# Contents

[Contents 1](#_Toc23729723)

[List of Figure 4](#_Toc23729724)

[Question 01 5](#_Toc23729725)

[Part A 5](#_Toc23729726)

[Part B 5](#_Toc23729727)

[Part B-1 5](#_Toc23729728)

[Part B-2 5](#_Toc23729729)

[Part C 5](#_Toc23729730)

[Part C-1 5](#_Toc23729731)

[Part C-2 6](#_Toc23729732)

[Part C-3 7](#_Toc23729733)

[Part C-4 7](#_Toc23729734)

[Part C-5 7](#_Toc23729735)

[Part C-6 7](#_Toc23729736)

[Part C-7 7](#_Toc23729737)

[Part D 8](#_Toc23729738)

[Part D-1 8](#_Toc23729739)

[Part D-2 8](#_Toc23729740)

[Part D-3 8](#_Toc23729741)

[Question 02: BER Simulation 9](#_Toc23729742)

[Part A-1 9](#_Toc23729743)

[1) 9](#_Toc23729744)

[2) 9](#_Toc23729745)

[Part A-2 10](#_Toc23729746)

[Part A-3 10](#_Toc23729747)

[Part B 12](#_Toc23729748)

[Part C 13](#_Toc23729749)

[Part D 13](#_Toc23729750)

[Part E 14](#_Toc23729751)

[Part F 15](#_Toc23729752)

[Part G 15](#_Toc23729753)

[Part H 16](#_Toc23729754)

[Part I 16](#_Toc23729755)

[Part J 16](#_Toc23729756)

[Part J-1 16](#_Toc23729757)

[Part J-2 17](#_Toc23729758)

[Part J-3 18](#_Toc23729759)

[Question 04 19](#_Toc23729760)

[Part A: Selection Diversity Combining 19](#_Toc23729761)

[Part A-1 19](#_Toc23729762)

[Part A-2 20](#_Toc23729763)

[Part A-3 20](#_Toc23729764)

[Part A-4 21](#_Toc23729765)

[Part B: Equal Gain Combining 21](#_Toc23729766)

[Part B-1 21](#_Toc23729767)

[Part B-2 22](#_Toc23729768)

[Part B-3 23](#_Toc23729769)

[Part B-4 23](#_Toc23729770)

[Part C: Maximum Ratio Combining 23](#_Toc23729771)

[Part C-1 23](#_Toc23729772)

[Part C-2 23](#_Toc23729773)

[Part C-3 25](#_Toc23729774)

[Part C-4 25](#_Toc23729775)

[Part C-5 26](#_Toc23729776)

[Part D: Compare the Performance 26](#_Toc23729777)

[Part D-1 26](#_Toc23729778)

[Part D-2 27](#_Toc23729779)

[Part E: Investigate the Diversity Gain 27](#_Toc23729780)

[Part E- 1 27](#_Toc23729781)

[Part E-2 27](#_Toc23729782)

[Part E-3 29](#_Toc23729783)

[Part E-4 29](#_Toc23729784)

[Part E-5 30](#_Toc23729785)

# List of Figure

[Figure 1: Angle and Mag 4](#_Toc23729786)

[Figure 2: Q01\_C\_1 5](#_Toc23729787)

[Figure 3: Q01\_C\_2 6](#_Toc23729788)

[Figure 4: Mean Value 6](#_Toc23729789)

[Figure 5: Q01\_D 7](#_Toc23729790)

[Figure 6: Syms make Positive 7](#_Toc23729791)

[Figure 7: Output a-3 10](#_Toc23729792)

[Figure 8; BER a-3 10](#_Toc23729793)

[Figure 9: Q2 Part B 11](#_Toc23729794)

[Figure 10: Q2 Part B Graph 11](#_Toc23729795)

[Figure 11: Q02 Part D Graph 13](#_Toc23729796)

[Figure 12: Q02 Part D 13](#_Toc23729797)

[Figure 13: Part E 14](#_Toc23729798)

[Figure 14: Part G 15](#_Toc23729799)

[Figure 15: Part J Graph 17](#_Toc23729800)

[Figure 16: BER Part J 17](#_Toc23729801)

[Figure 17: Q04 A Graph 19](#_Toc23729802)

[Figure 18: Q04 A 19](#_Toc23729803)

[Figure 19: L=3, L=1, L=5 20](#_Toc23729804)

[Figure 20: Q04 B Graph 21](#_Toc23729805)

