

# CT 111
# Introduction to Communication System

Group Id = 2, Project Team Id = 3

# Group Members

Shashank Didwania
202001078

Kartik Dangi
202001079

Abhimanyu Negi
202001080

Ronit Jain
202001081

Yash Chauhan
202001082


Vedant Patel
202001083

Krishi Patel
202001084

Sumit Vaniya
202001085

Shivam Kansagara
202001086

# Honor Code

We as a group 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.

Signed by all Group Members

# Technical Introduction

- We build the models of
  - the binary erasure channel (BEC(p)) with the erasure probability of p
  - the binary symmetric channel (BSC(p)) with bit flipping probability of p.
- Performance (expressed as:
  - convergence of the algorithm
  - the probability of decoding success)

of the LDPC coding scheme for different values of p for the three LDPC H matrices given to us for

- Hard Decision Decoding for BEC(p)
- Soft Decision Decoding for BEC(p)
- Hard Decision Decoding for BSC(p)
- Soft Decision Decoding for BSC(p)

# Team Contributions

| Name | Student Id | Contribution |
|---|---|---|
| Shashank Didwania | 202001078 | Logical implementation BSC SDD and BEC SDD Channel |
| Kartik Dangi | 202001079 | Code implementation of HDD BEC and SDD BEC and Monte Carlo Simulations |
| Abhimanyu Negi | 202001080 | Code implementation of BSC SDD, BSC HDD and graph analysis of BSC HDD and SDD |
| Ronit Jain | 202001081 | Code implementation of HDD BEC and SDD BEC and graph analysis of HDD BEC |

| | | |
|---|---|---|
| Yash Chauhan | 202001082 | Logical implementation BSC HDD and BEC HDD channel |
| Vedant Patel | 202001083 | Code implementation of BSC SDD, BSC HDD and graph analysis of SDD BEC |
| Krishi Patel | 202001084 | Helped in making the Presentation |
| Sumit Vaniya | 202001085 | Logical implementation BSC HDD and BEC HDD Channel |
| Shivam Kansagara | 202001086 | Logical implementation BSC SDD and BEC SDD Channel |

# MVP

We unanimously chose our MVP for their relentless contribution towards making this project.

## Abhimanyu Negi

# Table of Contents

# Learnings from Project

# Learnings From the Project

## Python Learnings

Learnt use of libraries such as numpy, matplotlib, scipy

## Study

Practical implementation of message passing algorithm in iterative decoding

## Analysis

Analysing the codes and checking their outputs

## Implementation of Codes

Tanner graph implementation, hard and soft decoders implementation for H matrices

## Team Work

## Time Management

## Coordination

1.

# BEC Hard Decision Decoding

# BEC Hard Decision Decoding

First we traverse through the H matrix and store the positions of the 1's present in the H matrix into a matrix (here R) which stores the position of ones in rows of the H matrix. Similarly we do it for columns and save them in a matrix (here C).

We pass the message through the noise channel and store the message.

We traverse through the columns of H matrix and for each of it we traverse through C. The received message bit which is corresponding to the column is saved in another matrix (A).

We do majority decoding on each column of H matrix for the CN's which are connected to their respective VN's.

We traverse through the rows of H and for each of it we traverse through R. We take modulo 2 sum for each of the connected VN and assign it to the matrix A. While taking the modulo 2 sum we do not take in consideration the VN to which we are passing. Also if the bit is an erasure we do not assign it to the matrix A.

Finally we traverse through the columns of H matrix and perform majority decoding.

We repeat all of the steps until the result of two consecutive iterations is same or if we exceed the limit of the iterations.

2.

# BSC Hard Decision Decoding

# BSC Hard Decision Decoding

First we traverse through the H matrix and store the positions of the 1's present in the H matrix into a matrix (here R) which stores the position of ones in rows of the H matrix. Similarly we do it for columns and save them in a matrix (here C).

We pass the message through the noise channel and store the message.

We then traverse through each column of H and C. The received message bit which is corresponding to the column is saved in another matrix (A).

After that we perform majority decoding on each column of H matrix and for the CN's which are connected to their respective VN's.

One by one we traverse through rows of H and the matrix R and take modulo 2 sum for each of the connected VN except for the one we are sending message to and assign the value to the matrix A. This helps in passing message to VN's

Finally we traverse again through the columns of H matrix and perform majority decoding. While majority decoding if we get equal number of 0's and 1's we assign randomly any one of the two.

We repeat all of the steps until the result of two consecutive iterations is same or if we exceed the limit of the iterations.

3.

# Monte - Carlo Simulations

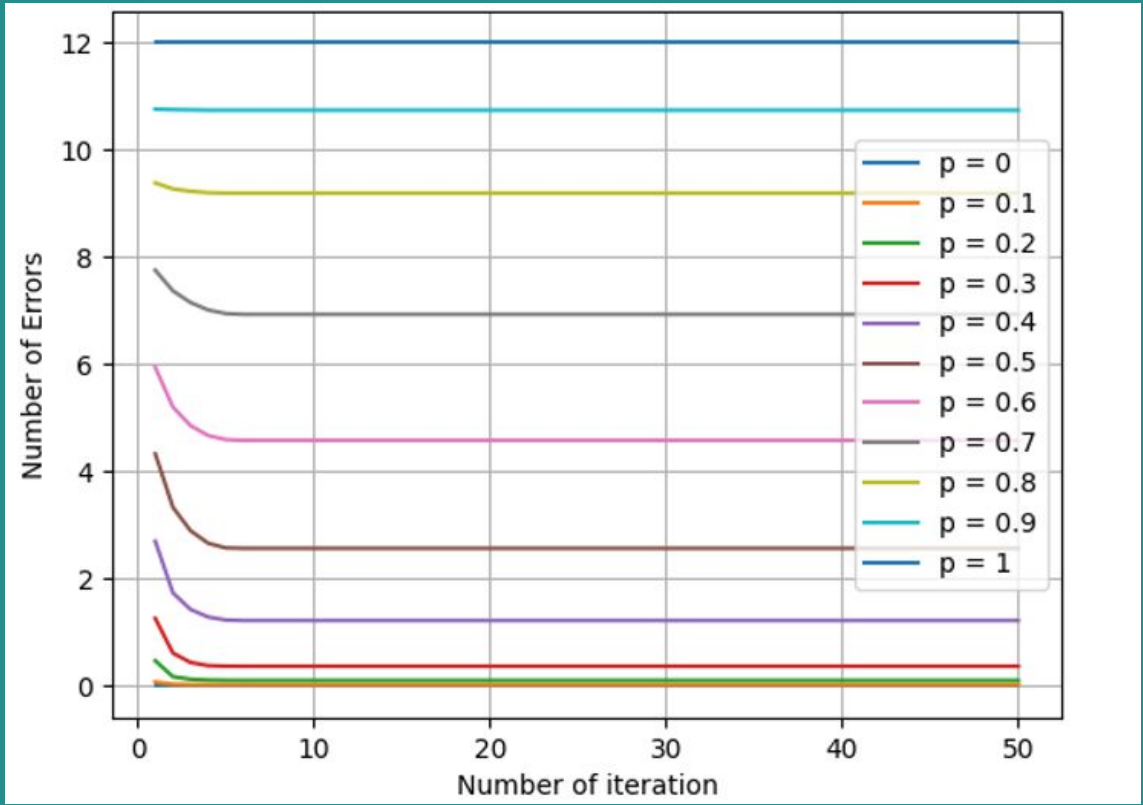
# Monte - Carlo Simulations

- For  H matrix 1 and H matrix 2 we ran the Monte-Carlo simulations for 100 times.
- We ran it for only 100 simulations because our program takes around 2-3 hours to generate the result with these simulations.
- For the 9*12 H Matrix all the graphs were generated using 10000 Monte-Carlo simulations which helps to gain higher accuracy.

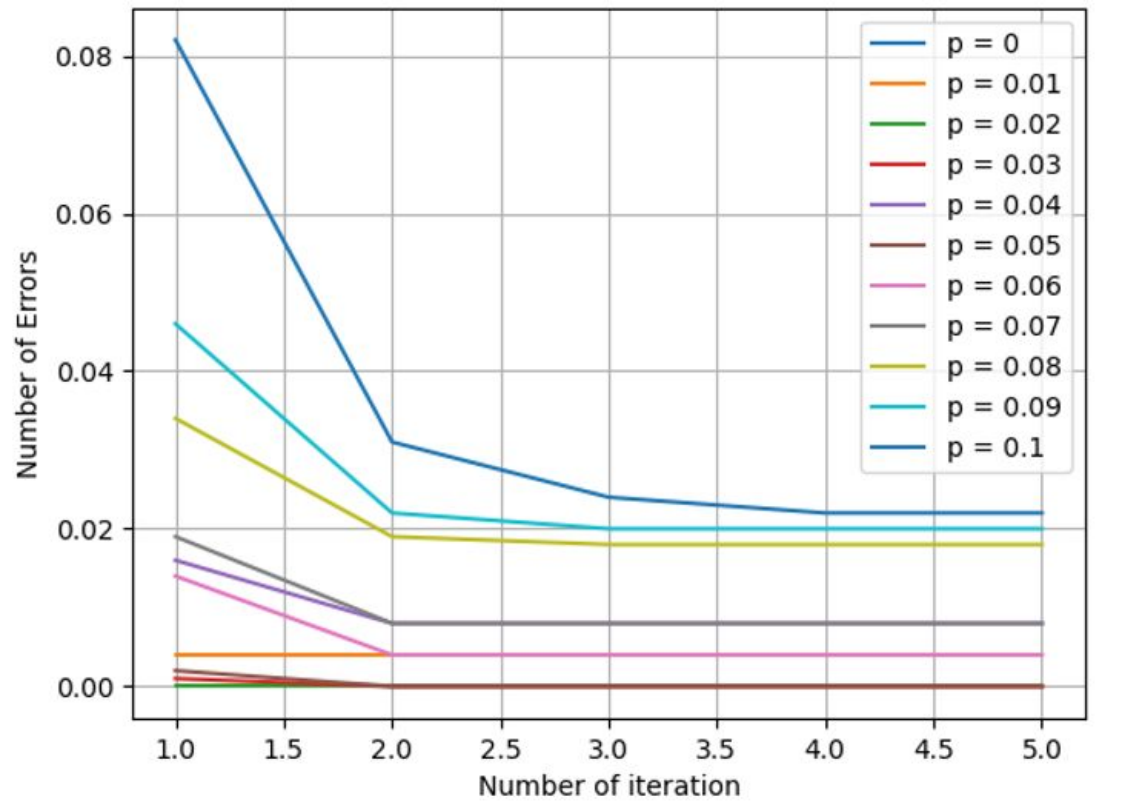
4.

# Graphs

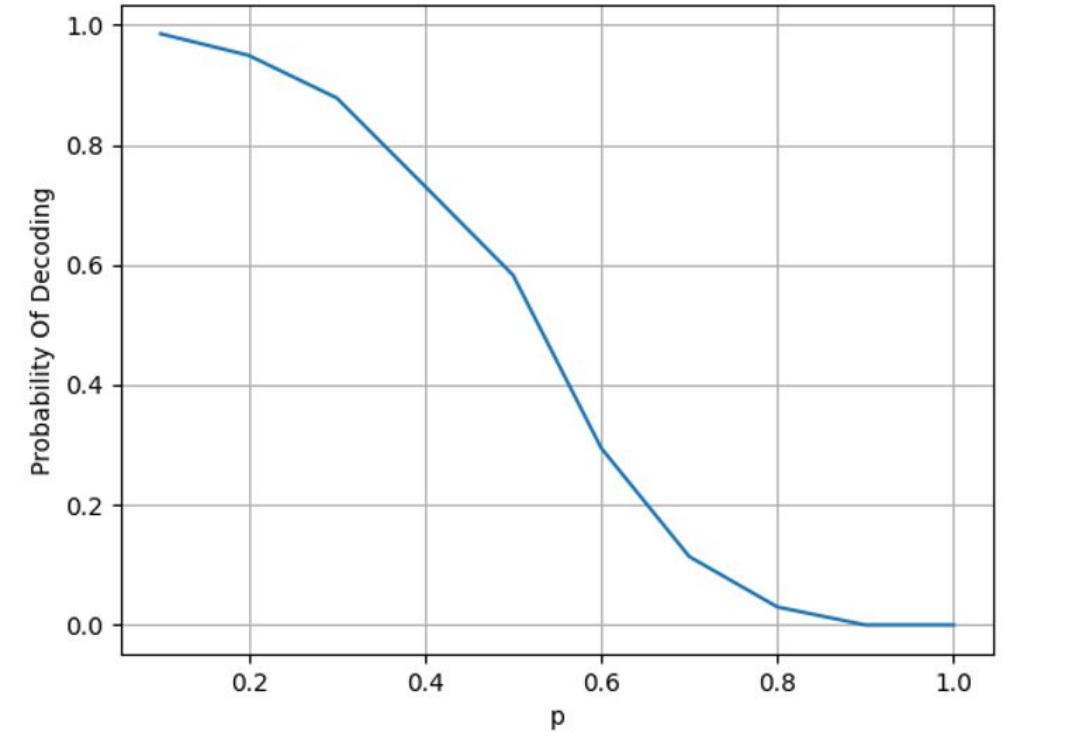
# HDD BEC 9*12 Matrix Error v/s Iteration



