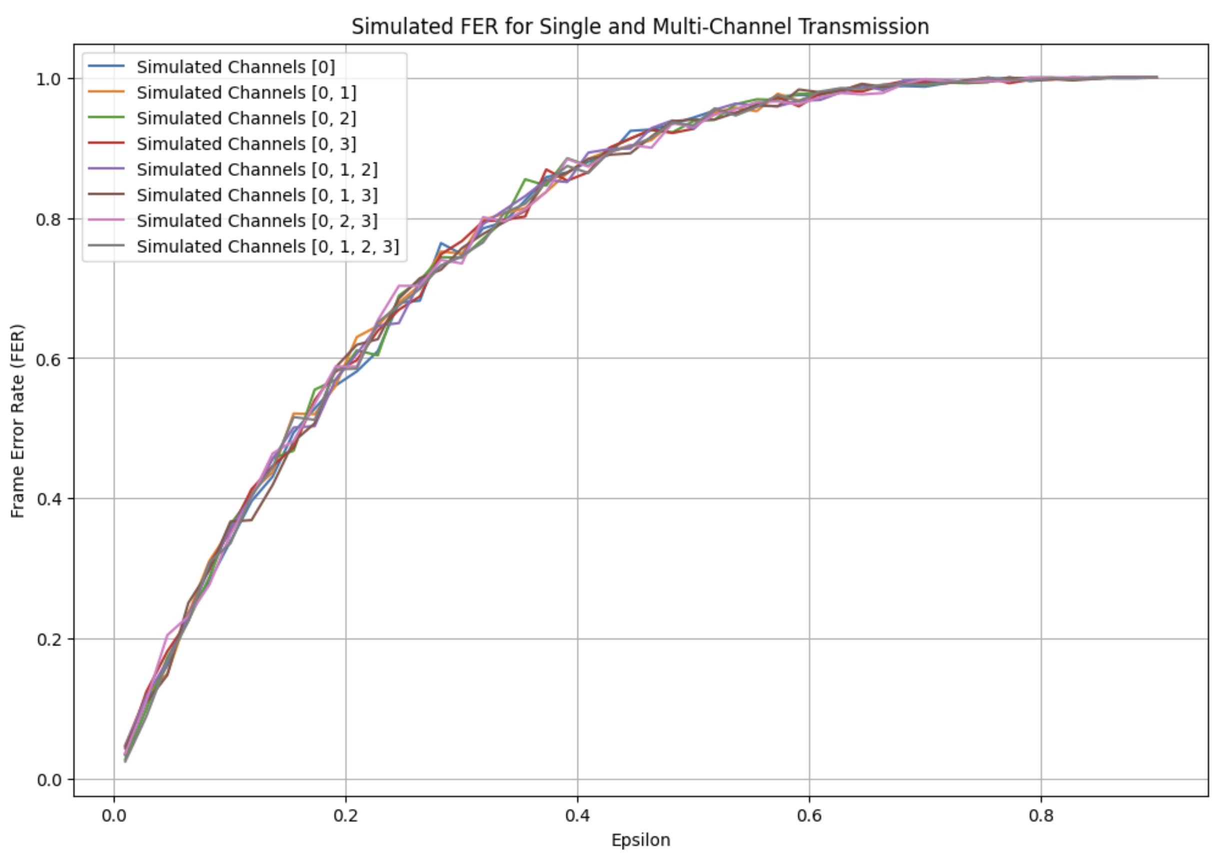


Figure 13: FER simulation for all bit-channels combination that include the zero channel

Question 8: Selecting the best two bit-channels

Our objective is to identify the two bit-channels with the lowest FER. To achieve this, we perform a FER simulation that evaluates all possible combinations of two bit-channels to determine the optimal pair. The simulation results clearly demonstrate that channels 2 and 3 are the best choice.