[Figure 21: Compare Six Graph 22](#_Toc23729806)

[Figure 22: Q04 C Graph 24](#_Toc23729807)

[Figure 23: Compare Six Graph 25](#_Toc23729808)

[Figure 24: Nine graph 26](#_Toc23729809)

[Figure 25: Q04 E Graph 28](#_Toc23729810)

[Figure 26: Q04 E 28](#_Toc23729811)

[Figure 27: Compare of modified and original 29](#_Toc23729812)

# Question 01

## Part A

clc

clear all

z=3;

y=5;

h=z\*randn + j\*y\*randn

## Part B

### Part B-1

h=z+iy

|h|=sqrt (z^2 + y^2)

and

|h|^2 =z^2 + y^2

### Part B-2

**Code**

clc

clear all

z=3;

y=1;

h=z\*randn + j\*y\*randn

mag=sqrt(real(h)^2+ imag(h)^2)

angle\_=phase (h)

**Output**

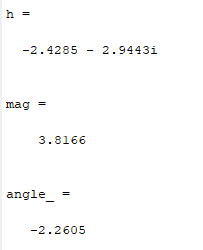


Figure : Angle and Mag

## Part C

### Part C-1

**Code**

x=0:0.01:5;

f=2\*x.\*exp(-x.^2);

plot(x,f)

**Output**

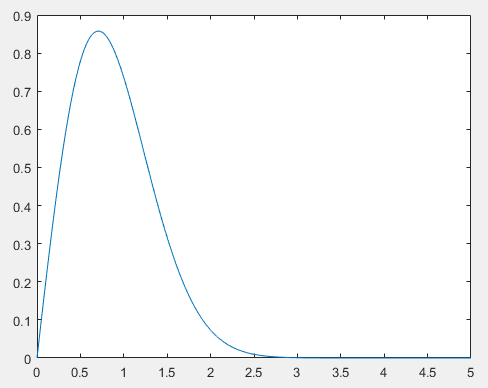


Figure : Q01\_C\_1

### Part C-2

**Code**

close all

x=0:0.01:5;

f\_c=exp(-x);

plot(x,f\_c)

**Output**

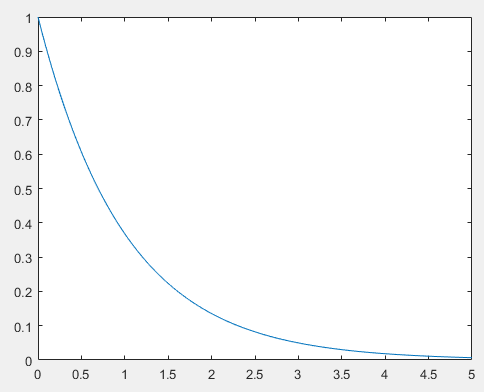


Figure : Q01\_C\_2

### Part C-3

Complex expansional is the missing here.

### Part C-4

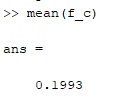


Figure : Mean Value

### Part C-5

We can use the h in both term like rectangular and polar. For the energy calculate just need to take the magnitude of the h. Magnitude take by just square root of sum of the z and y square.

### Part C-6

The performance is better in AWGN. But we can see in the AWGN we need to take the conjugate and calculate the sum of series.

### Part C-7

Both are same for the decision variable D. Gaussian noise is add according to need and take decision accurate according to specification. The noise is show separate in equation which is due to gaussian noise.

## Part D

### Part D-1

clear

% clear the worksapce

syms c EbN0

% declear the variable symbolic

% calculate the Error instance

instantBER=0.5\*(1-erf(sqrt(c\*EbN0)));

% BER fading calculate

BER\_fading=int(instantBER\*exp(-c),c,0,inf);

% Simplify the BER

pretty((BER\_fading))

### Part D-2

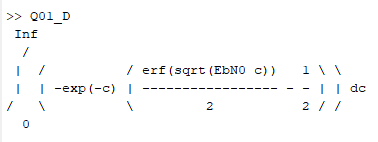


Figure : Q01\_D

The equation is mention in the question is same as we obtain the output. We can see the both equations are same if we more simplify the equation.

### Part D-3

**Code**

clear

% clear the worksapce

syms c EbN0 positive

% declear the variable symbolic

% calculate the Error instance

instantBER=0.5\*(1-erf(sqrt(c\*EbN0)));

% BER fading calculate

BER\_fading=int(instantBER\*exp(-c),c,0,inf);

% Simplify the BER

pretty((BER\_fading))

**Output**

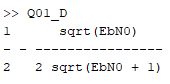


Figure : Syms make Positive

# Question 02: BER Simulation

## Part A-1

### 1)

The BER value depend on the bold part of the given code. In this code calculate the BER by using the three-main value which are bold.