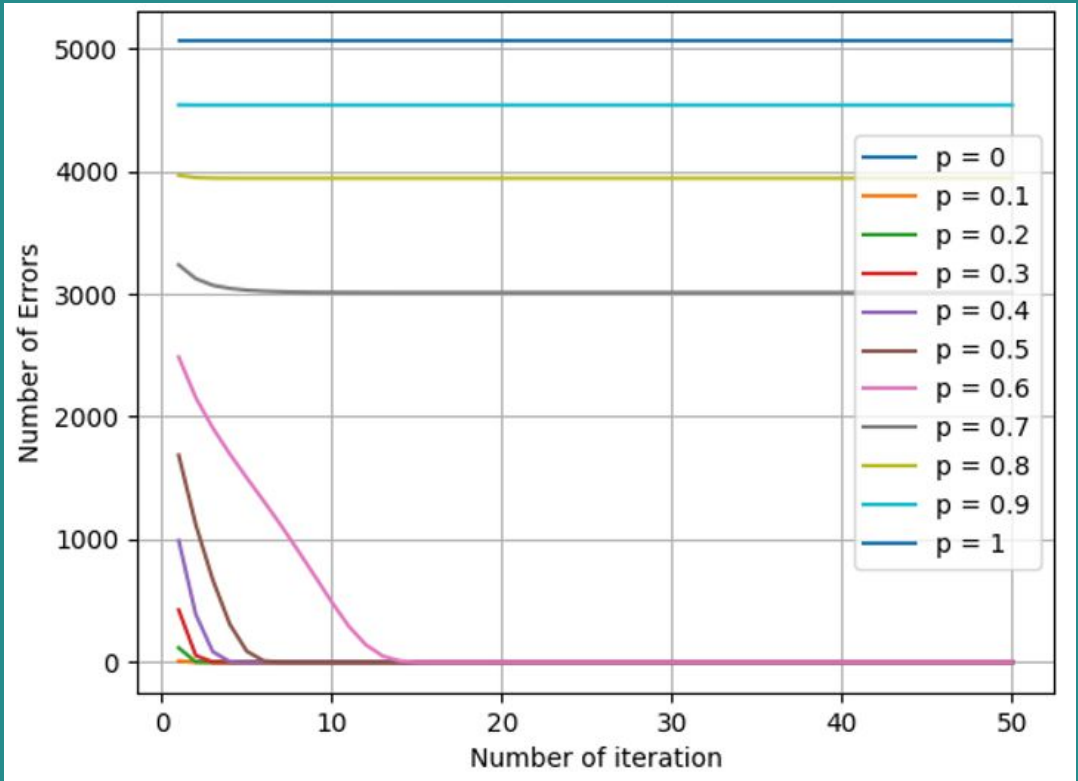
# HDD BEC 9*12 Matrix Error v/s Iteration



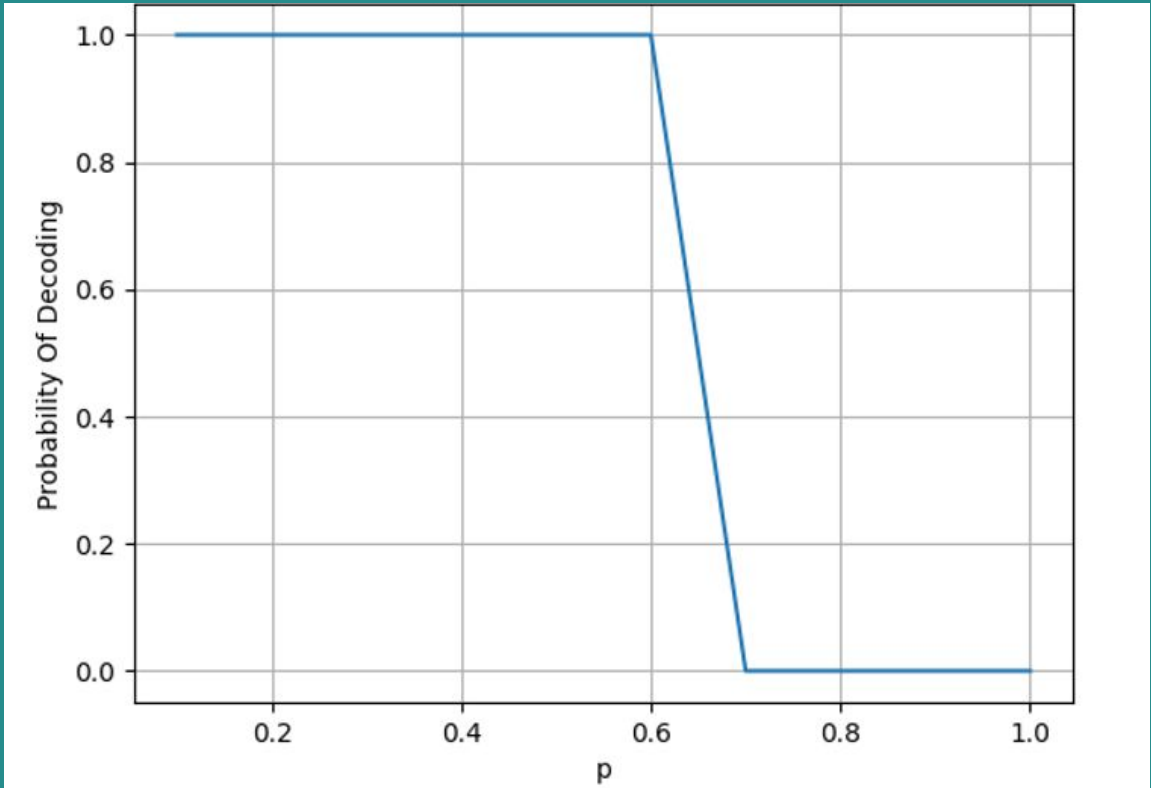
# HDD BEC 9*12 Matrix Probability of Decoding



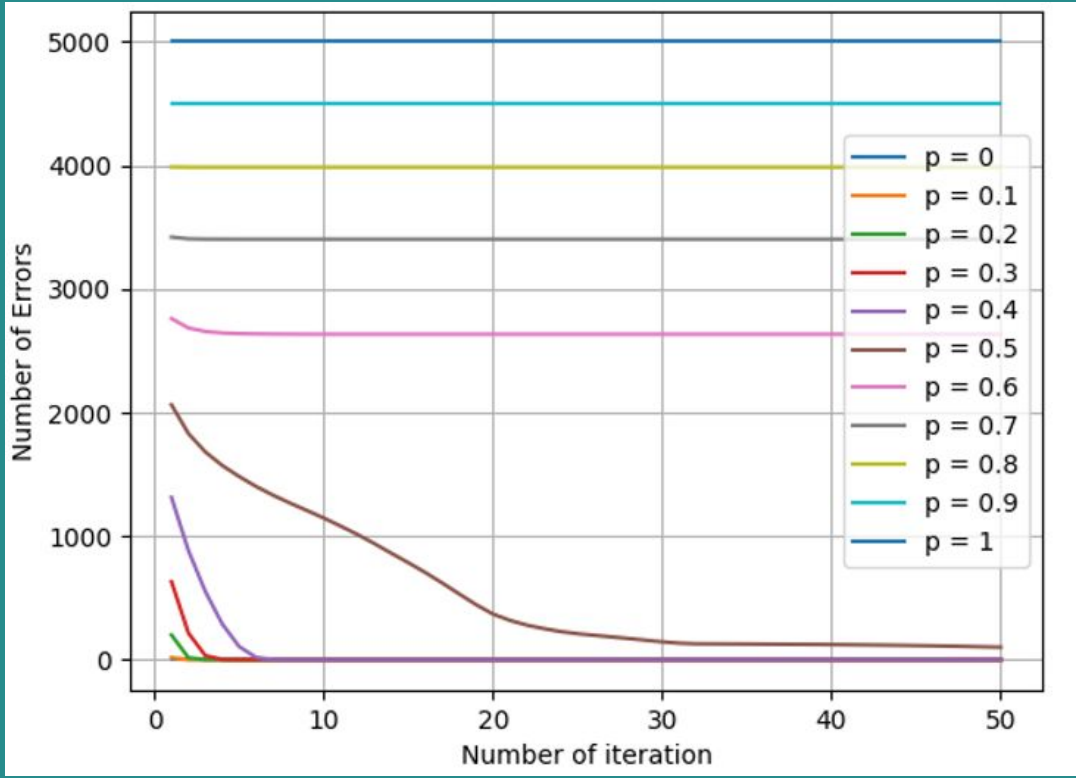
# HDD BEC H Matrix Error v/s Iteration



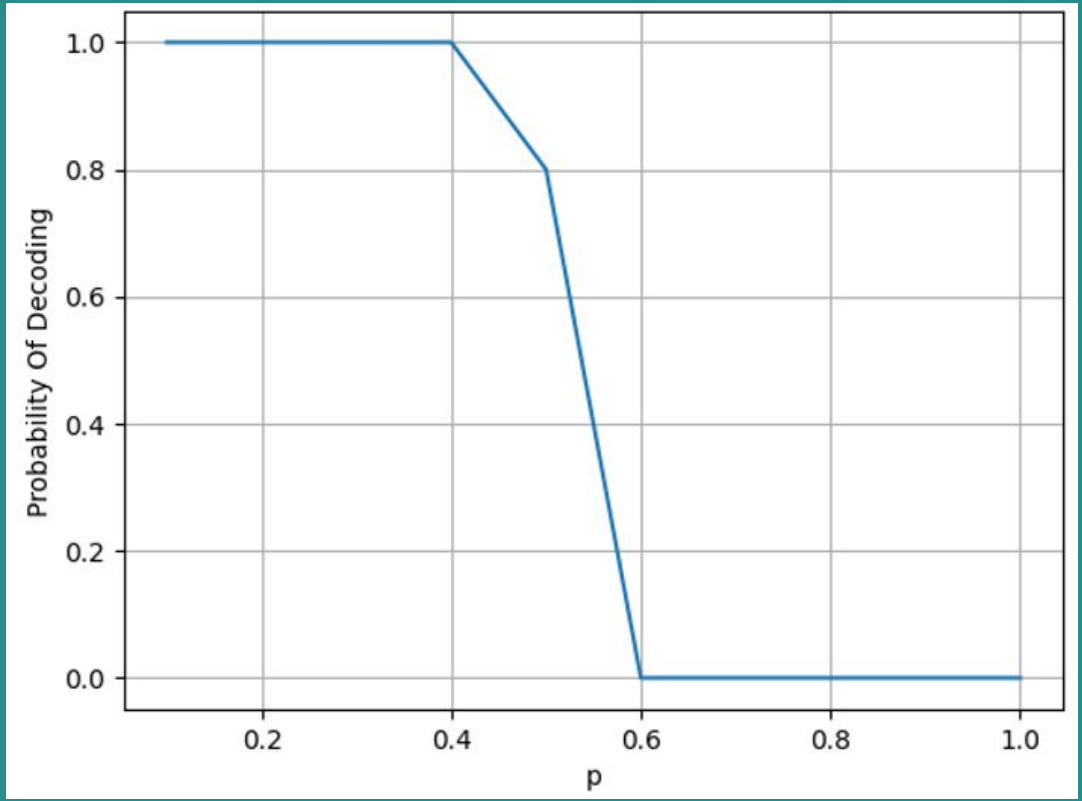
# HDD BEC H Matrix Probability of Decoding



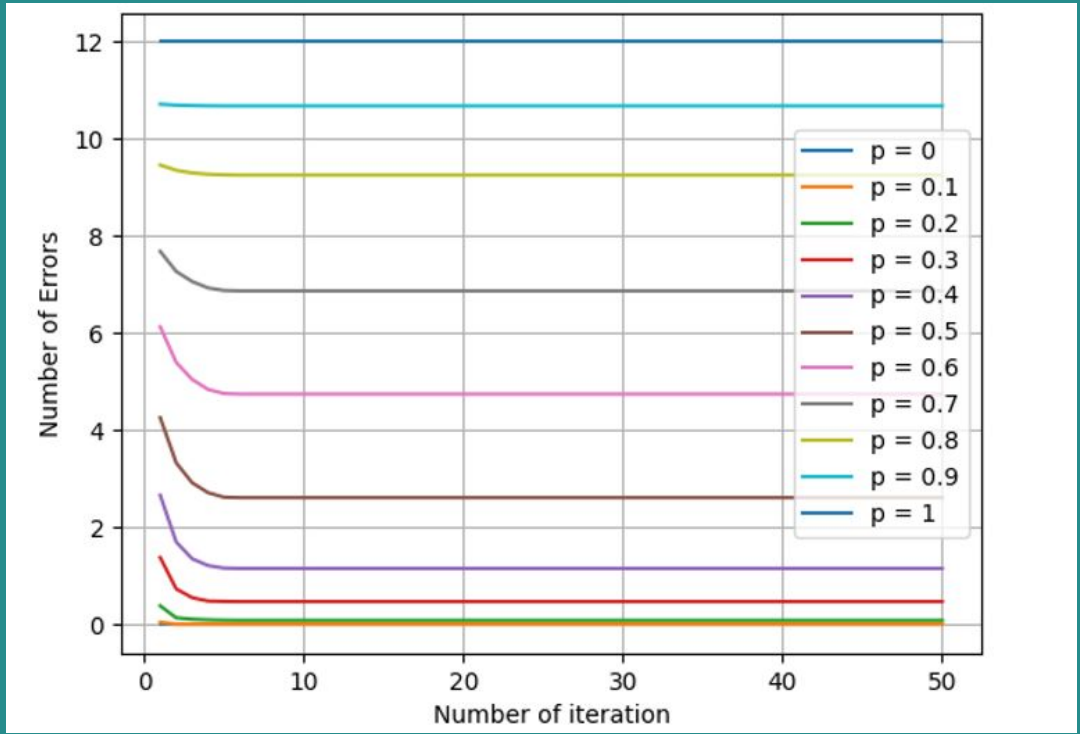
# HDD BEC H Matrix 2
# Error v/s Iteration