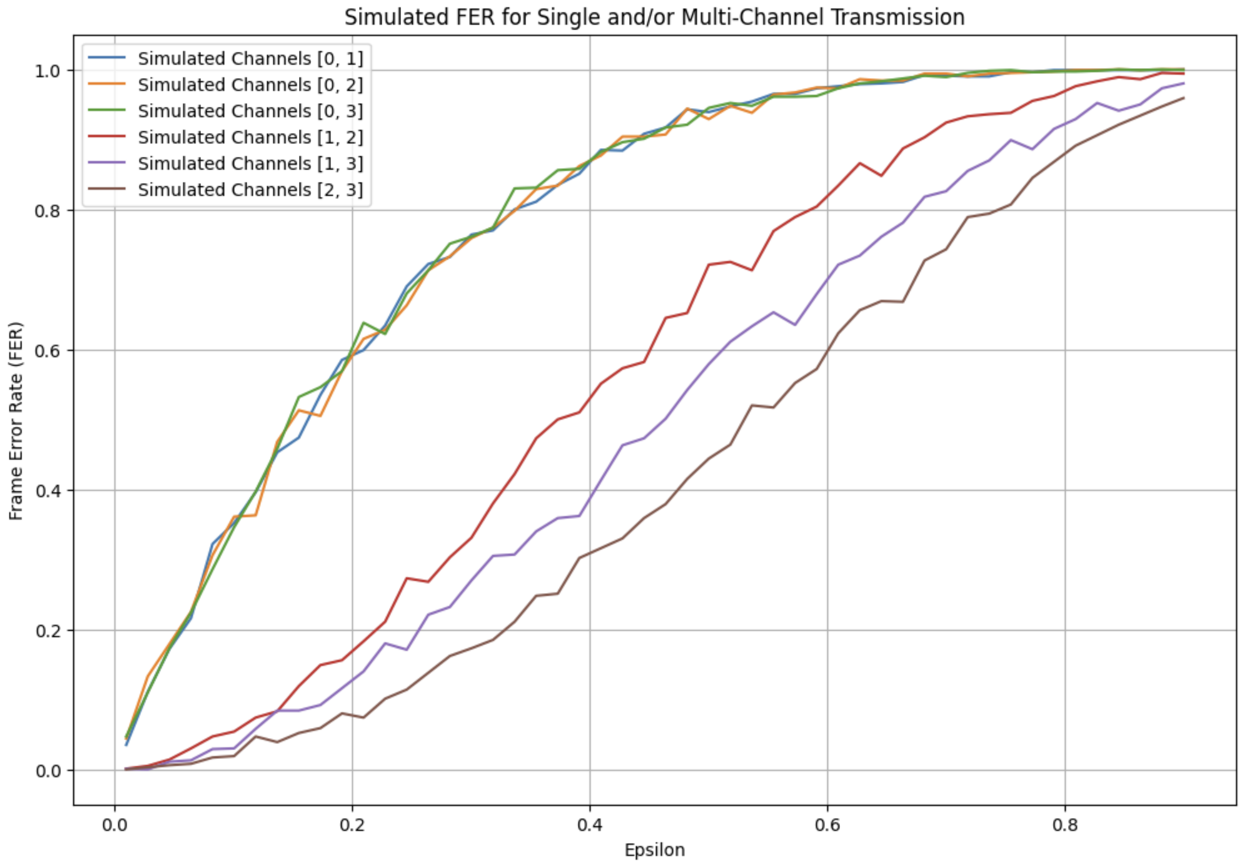


Figure 14: FER simulation for all possible pairs of bit-channel combinations

Question 9: Computing FER for bit-channels 1 and 2

Calculating the FER:

$FER = P(\{u_1 = E\} \cup \{u_2 = E\} | u_0 \; given, x)$

Using the total probability:

$(a) \; FER = P(u_1 = E | u_0 \; given, x) \cdot P(\{u_1 = E\} \cup \{u_2 = E\} | u_0 \; given, u_1 = E, x) +$

$P(u_1 \neq E | u_0 \; given, x) \cdot P(\{u_1 = E\} \cup \{u_2 = E\} | u_0 \; given, u_1 \neq E, x)$

$(b) \; P(\{u_1 = E\} \cup \{u_2 = E\} | u_0 \; given, u_1 = E, x) = P(u_1 = E | u_0, u_1 = E, x) + P(u_2 = E | u_0, u_1 = E, x) -$

$P(u_1 = E | u_0, u_1 = E, x) \cdot P(u_2 = E | u_0, u_1 = E, x) =$

$1 + P(u_2 = E | u_0, u_1 = E, x) - 1 \cdot P(u_2 = E | u_0, u_1 = E, x) = 1$

$(c) \; P(\{u_1 = E\} \cup \{u_2 = E\} | u_0 \; given, u_1 \neq E, x) = P(u_1 = E | u_0, u_1 \neq E, x) + P(u_2 = E | u_0, u_1 \neq E, x) -$

$P(u_1 = E|u_0, u_1 \neq E, x) \cdot P(u_2 = E|u_0, u_1 \neq E, x) =$

$0 + P(u_2 = E|u_0, u_1 \neq E, x) - 0 \cdot P(u_2 = E|u_0, u_1 \neq E, x) = P(u_2 = E|u_0, u_1 \neq E, x)$

From the "Encoder-Channel-Decoder Model" equation for u_2:

$P(u_2 = E|u_0, u_1 \neq E, x) = P_e(y_1 + y_3) \cdot P_e(u_0 + y_0 + y_3) \cdot P_e(u_0 + y_0 + y_2) \cdot P_e(y_1 + y_2)$