### 2)

clear all

clc

close all

% clear all work space and command window and close all figure

EbN0dB\_vector=0:2:20;

% make vector like vale 0 2 4 6 ... 20

Eb=1;

for snr\_i=1:length(EbN0dB\_vector) %repeat the loop for the length of vector

EbN0dB=EbN0dB\_vector(snr\_i); %load one by one value of the vector

EbN0=10.^(EbN0dB/10); %convert from db to back value

N0=Eb/EbN0; %decimal value

sym\_cnt=0; %counter for count the error and system count

err\_cnt=0;

while err\_cnt<500 %repeat the loop for count Error less then 500

s=sqrt(Eb)\*sign(rand-0.5); %create the s is negative or positive

h=sqrt(1/2)\*(randn+j\*randn);% create h and n by calling randn function and sqrt use

n=sqrt(N0/2)\*(randn+j\*randn);

r=h\*s+n; %calculate the value r

D=r\*exp(-j\*angle(h)); %calculate the angle and express in exp form

s\_hat=sign(D); %calculate real value

if s\_hat~=sign(s) %if the sign is different it error increment

err\_cnt=err\_cnt+1;

end

sym\_cnt=sym\_cnt+1; % otherwise system counter increment

end

BER(snr\_i)=err\_cnt/sym\_cnt %BER store the ratio b/w error count vs system count

end

% plot the vecter and BER value and label xaxis and yaxis

figure

semilogy(EbN0dB\_vector, BER)

xlabel('E\_b/N\_0 [dB]')

ylabel('BER')

grid

## Part A-2

If we use the h=sqrt(1/2)\*(randn+j\*randn) before while loop then h value is fixed and no chance to change the count error and loop move towards infinity. This one is happening when we change the sign(D) to sign(real(D)). Otherwise result is same.

## Part A-3

**Code**

clear all

clc

close all

% clear all workspace and command window and close all figure

EbN0dB\_vector=0:2:20;

% make vector like vale 0 2 4 6 ... 20

Eb=1;

for snr\_i=1:length(EbN0dB\_vector) %repeat the loop for the length of vector

EbN0dB=EbN0dB\_vector(snr\_i); %load one by one value of the vector

EbN0=10.^(EbN0dB/10); %convert from db to back value

N0=Eb/EbN0; %decimal value

sym\_cnt=0; %counter for count the error and system count

err\_cnt=0;

h=sqrt(1/2)\*(randn+j\*randn);

while err\_cnt<500 %repeat the loop for count\_Error less then 500

s=sqrt(Eb)\*sign(rand-0.5); %creat the s is negtive or postive

% create h and n by calling randn funcation and sqrt use

n=sqrt(N0/2)\*(randn+j\*randn);

r=h\*s+n; %calculatte the value r

D=r\*exp(-j\*angle(h)); %calculate the angle and express in exp form

s\_hat=sign(D); %calculate real value

if s\_hat~=sign(s) %if the sign is differnet it error increment

err\_cnt=err\_cnt+1;

end

sym\_cnt=sym\_cnt+1; % otherwise system counter increment

end

BER(snr\_i)=err\_cnt/sym\_cnt %BER store the ratio b/w error count vs system count

end

% plot the vecter and BER value and label xaxis and yaxis

figure

semilogy(EbN0dB\_vector, BER)

xlabel('E\_b/N\_0 [dB]')

ylabel('BER')

grid

**Output**

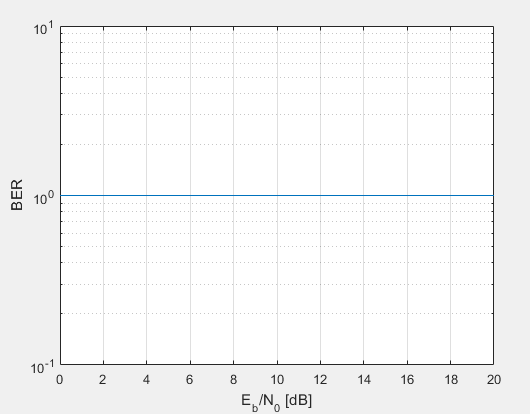


Figure : Output a-3

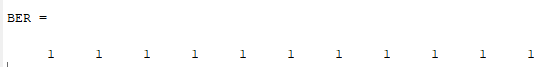


Figure ; BER a-3

**Explanation**

The output is same if we use the sign(D). Loop move toward infinity if we run for sign(real(D)).

