**Course Code [CSE-260]**

[Data and Telecommunication]

**LAB Report [02]**

[On-off Binary Signal Transmission]

**Date of Submission: 21.12.2021**

**Submitted by**

**Name:**  Meraj al Maksud

**Class Roll:** 2247

**Submitted to**

Dr. Abu Sayed Md. Mostafizur Rahaman

*Professor*

*Department of Computer Science and Engineering*

*Jahangirnagar University*

*Savar, Dhaka*



**Department of Computer Science and Engineering**

**Jahangirnagar University**

**Savar, Dhaka, Bangladesh**

**Title:** Implementation of On-Off Binary Signal Transmission

1. **Statement of the Problem:**

Here we will try to find out how a digital signal is transmitted and how it gets mixed with noise and reaches the receiver and varies in output form of the signal. We will also cover the effect of SNR on the signal and the final result of the signal.

1. **Hypothesis:**

Possible solution could be considered a signal and mix it with noise and change the SNR with it and see the change in output and analyse it. We can see the change using graphical representation. For this we will use-

**Python Language**

**Matlab Command prompt**

1. **Materials:** 
   * + 1. Matlab
       2. Google Co-Lab
2. **Procedure:**

We have to generate the probability of error when 0 is transmitted and then the probability of error when 1 is transmitted. After that We will add active white gaussian noise in the signal and calculate the probability of the error and the Confidence level in the final output.

1. **Results in MATLAB (Code with figure):**

For value, M =500;

**Code:**

M=500;

for k=1:8,

SNR=2+k\*2;

tx=randi(2, M, 1)-1;

rx=awgn(tx,SNR);

e(k)=0;

for i=1:M,

if tx(i)==1;

if rx(i)<=0.5;

e(k)=e(k)+1;

end

end

if tx(i)==0;

if rx (i)>=0.5;

e(k)=e(k)+1;

end

end

end

end

pe=e/M;

SNR=4:2:18;

SNR\_a=10.^(SNR/10);

pb=qfunc(sqrt(SNR\_a/4));

plot(SNR,pe,'r>:',SNR,pb,'bs:')

legend('simulation','theory')

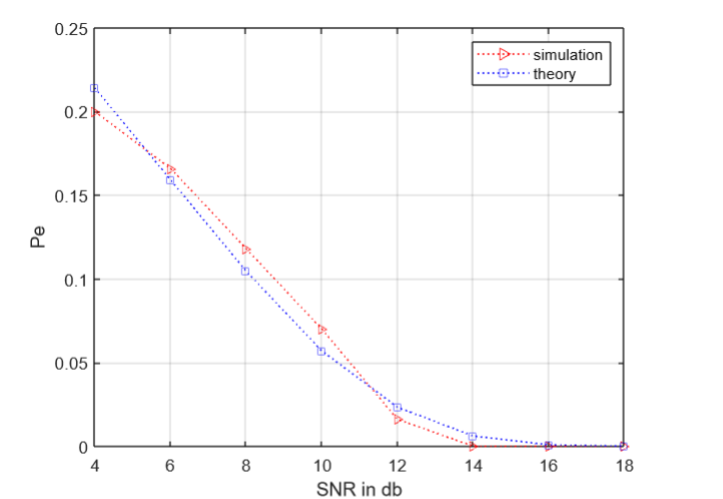
xlabel('SNR in db')

ylabel('Pe')

grid on;