Using again the rule $P(A \cup B) = P(A) + P(B) - P(A \cap B)$ and the known erasure probabilities from Question 1:

$P_e(y_1 + y_3) = P_e(y_1 + y_2) = P_e(u_0 + y_0 + y_3) = P_e(u_0 + y_0 + y_2) = 1 - (1 - \epsilon)^2$

$\rightarrow (d)$ $P(\{u_1 = E\} \cup \{u_2 = E\}|u_0\ given, u_1 \neq E, x) = P(u_2 = E|u_0, u_1 \neq E, x) = (1 - (1 - \epsilon)^2)^4$

Substituting (b) and (d) into (a) and using the equation for $P(u_1 \neq E|u_0\ given, x)$:

$FER = P(u_1 = E|u_0\ given, x) + P(u_1 \neq E|u_0\ given, x) \cdot P(\{u_1 = E\} \cup \{u_2 = E\}|u_0\ given, u_1 \neq E, x) =$

$\mathbf{(1 - (1 - \epsilon)^2)^2 + \left(1 - (1 - (1 - \epsilon)^2)^2\right) \cdot (1 - (1 - \epsilon)^2)^4}$

Simulation Results:

The simulation results show that the simulated FER aligns closely with the calculated FER curve on the plot.

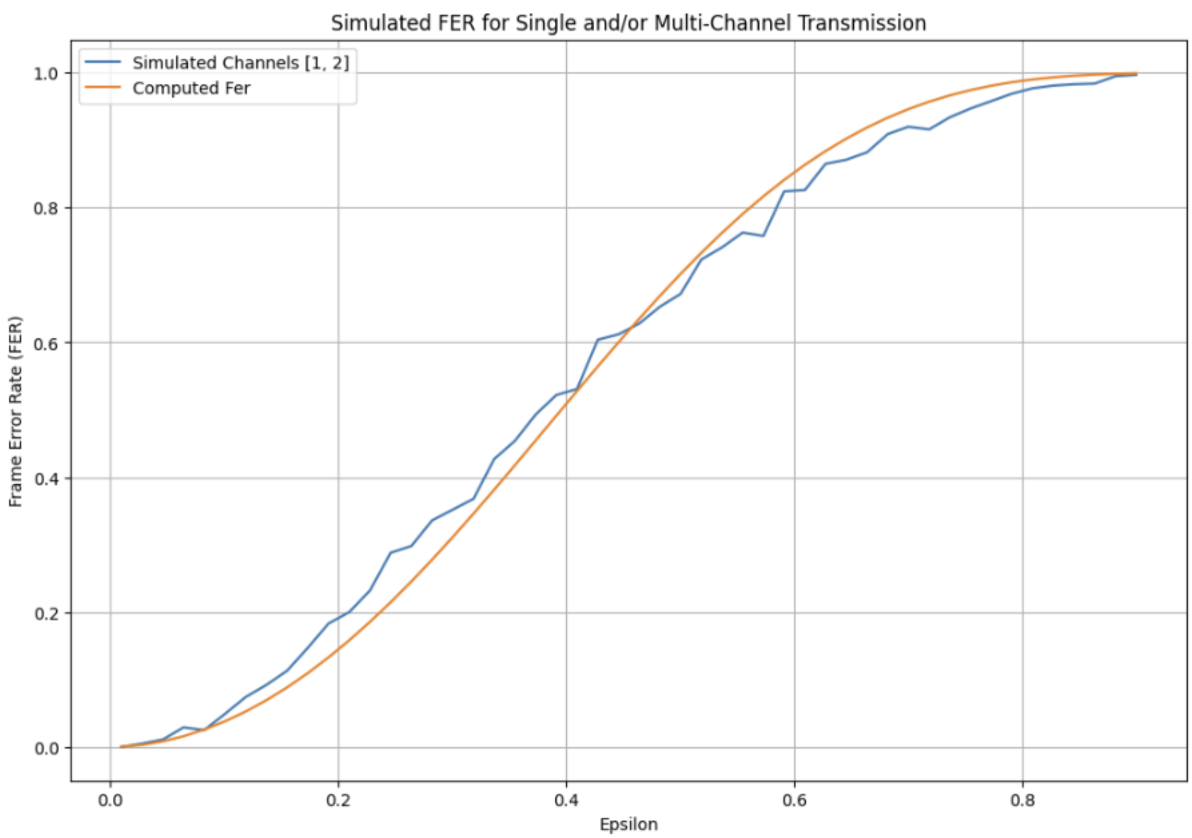


Figure 15: FER simulation for bit-channels 1 and 2