## Part B

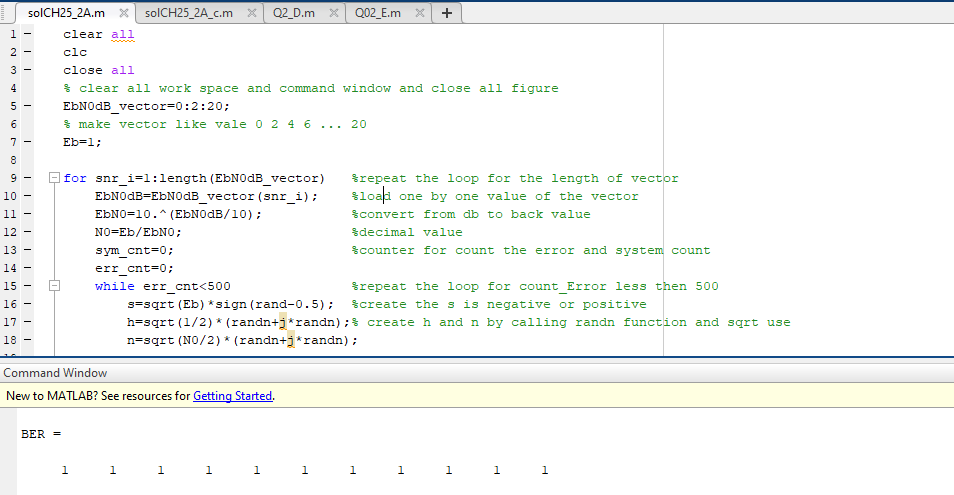


Figure : Q2 Part B

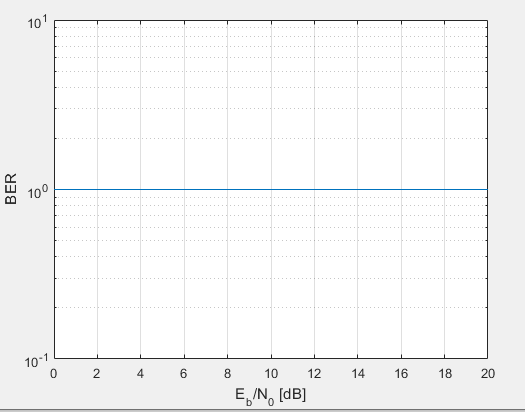


Figure : Q2 Part B Graph

## Part C

The output of BER is always become is 1. Because we don’t use the real value. Sign with complex number convert the value to 1.

## Part D

**Code**

clear all

clc

close all

% clear all workspace and command window and close all figure

EbN0dB\_vector=0:2:20;

% make vector like vale 0 2 4 6 ... 20

Eb=1;

for snr\_i=1:length(EbN0dB\_vector) %repeat the loop for the length of vector

EbN0dB=EbN0dB\_vector(snr\_i); %load one by one value of the vector

EbN0=10.^(EbN0dB/10); %convert from db to back value

N0=Eb/EbN0; %decimal value

sym\_cnt=0; %counter for count the error and system count

err\_cnt=0;

while err\_cnt<500 %repeat the loop for count\_Error less then 500

s=sqrt(Eb)\*sign(rand-0.5); %creat the s is negtive or postive

h=sqrt(1/2)\*(randn+j\*randn);% create h and n by calling randn funcation and sqrt use

n=sqrt(N0/2)\*(randn+j\*randn);

r=h\*s+n; %calculatte the value r

D=r\*exp(-j\*angle(h)); %calculate the angle and express in exp form

s\_hat=sign(real(D)); %calculate real value

if s\_hat~=sign(s) %if the sign is differnet it error increment

err\_cnt=err\_cnt+1;

end

sym\_cnt=sym\_cnt+1; % otherwise system counter increment

end

BER(snr\_i)=err\_cnt/sym\_cnt %BER store the ratio b/w error count vs system count

end

% plot the vecter and BER value and label xaxis and yaxis

figure

semilogy(EbN0dB\_vector, BER)

xlabel('E\_b/N\_0 [dB]')

ylabel('BER')