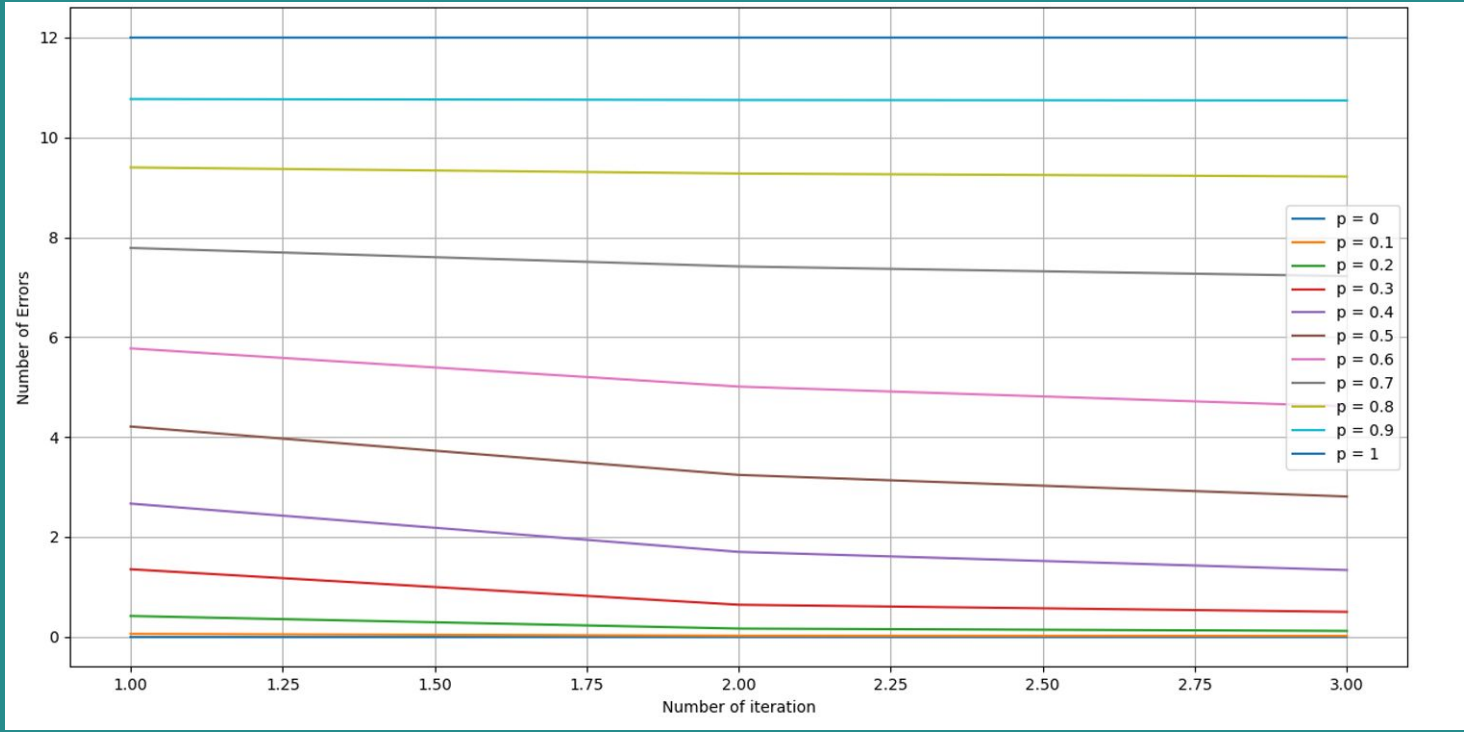
# HDD BEC H Matrix 2
# Probability of Decoding



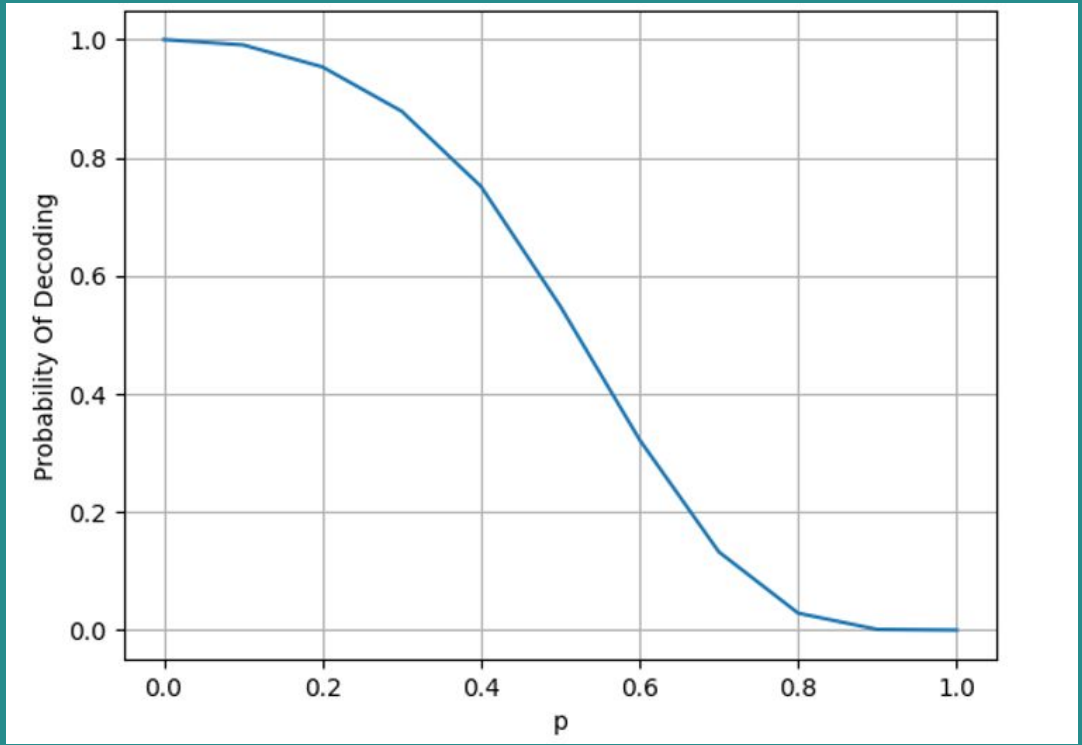
# SDD BEC 9*12 Matrix Error v/s Iteration



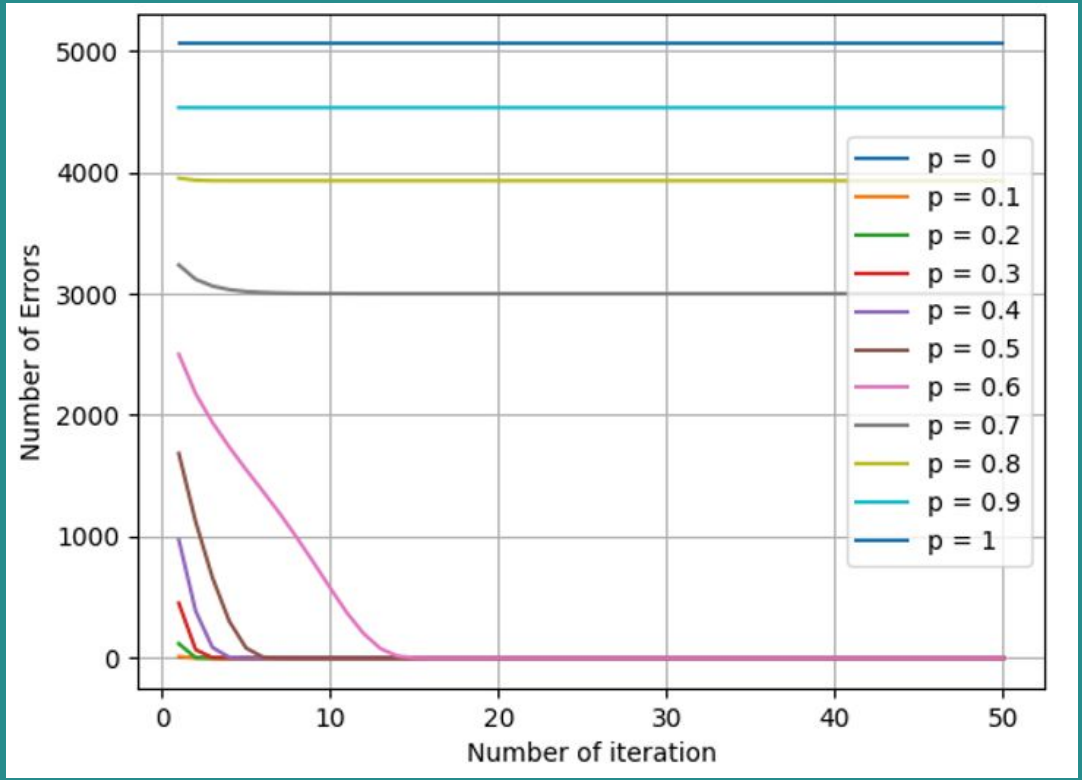
# SDD BEC 9*12 Matrix Error v/s Iteration



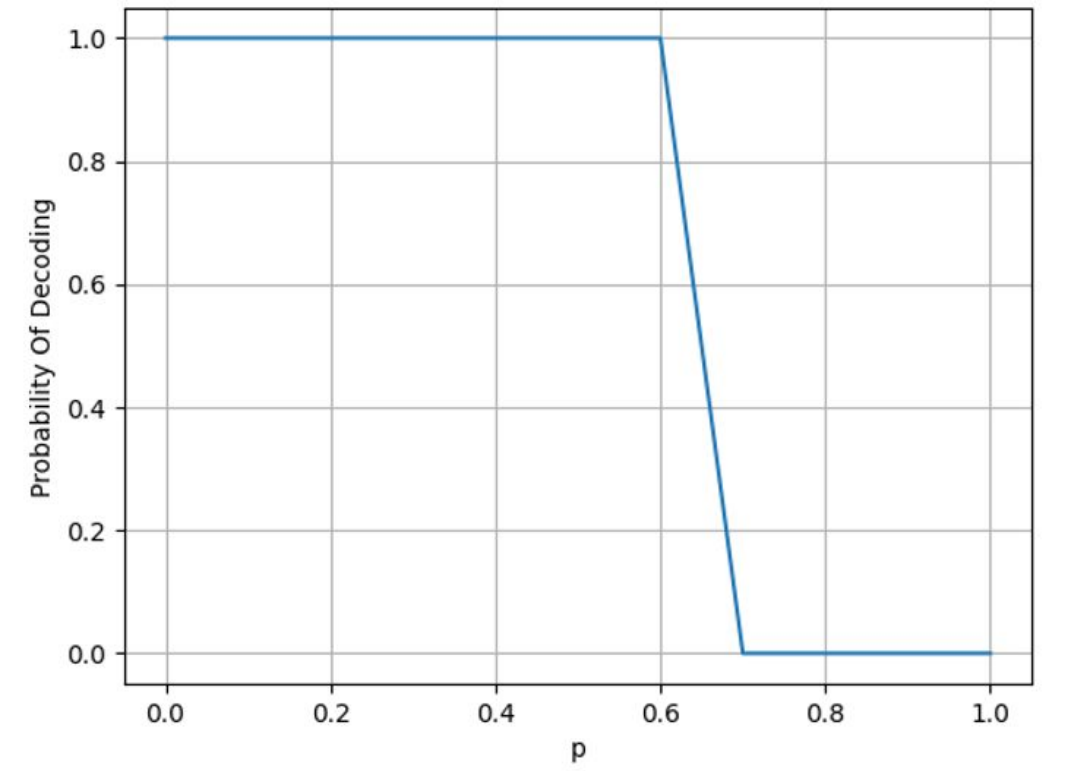
# SDD BEC 9*12 Matrix Probability of Decoding



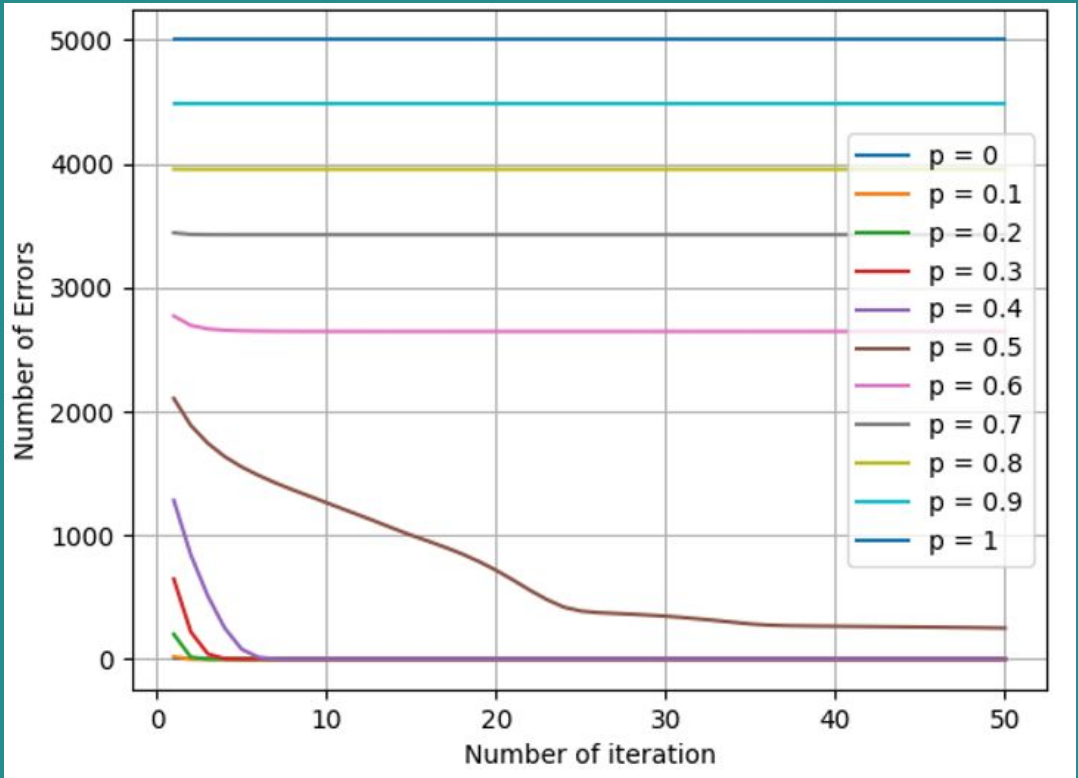
# SDD BEC H Matrix Error v/s Iteration



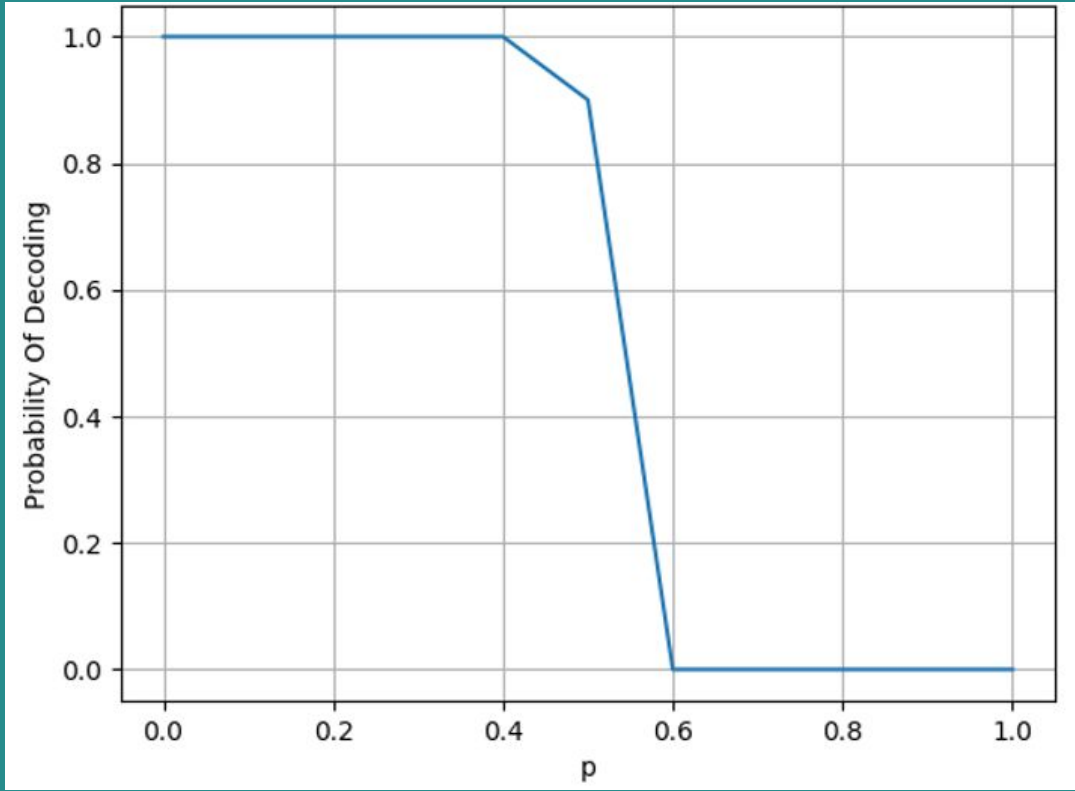
# SDD BEC H Matrix Probability of Decoding