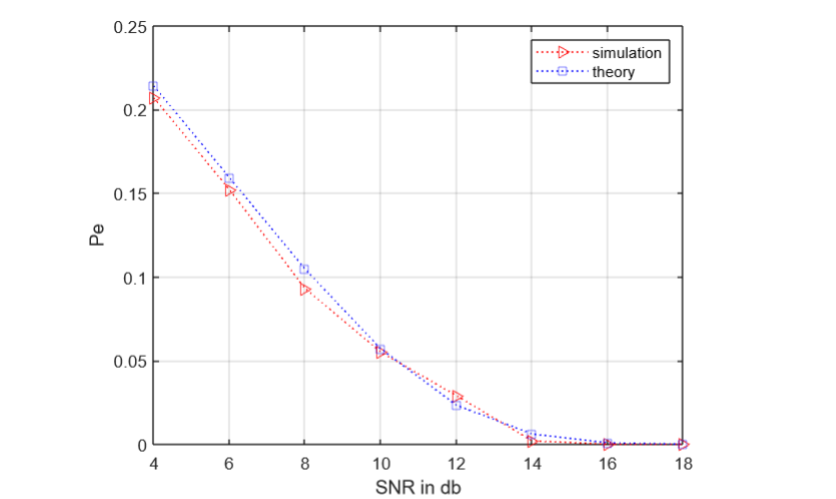
C = (1-abs((pe-pb)./pb))\*100;

**Output in MATLAB:**

****

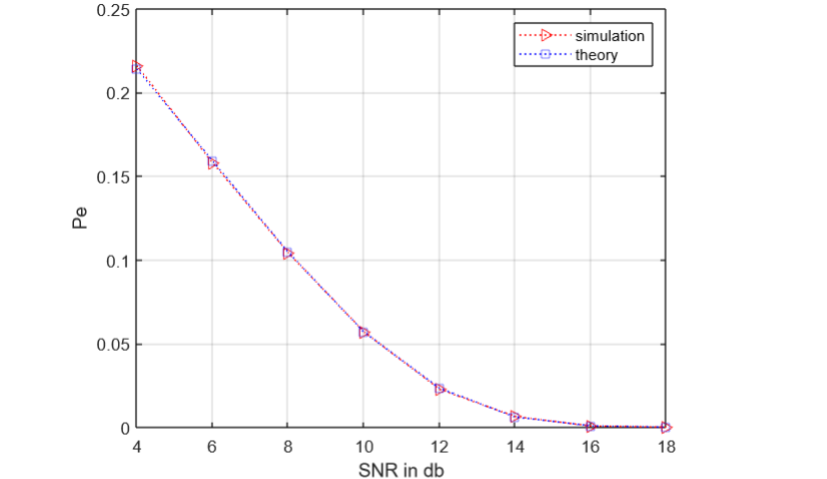
For value, M=1000;

**Output in MATLAB:**



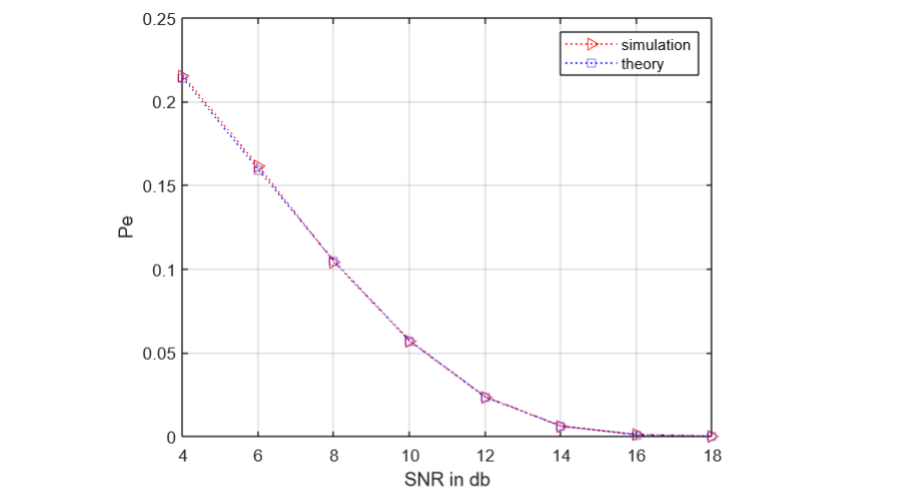
For value, M=50000;