grid

**Output**

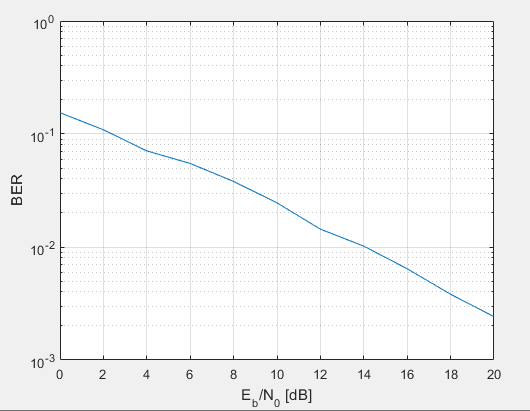


Figure : Q02 Part D Graph

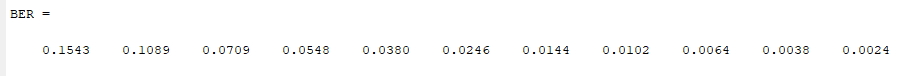


Figure : Q02 Part D

## Part E

**Code**

EbN0\_vector=10.^(EbN0dB\_vector/10);

BER\_theory=0.5\*(1-sqrt((EbN0\_vector)./(1+EbN0\_vector)));

hold on;

semilogy(EbN0dB\_vector, BER\_theory,'r')

legend('Rayleigh fading, Simulation','Rayleigh fading, Theory')

**Output**

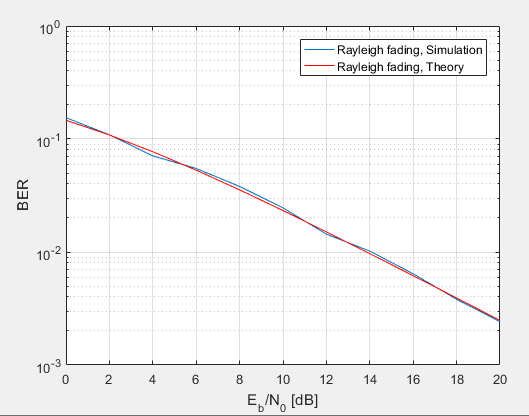


Figure : Part E

## Part F

Yes, the rayleigh fading simulation and theory are matching exactly. We can conformation we matching in graph mention above.

## Part G

**Code**

BER\_AWGN=0.5\*erfc(sqrt(EbN0\_vector));

hold on;

semilogy(EbN0dB\_vector, BER\_AWGN,'g')

legend('Rayleigh fading, Sim','Rayleigh fading, Theory' ,'AWGN')

axis([0 20 1e-6 1])

**Output**

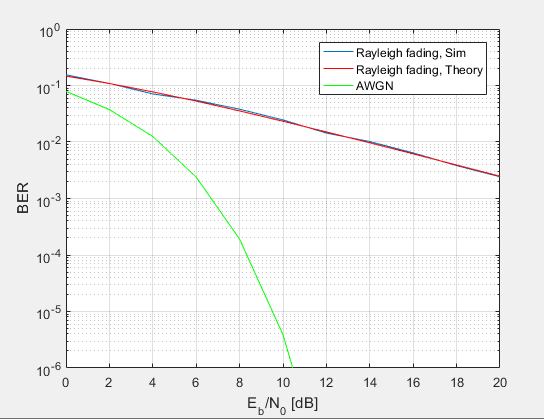


Figure : Part G

## Part H

The performance of the BER is lower as compere to AWGN. For more detail we can see the graph mention above.

## Part I

The receiver power of the both are same in BER and AWGN. But the performance is not same because we notify the result in previous graph.

## Part J

### Part J-1

clear

EbN0dB\_vector=0:2:20;

Eb=1;

for snr\_i=1:length(EbN0dB\_vector)

EbN0dB=EbN0dB\_vector(snr\_i);

EbN0=10.^(EbN0dB/10);