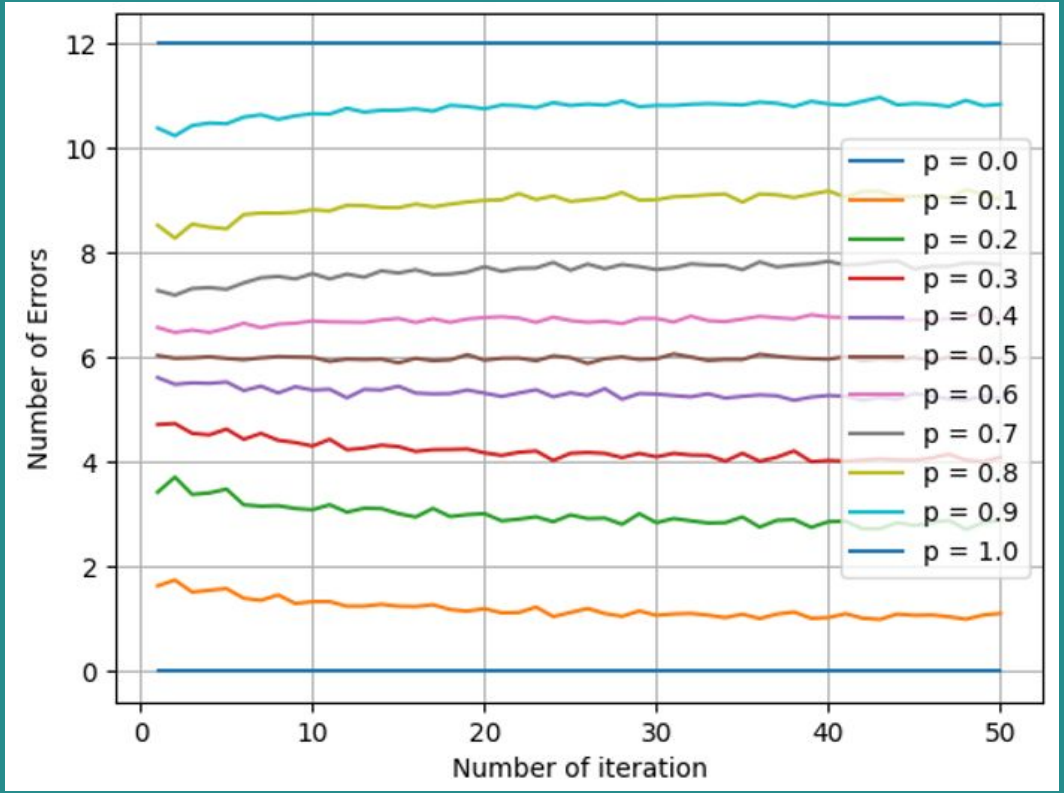
# SDD BEC H Matrix 2
# Error v/s Iteration



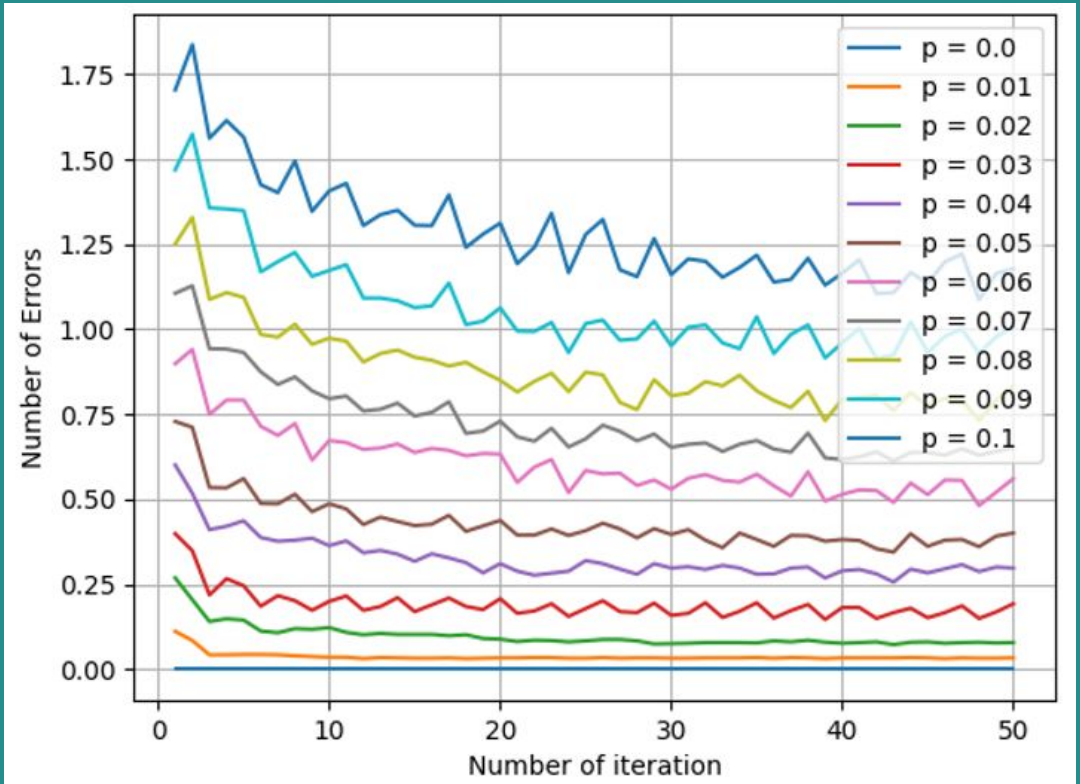
# SDD BEC H Matrix 2 Probability of Decoding



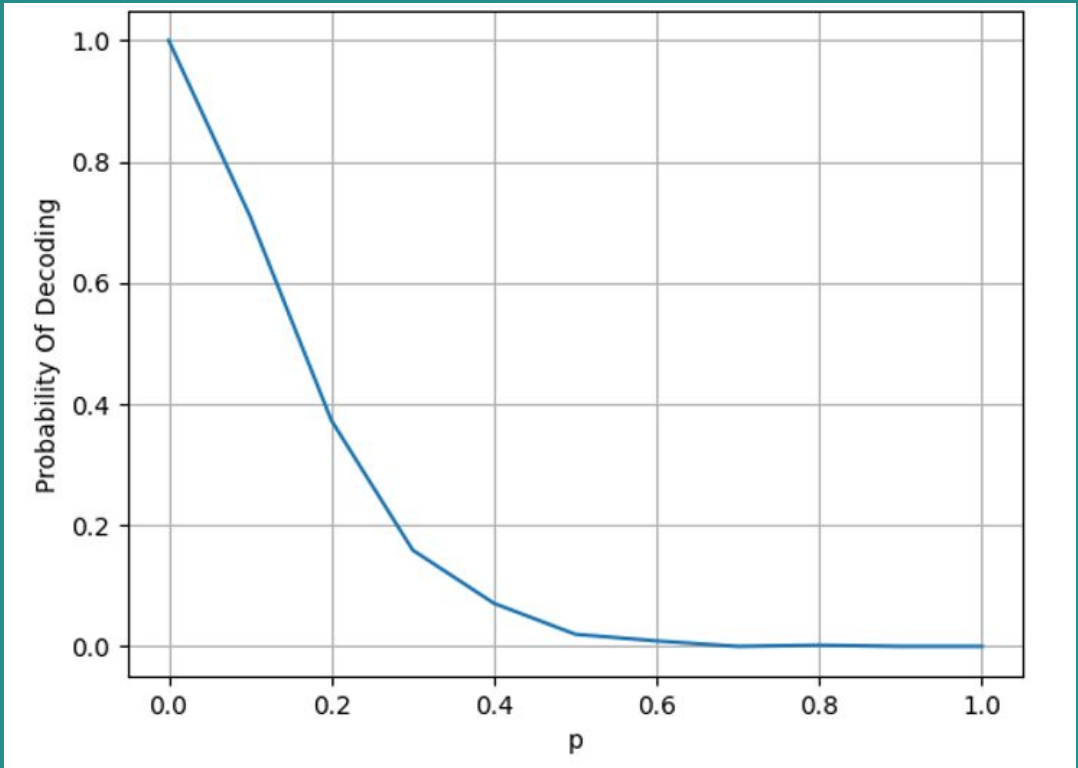
# HDD BSC 9*12 Matrix Error v/s Iteration



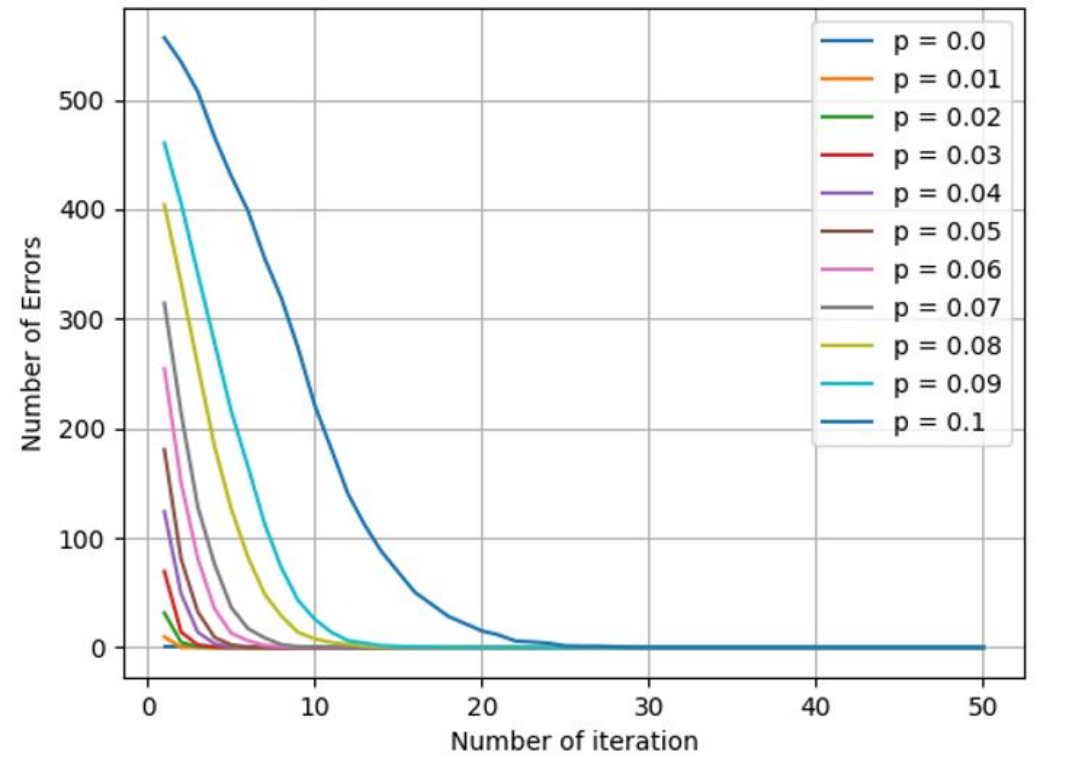
# HDD BSC 9*12 Matrix Error v/s Iteration