**Output in MATLAB:**

****

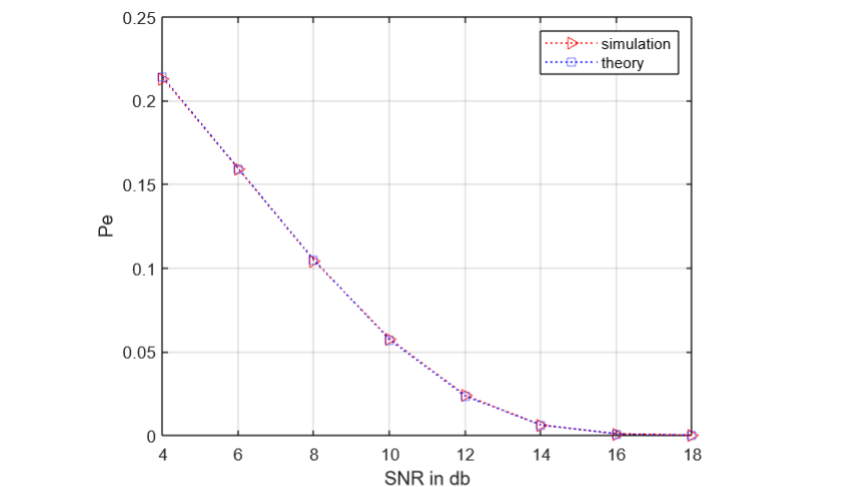
**For value, M=100000;**

**Output in MATLAB:**

****

**For value, M=200000;**

**Code in MATLAB:**

****

1. **Results in PYTHON (Code with figure):**

**Code:( For value, M=500 )**

**from** typing **import** List, Any

**import** numpy **as** np

**import** random **as** rand

**from** numpy **import** sum,isrealobj,sqrt

**from** numpy.random **import** standard\_normal

**from** scipy **import** special **as** sp

**import** matplotlib.pyplot **as** plt

**def** qfunc(x):

**return** 0.5-0.5\*sp.erf(x/sqrt(2))

**def** awgn(s,SNRdB,L=1):

gamma = 10\*\*(SNRdB/10)

**if** s.ndim==1:

P=L\*sum(abs(s)\*\*2)/len(s)

**else**:

P=L\*sum(sum(abs(s)\*\*2))/len(s)

N0=P/gamma

**if** np.isrealobj(s):

n = sqrt(N0/2)\*standard\_normal(s.shape)

**else**:

n = sqrt(N0/2)\*(standard\_normal(s.shape)+1j\*standard\_normal(s.shape))

r = s + n

**return** r

M = 500;

e = [0, 0, 0, 0, 0, 0, 0, 0]

**for** k **in** range(0, 8, 1):

SNR = 2\*k+2

tx = []

**for** i **in** range(0, M):

tx.append(rand.randint(0, 1))

rx = awgn(np.array(tx), SNR);

e[k] = 0

**for** i **in** range(0, M):

**if** tx[i]==1:

**if** rx[i] <= 0.5:

e[k] = e[k]+1;

**if** tx[i]==0:

**if** rx[i]>=0.5:

e[k] = e[k]+1;

pe = []

**for** x **in** e:

pe.append(x/M)

SNR = []

pb: List[Any] = []

**for** i **in** range(4, 19, 2):

SNR.append(i)

SNR\_a = pow(10, i/10)

pb.append(qfunc(sqrt(SNR\_a/4)))

plt.xlabel("SNR in db")

plt.ylabel("pe")

plt.grid()

xpoint = np.array(SNR)

ypoint = np.array(pe)

plt.plot(xpoint, ypoint, 'r')

xpoint = np.array(SNR)

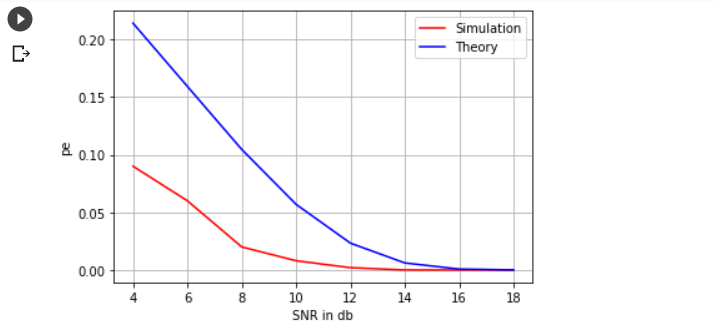
ypoint = np.array(pb)

plt.plot(xpoint, ypoint, 'b')

plt.legend(["Simulation", "Theory"])