N0=Eb/EbN0;

sym\_cnt=0;

err\_cnt=0;

while err\_cnt<500 %modification done in this loop

s=sqrt(Eb)\*sign(rand-0.5)+j\*sqrt(Eb)\*sign(rand-0.5);

h=sqrt(1/2)\*(randn+j\*randn);

n=sqrt(N0/2)\*(randn+j\*randn);%get the s h and n value as same

r=h\*s+n; %calculate the value of r

D=r\*exp(-j\*angle(h));

s\_hat=sign(real(D)); %take real part from complex number

s\_hat2=sign(imag(D)); %take img part

if sign(real(s)) ~= s\_hat % error increase real value

err\_cnt=err\_cnt+1; %

end

if sign(imag(s)) ~= s\_hat2 % error increase based on angle

err\_cnt=err\_cnt+1;

end

sym\_cnt=sym\_cnt+2; % system counter by 2 because now 2 condition for error

end

BER(snr\_i)=err\_cnt/sym\_cnt %BER display

end

figure

semilogy(EbN0dB\_vector, BER)

xlabel('E\_b/N\_0 [dB]')

ylabel('BER')

grid

### Part J-2

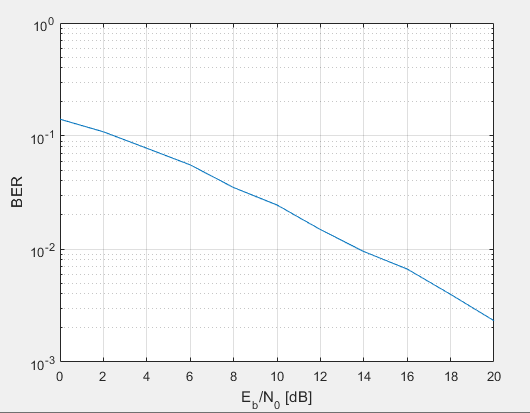


Figure : Part J Graph

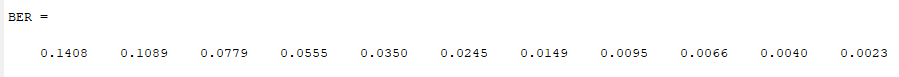


Figure : BER Part J

### Part J-3

BER are cure are same in both case QPSK and BPSK. We can both graph is plot above in part-j and part-e. Both are same because in QPSK are deal with separate real and imaginary part. But in the BPSK we deal with only sign and real value based. If we deal with one parameter we increment the system increment by 1 and same if both condition apply then increment by 2.

# Question 04

## Part A: Selection Diversity Combining

### Part A-1

clear

close all

clc

EbN0dB\_vector=0:3:15; %make vector by the diff value 3

Eb=1;

L=3;

for snr\_i=1:length(EbN0dB\_vector) %run the loop for length of vector

EbN0dB=EbN0dB\_vector(snr\_i); %one by one access the vector

EbN0=10.^(EbN0dB/10);

N0=Eb/EbN0; %db to decimal

sym\_cnt=0;

err\_cnt=0; %reset the both counter

while err\_cnt<100 % If you increase err\_cnt (currently 100), the accuracy increases and time also increase.

b=sign(rand-0.5); %BPSK symbol {1,-1}

s=sqrt(Eb/L)\*b;

for k=1:L

h(k)=sqrt(1/2)\*(randn+j\*randn);

n(k)=sqrt(N0/2)\*(randn+j\*randn);

r(k)=h(k)\*s + n(k);

end

[T1 T2]=max(r); % Refer to (25.9). To see how to use max( ).

D=r(T2)\*exp(-j\*angle(h(T2))); %Refer to (25.9).

b\_hat=sign(real(D));

if b\_hat~=b;

err\_cnt=err\_cnt+1;

end

sym\_cnt=sym\_cnt+1;

end

BER(snr\_i)=err\_cnt/sym\_cnt

end

figure

semilogy(EbN0dB\_vector, BER)

xlabel('E\_b/N\_0 [dB]')

ylabel('BER')