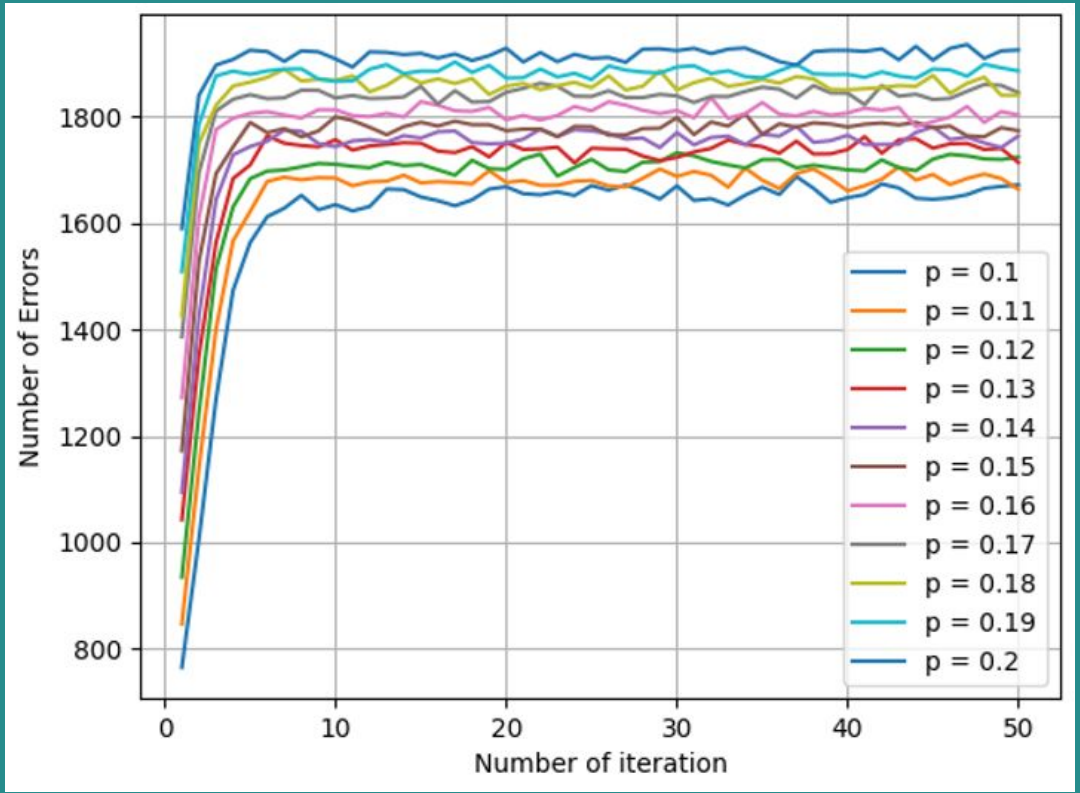
# HDD BSC 9*12 Matrix Probability of Decoding



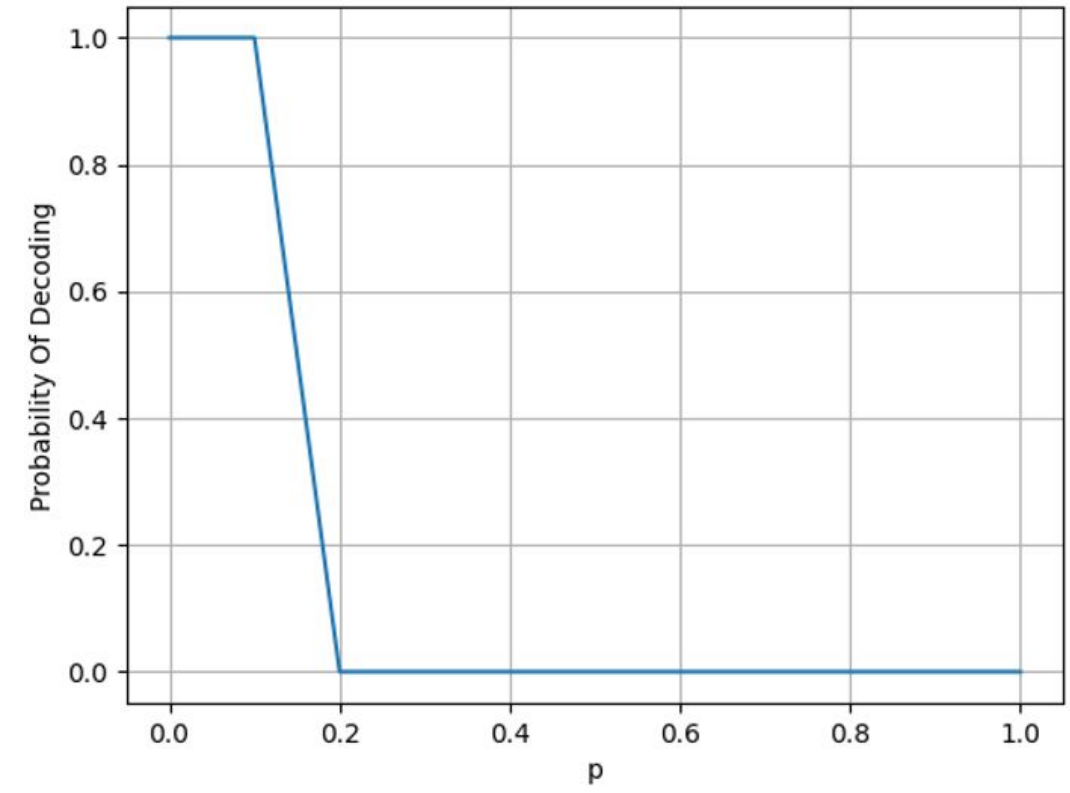
# HDD BSC H Matrix Error v/s Iteration



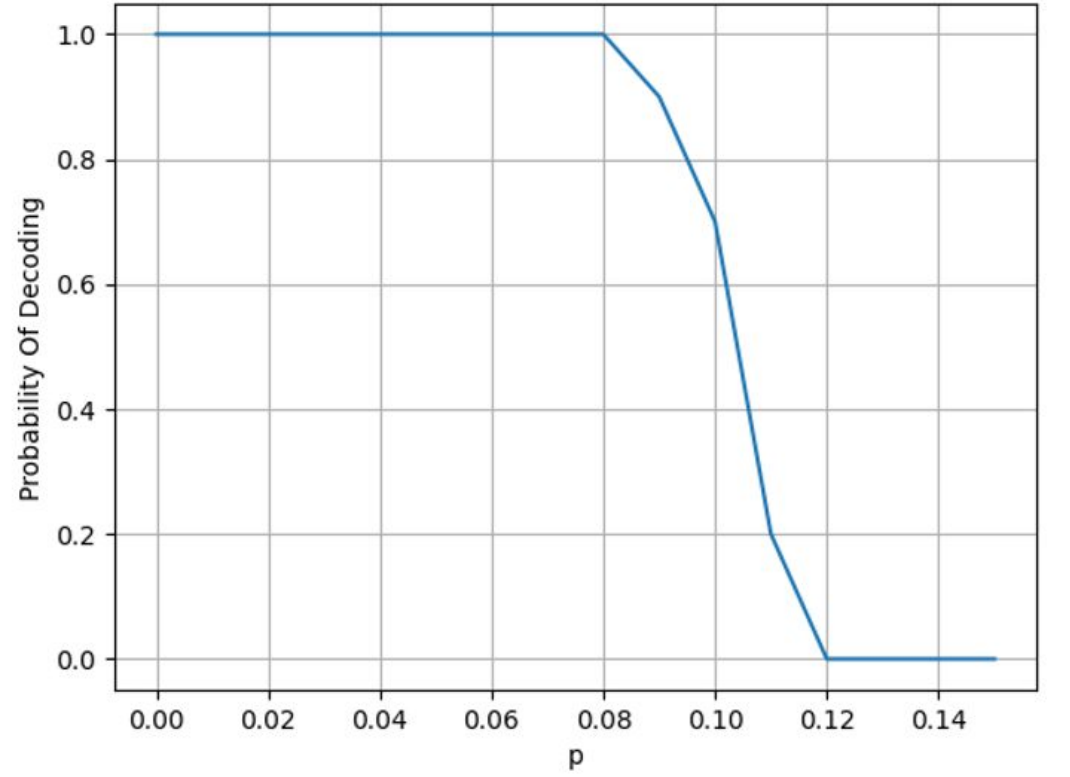
# HDD BSC H Matrix Error v/s Iteration



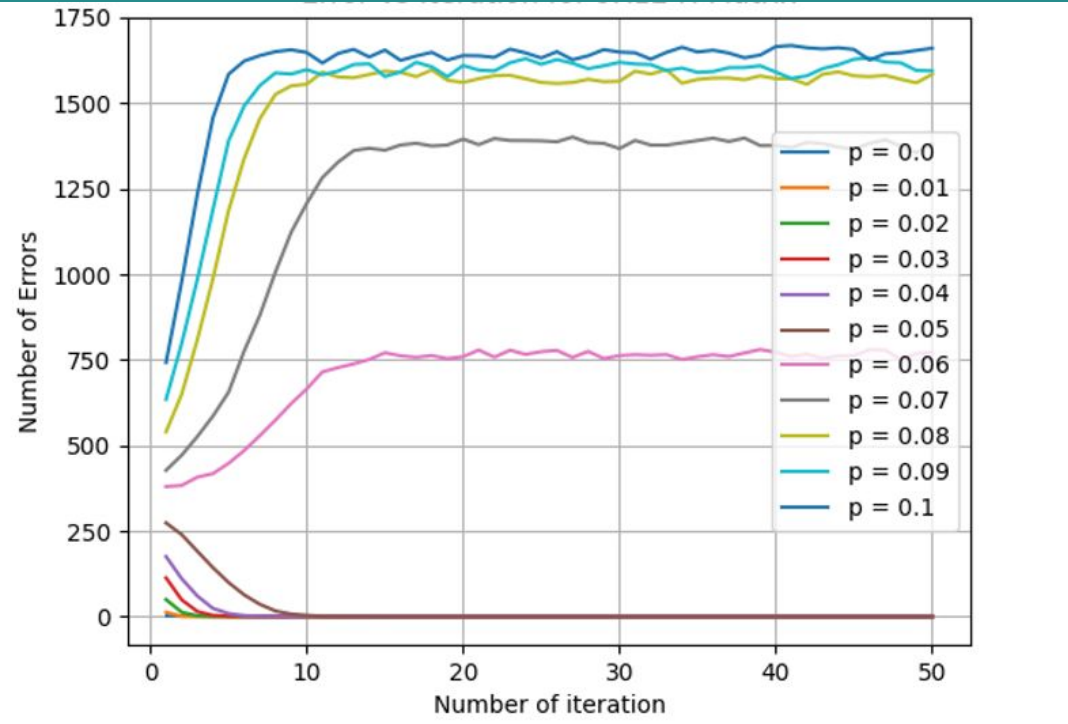
# HDD BSC H Matrix Probability of Decoding



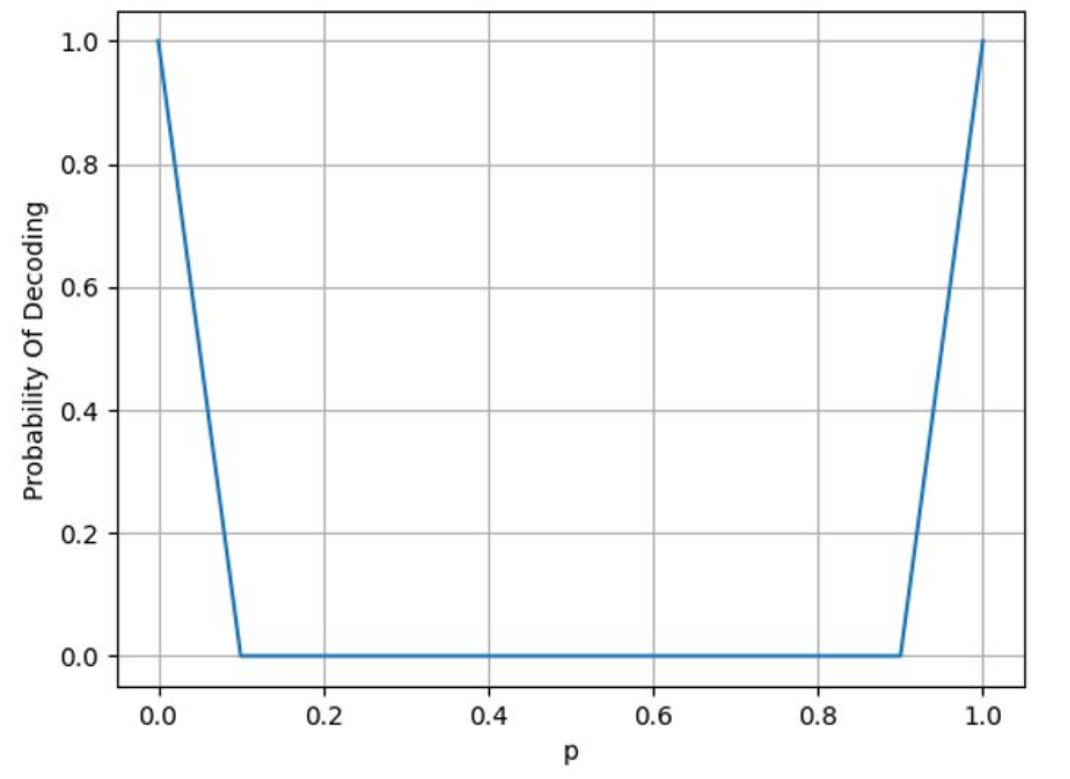
# HDD BSC H Matrix Probability of Decoding



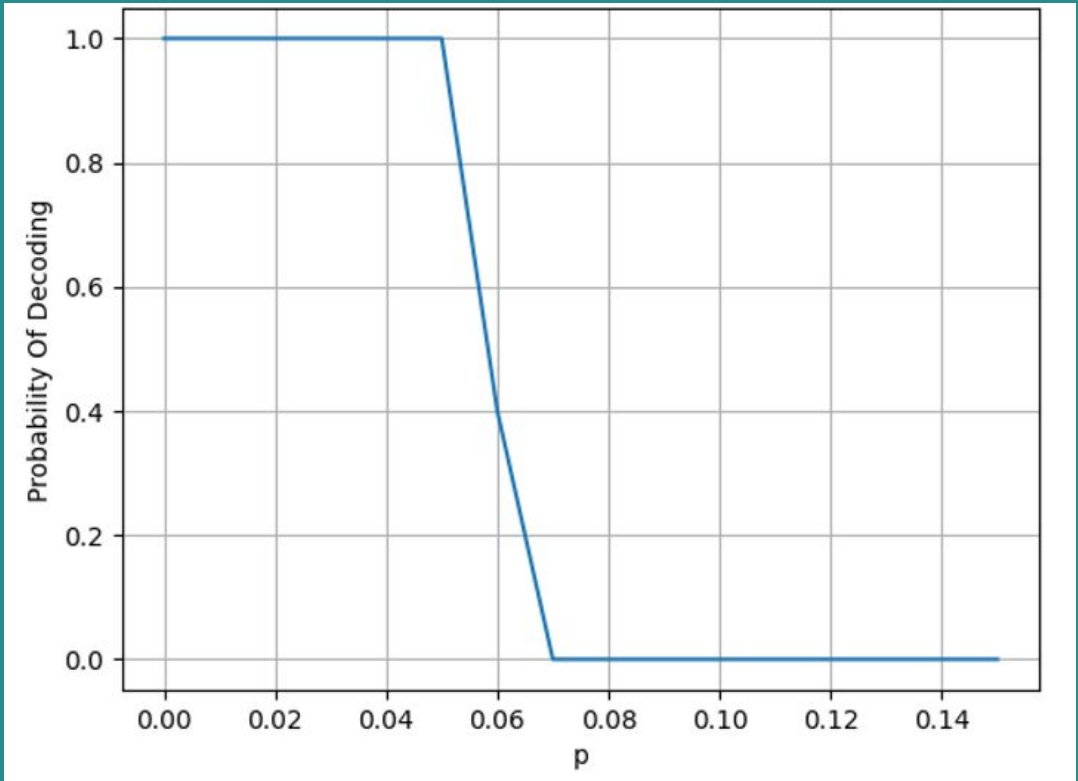
# HDD BSC H Matrix 2
# Error v/s Iteration