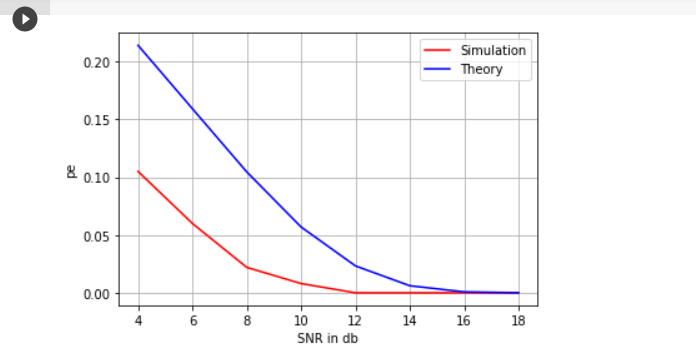
plt.show()

**Output in Python:**

****

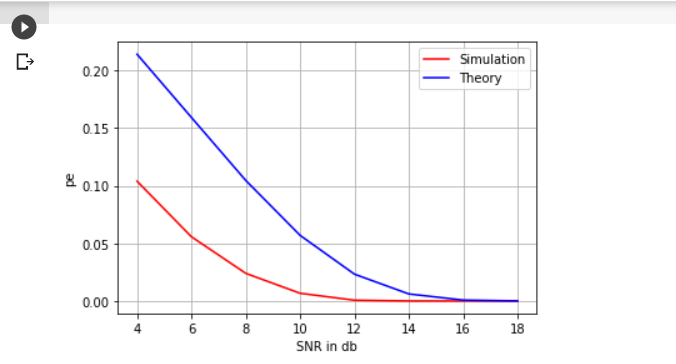
**( For value, M=1000 )**

**Output in Python:**

****

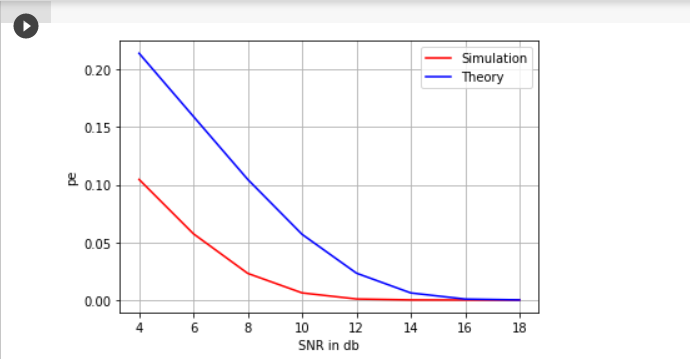
**( For value, M=50000 )**

**Output in Python:**

****

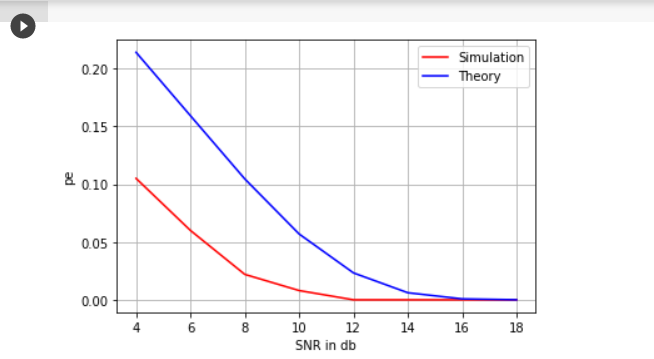
**( For value, M=100000 )**

**Output in Python:**

****

**( For value, M=500000 )**

**Output in Python:**

****

1. **Conclusions:**

Here, I accept my hypothesis because the concept of pdf and cdf is perfect for binary signal transmission. But there are some errors.