grid

### Part A-2

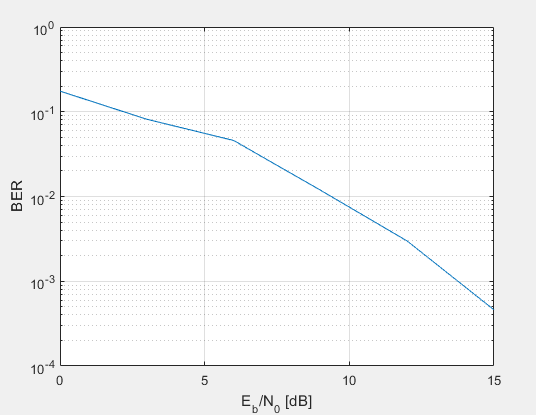


Figure : Q04 A Graph

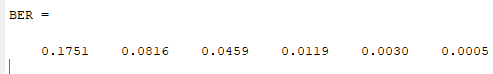


Figure : Q04 A

### Part A-3

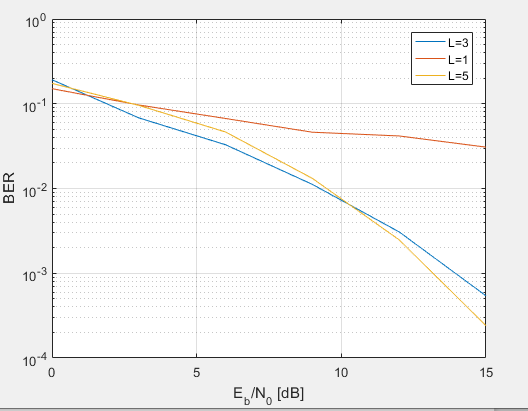


Figure : L=3, L=1, L=5

### Part A-4

**A)**

In the BER we see the L use in the value of s. As we increase the order L cure move toward down. By increase the order we can say error decrease and system work efficiently.

**B)**

The SNR depends on the order. if the order is lower then SNR case is worst and it show in the graph. This one only reason due to inversely proportional relation.

## Part B: Equal Gain Combining

### Part B-1

clc

clear all,close all

EbN0dB\_vector=0:3:15; %make vector by the diff value 3

Eb=1;

L=3;

for snr\_i=1:length(EbN0dB\_vector) %run the loop for length of vector

EbN0dB=EbN0dB\_vector(snr\_i); %one by one access the vector

EbN0=10.^(EbN0dB/10);

N0=Eb/EbN0; %db to decimal

sym\_cnt=0;

err\_cnt=0; %reset the both counter

while err\_cnt<100 % If you increase err\_cnt (currently 100), the accuracy increases and time also increase.

b=sign(rand-0.5); %BPSK symbol {1,-1}

s=sqrt(Eb/L)\*b;

for k=1:L

h(k)=sqrt(1/2)\*(randn+j\*randn);

n(k)=sqrt(N0/2)\*(randn+j\*randn);

r(k)=h(k)\*s + n(k);

end

[T1 T2]=max(r); % Refer to (25.9). To see how to use max( ).

% D=r(T2)\*exp(-j\*angle(h(T2))); %Refer to (25.9).

D=sum(r.\*exp(-j\*angle(h)));

b\_hat=sign(real(D));

if b\_hat~=b;

err\_cnt=err\_cnt+1;

end

sym\_cnt=sym\_cnt+1;

end

BER(snr\_i)=err\_cnt/sym\_cnt

end

semilogy(EbN0dB\_vector, BER)

xlabel('E\_b/N\_0 [dB]')

ylabel('BER')

grid

### Part B-2

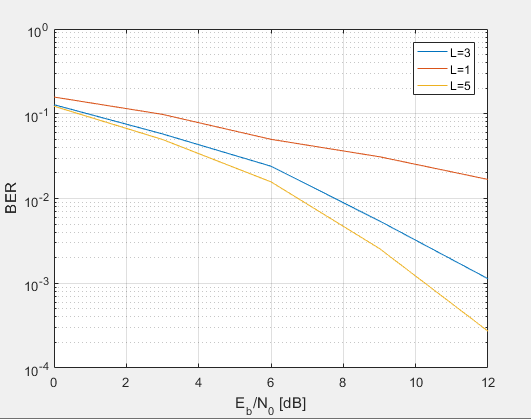


Figure : Q04 B Graph

### Part B-3

By increasing the order of L BER decrease because SNR and L are in reversely proportional.

### Part B-4

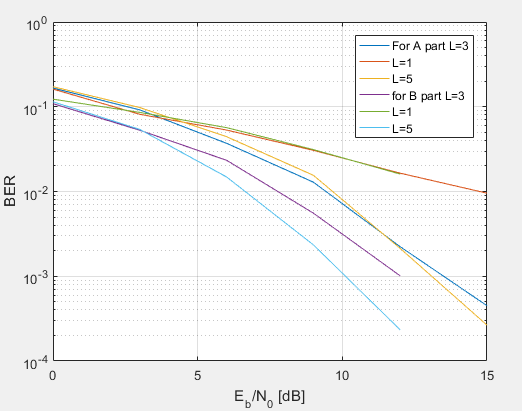


Figure : Compare Six Graph

## Part C: Maximum Ratio Combining

### Part C-1

for the value of D we use both equation both are same. First one in polar form and second one is rectangular form.