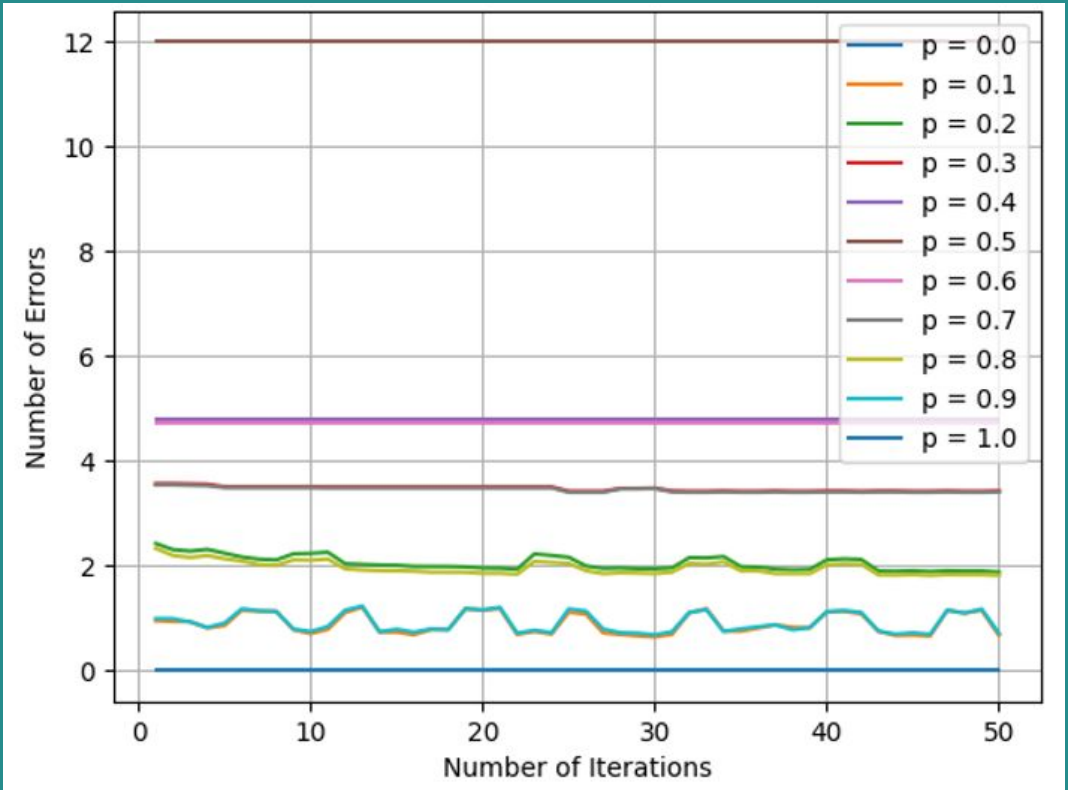
# HDD BSC H Matrix 2 Probability of Decoding



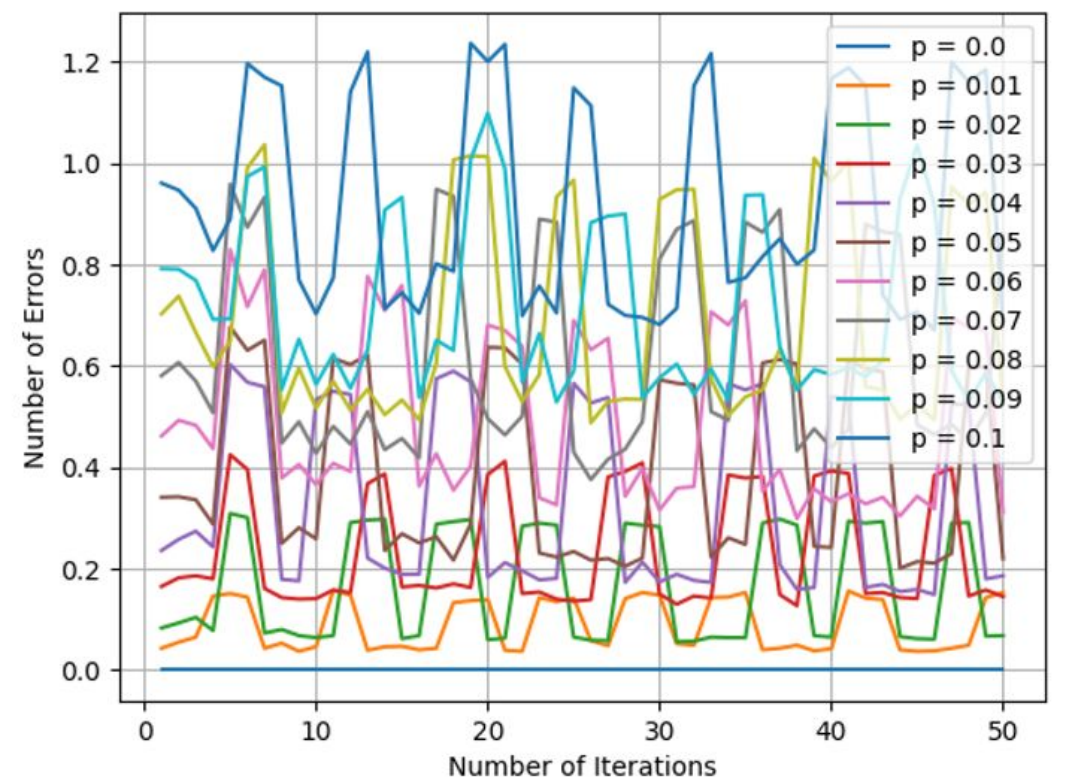
# HDD BSC H Matrix 2
# Probability of Decoding



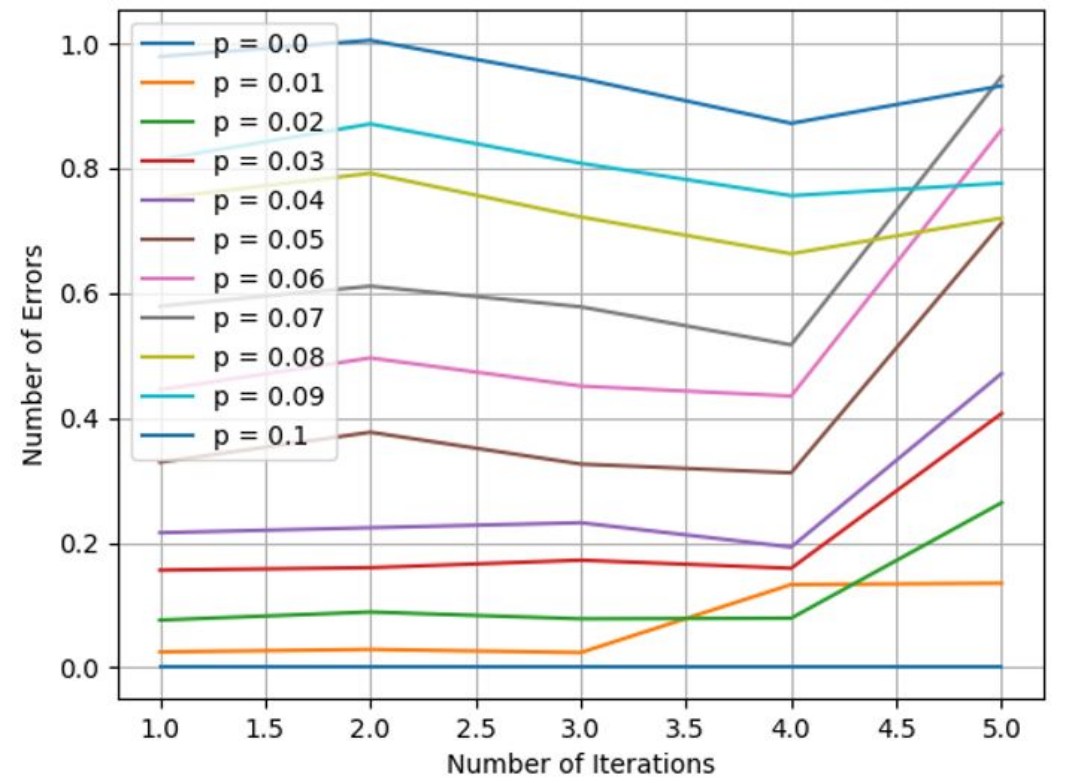
# SDD BSC 9*12 Matrix Error v/s Iteration



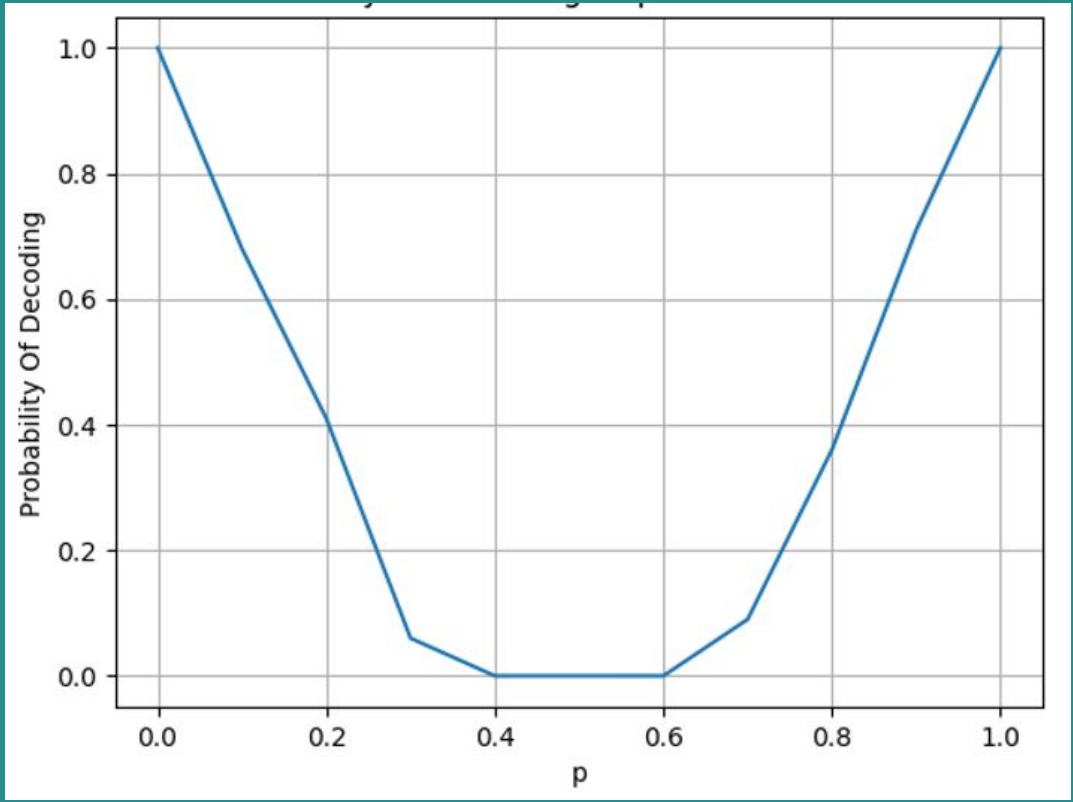
# SDD BSC 9*12 Matrix Error v/s Iteration



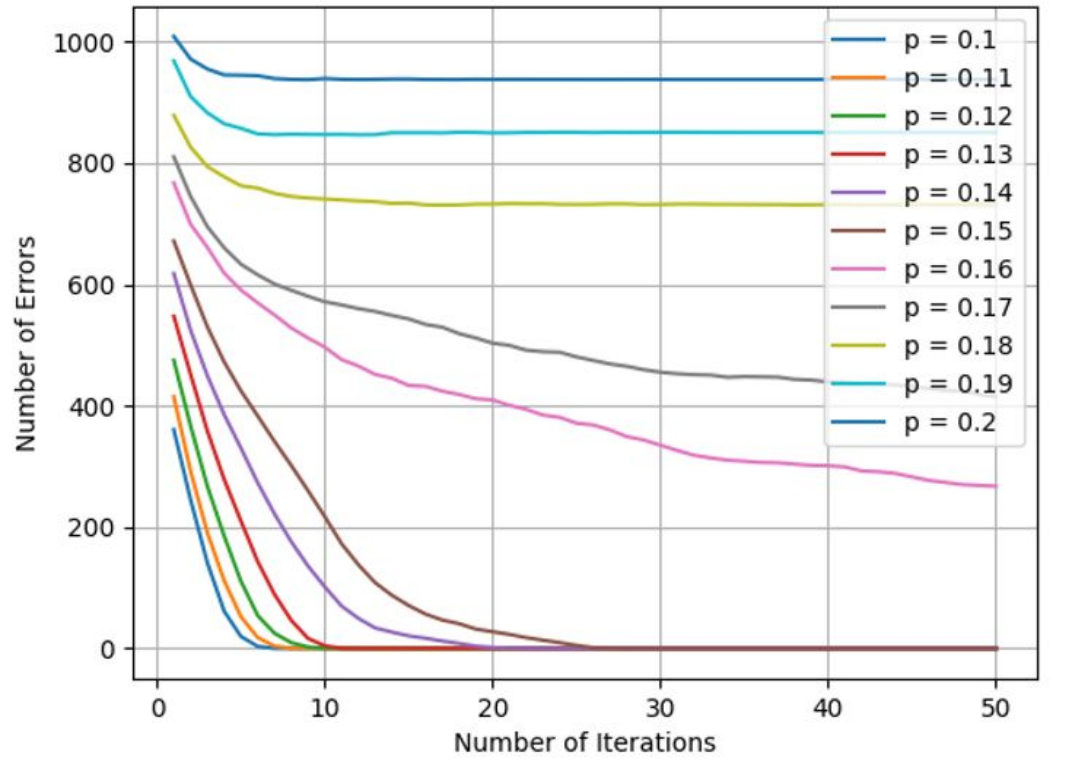
# SDD BSC 9*12 Matrix Error v/s Iteration



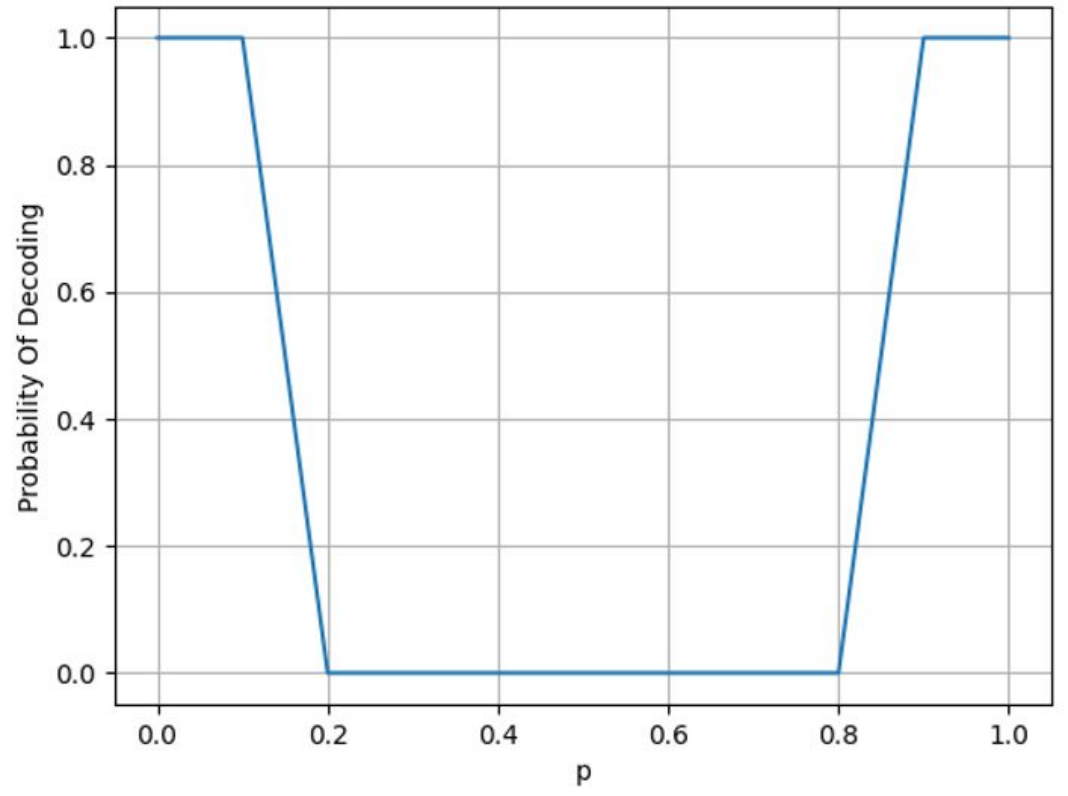
# SDD BSC 9*12 Matrix Probability of Decoding



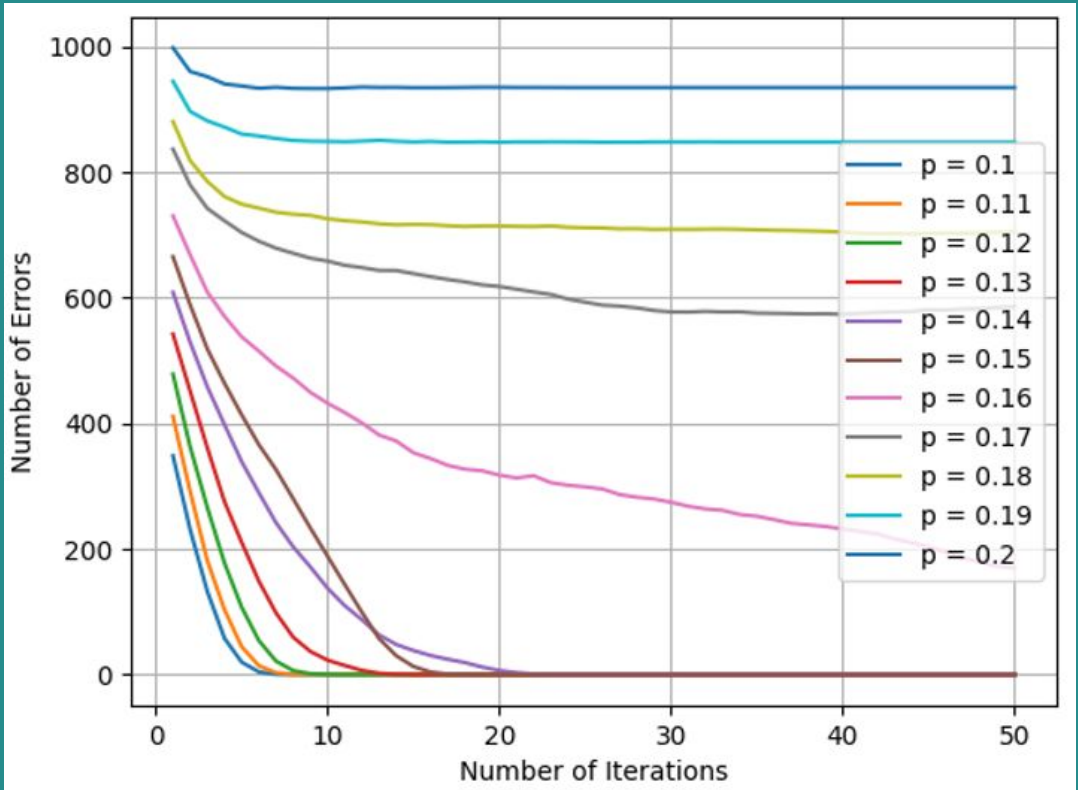
# SDD BSC H Matrix Error v/s Iteration



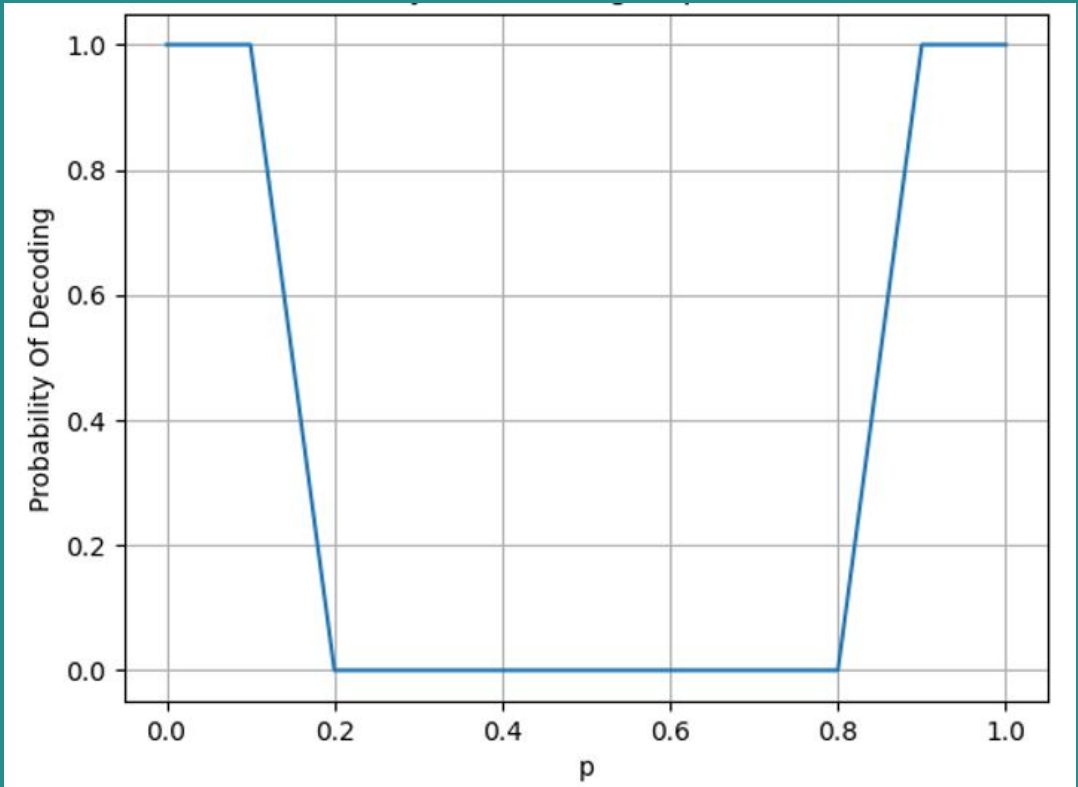
# SDD BSC H Matrix Probability of Decoding



# SDD BSC H Matrix 2
# Error v/s Iteration



# SDD BSC H Matrix 2 Probability of Decoding



5.

# Summary

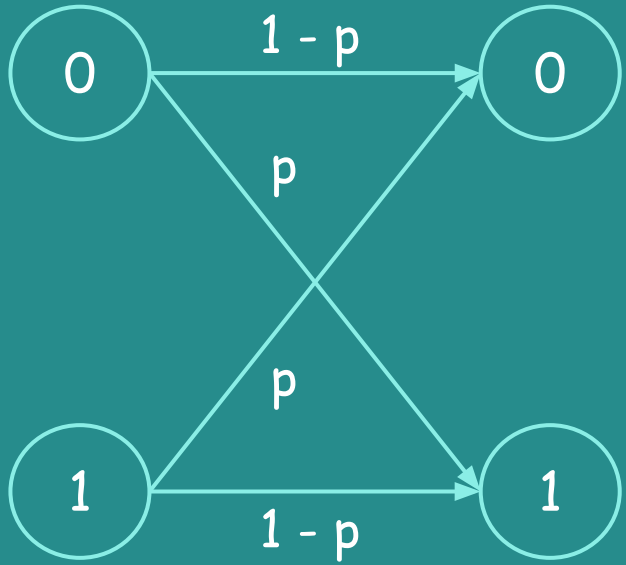
# Summary

- We'd never developed anything like this before. We found the project challenging but at the same time fascinating.
- We ran into a few logical errors, but we were able to overcome them thanks to our perseverance, even though it took some time.
- In general, the project helped us in grasping key concepts such as channel coding and graphs.
- It also taught us the value of algorithms that are both quick and efficient in terms of running time and storage space.

# Appendix

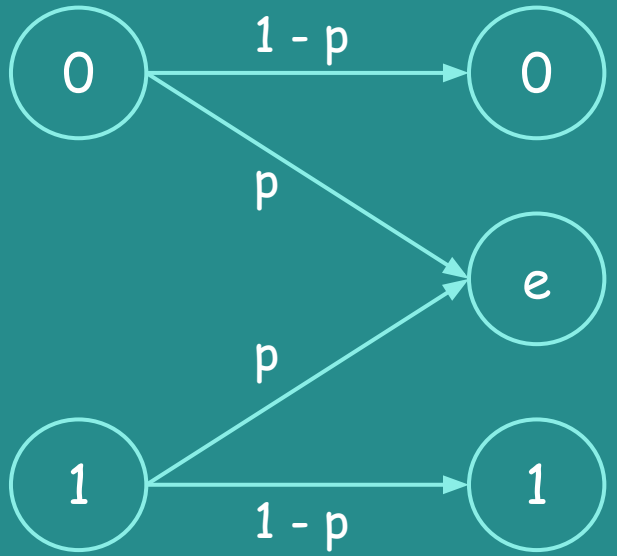
# Binary Symmetric Channel

- A binary symmetric channel (BSC) is a communications channel model.
- This channel has a Binary input and Binary output. In this model, a transmitter sends a bit 0 or bit 1, and a receiver receives this bit.
- The bit is either received correctly or is flipped with a crossover probability of P.



# Binary Erasure Channel

- A binary erasure channel (BEC) is a communications channel model.
- This channel has a Binary input and Binary output. In this model, a transmitter sends a bit 0 or bit 1, and a receiver receives this bit.
- The bit is either received correctly or is erased with a crossover probability of P.
- If bit is not erased then it is sure it contains right information



# Low - density Parity - check

- A low-density parity-check (LDPC) code is an error-correcting code. It is a method of transmitting messages over a noisy channel. It is constructed using a tanner graph which is a sub-class of Bipartite graphs.
- The benefit of using LDPC code is that these approach Shannon's limit more than any other code.
- LDPC codes are also known as Gallager codes, in honor of Robert G. Gallager, who developed the LDPC concept at the Massachusetts Institute of Technology in 1960.

# Tanner Graph

Tanner graph contains two type of nodes:

1. N-K Check nodes (CN)
2. N coded bit nodes or variable nodes (VN)

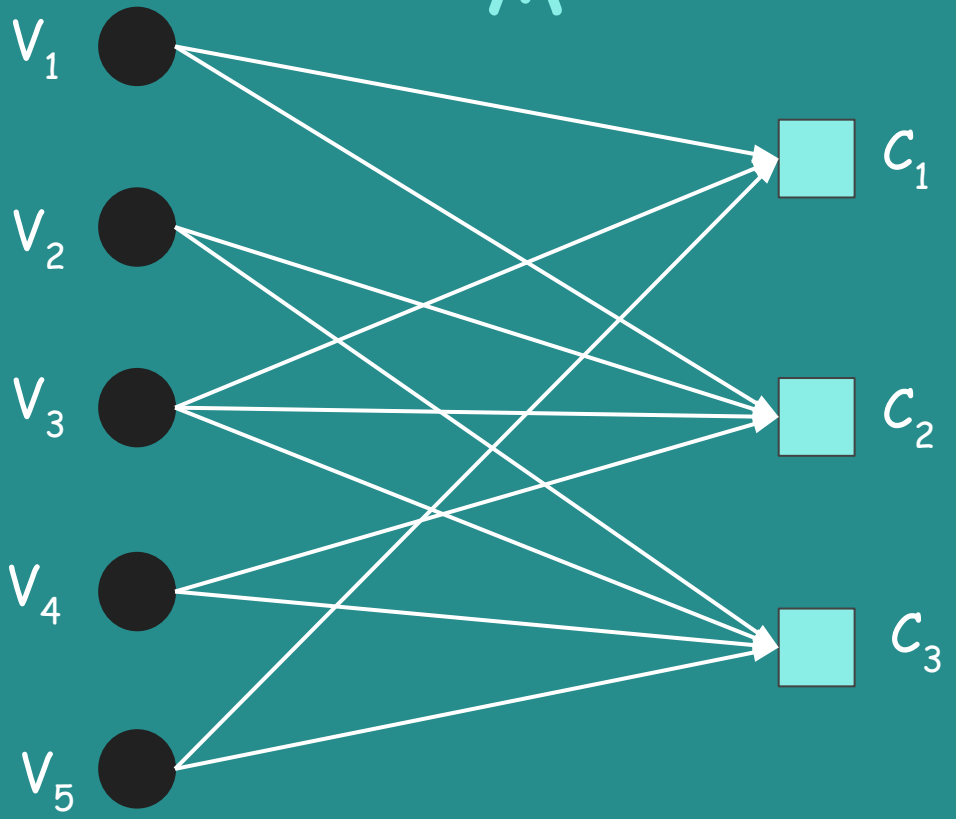
In tanner graph same types of nodes are never connected due to this binary partition of graphs it known as Bipartite Graph.

Degree of Node - number of edges connected to node

- ★ dc - degree of Check nodes
- ★ dv - degree of variable nodes

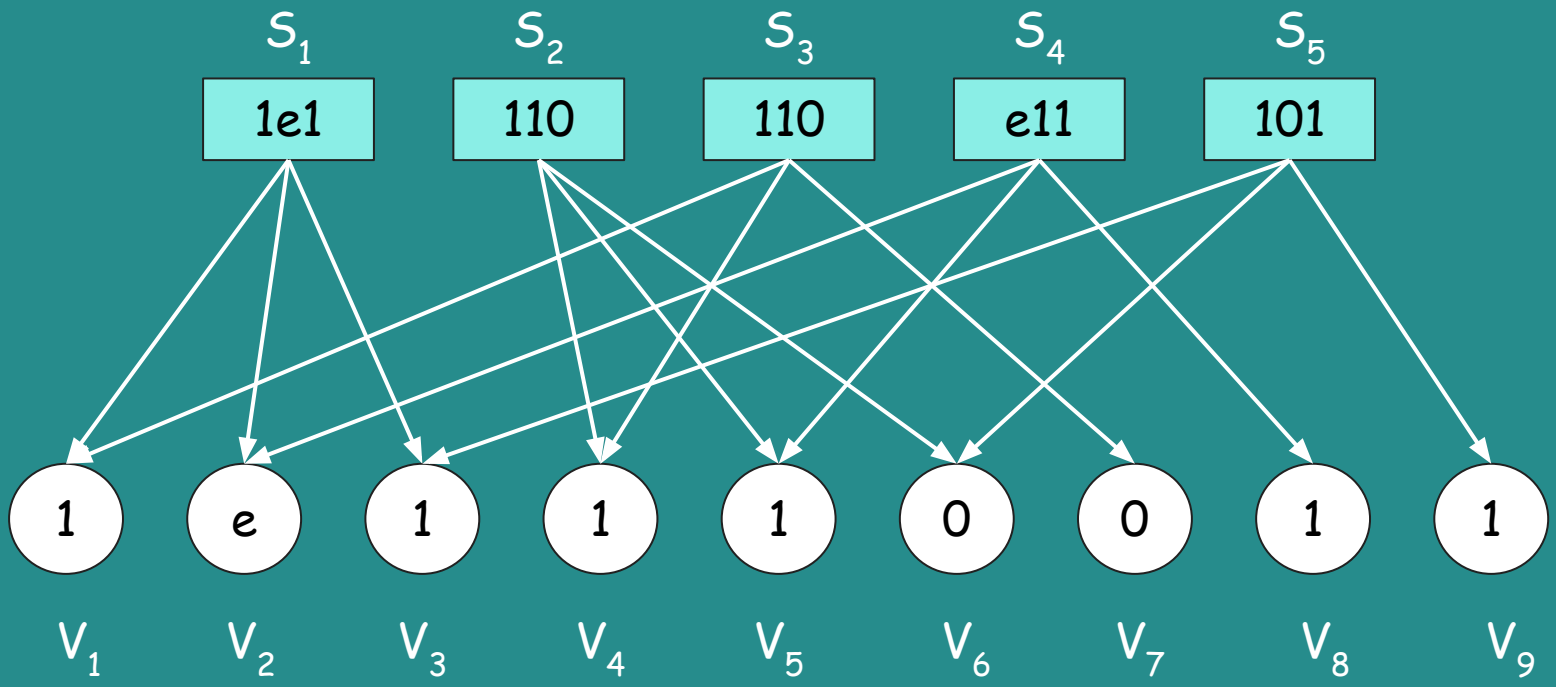
# Tanner Graph

| $V_1$ | $V_2$ | $V_3$ | $V_4$ | $V_5$ | |
|---|---|---|---|---|---|
| 1 | 0 | 1 | 0 | 1 | $C_1$ |
| 1 | 1 | 1 | 1 | 0 | $C_2$ |
| 0 | 1 | 1 | 1 | 1 | $C_3$ |

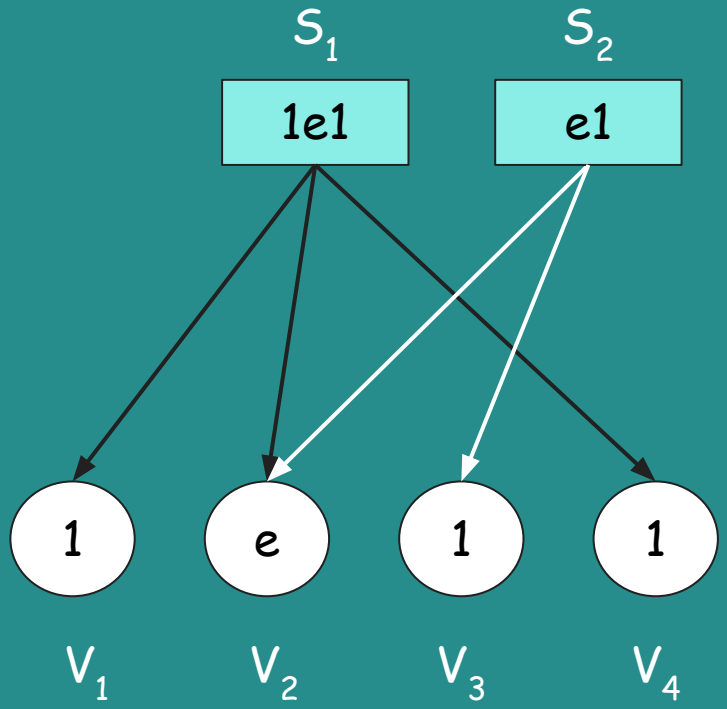


# BEC Hard Decision Decoding

Step 1: All the variable nodes send their received bits to the Check Nodes.

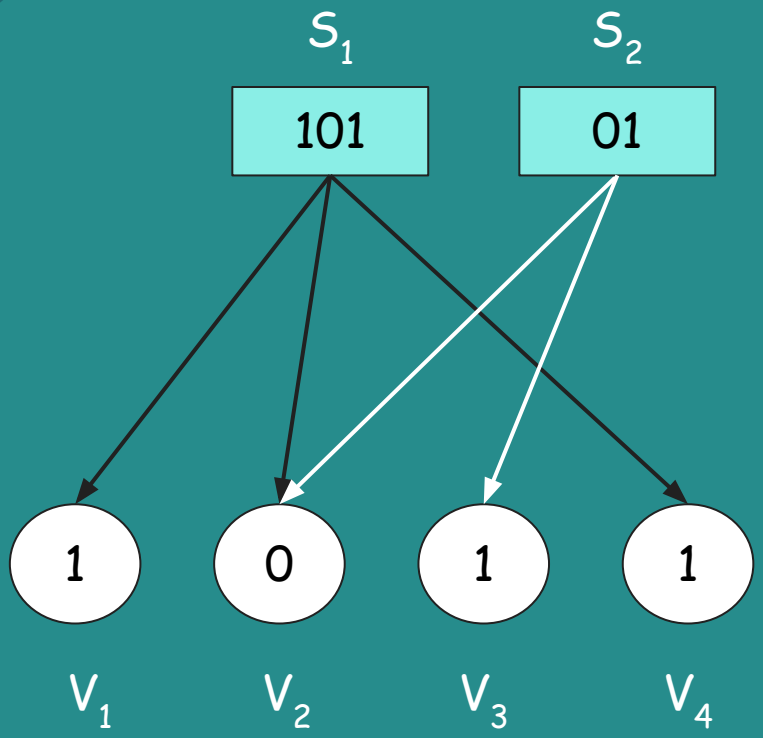


Step 2: The check nodes calculate the XOR of these received bits.




$$S_1 : v_1 \oplus v_2 \oplus v_3$$

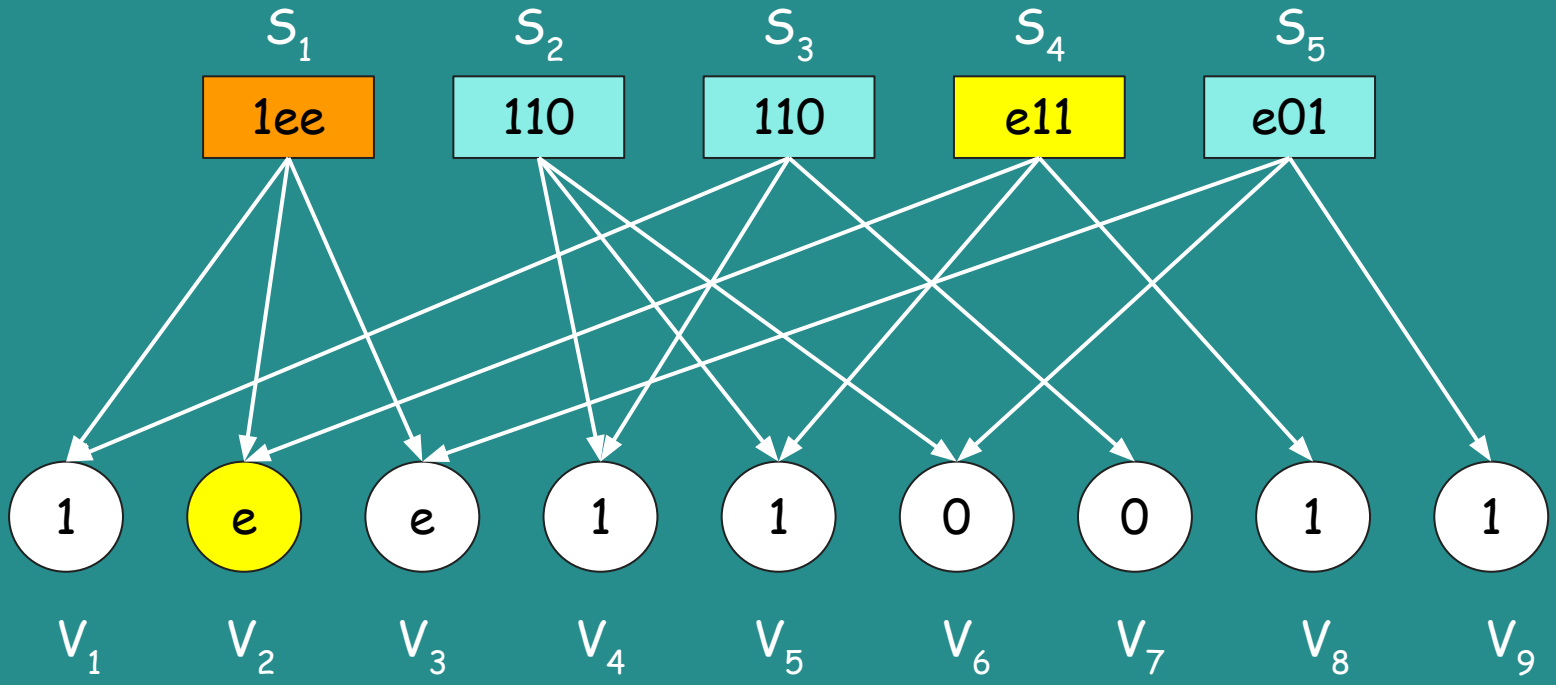
# Step 3: The calculated value is assigned to the variable node.




Here, $s_1$ should be equal to zero
Therefore, $v_2$ is assigned the value $v_1 \oplus v_4 = 1 \oplus 0 = 1$

Step 4: The variable nodes update their response messages to the check nodes.

If there is an erasure present in the other variable nodes, connected to the check node then the response is calculated using the other check nodes connected to the present variable node.



# Monte Carlo Simulations

- Monte Carlo simulation is a computerized mathematical technique that allows people to account for risk in quantitative analysis and decision making.
- Monte Carlo simulation furnishes the decision-maker with a range of possible outcomes and the probabilities they will occur for any choice of action. It shows the extreme possibilities—the outcomes of going for broke and for the most conservative decision—along with all possible consequences for middle-of-the-road decisions.

# Bibliographical Reference


- Lecture Notes
- BSC: https://en.wikipedia.org/wiki/Binary_symmetric_channel
- BEC: https://en.wikipedia.org/wiki/Binary_erasure_channel
- HDD and SDD:
  https://www.tutorialspoint.com/hard-and-soft-decision-decoding
  https://link.springer.com/chapter/10.1007/978-3-319-51103-0_2
- Monte - Carlo Simulations:
  https://en.wikipedia.org/wiki/Monte_Carlo_method

“The fundamental problem of communication is that of reproducing of one point either exactly or approximately a message selected at another point.”

*Claude Shannon*

# Thank You