### Part C-2

clc

clear all

close all

EbN0dB\_vector=0:3:15; %make vector by the diff value 3

Eb=1;

L=3;

for snr\_i=1:length(EbN0dB\_vector) %run the loop for length of vector

EbN0dB=EbN0dB\_vector(snr\_i); %one by one access the vector

EbN0=10.^(EbN0dB/10);

N0=Eb/EbN0; %db to decimal

sym\_cnt=0;

err\_cnt=0; %reset the both counter

while err\_cnt<100 % If you increase err\_cnt (currently 100), the accuracy increases and time also increase.

b=sign(rand-0.5); %BPSK symbol {1,-1}

s=sqrt(Eb/L)\*b;

for k=1:L

h(k)=sqrt(1/2)\*(randn+j\*randn);

n(k)=sqrt(N0/2)\*(randn+j\*randn);

r(k)=h(k)\*s + n(k);

end

[T1 T2]=max(r); % Refer to (25.9). To see how to use max( ).

% D=r(T2)\*exp(-j\*angle(h(T2))); %Refer to (25.9).

D=sum(conj(h).\*r);

b\_hat=sign(real(D));

if b\_hat~=b;

err\_cnt=err\_cnt+1;

end

sym\_cnt=sym\_cnt+1;

end

BER(snr\_i)=err\_cnt/sym\_cnt

end

hold on

semilogy(EbN0dB\_vector, BER)

xlabel('E\_b/N\_0 [dB]')

ylabel('BER')

grid

### Part C-3

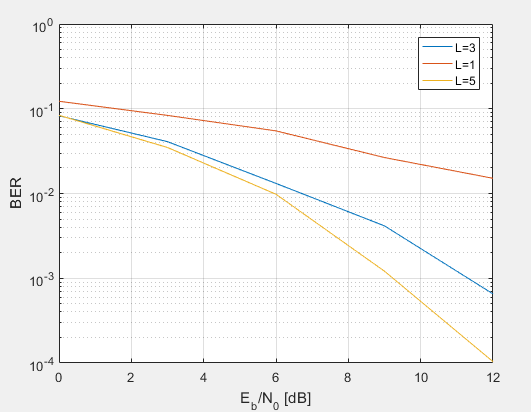


Figure : Q04 C Graph

### Part C-4

BER and slop of BER are change according to the order L As we increase the order L BER slop cure change and low error occur.

### Part C-5

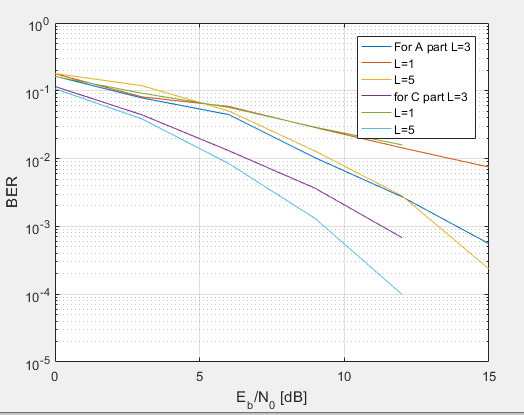


Figure : Compare Six Graph

## Part D: Compare the Performance

### Part D-1

A)

Best perform the MRC.

B)

At the order of 5, SNR is lowest and BER is low as compare to other order.

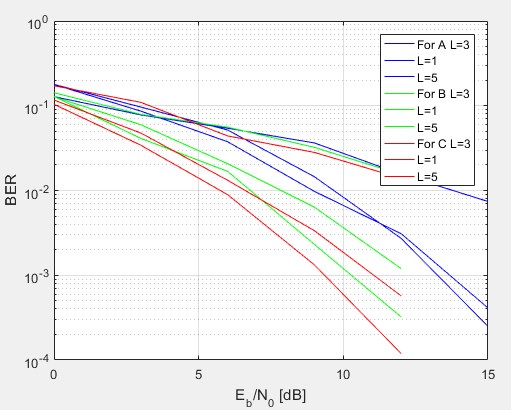


Figure : Nine graph

### Part D-2

The lowest complexity is SDC because we just compute the equation one time. The largest complexity is MRC because in the MRC we need to compute the sum and conjugate as well. In other term first, we conjugate and then compute the sum of the series.

## Part E: Investigate the Diversity Gain

### Part E- 1

Common trend is that by increase the order L the diversity gain EBR is decrease all method.

### Part E-2

clc

clear all

close all

EbN0dB\_vector=0:3:15; %make vector by the diff value 3

Eb=1;

L=5;

for snr\_i=1:length(EbN0dB\_vector) %run the loop for length of vector

EbN0dB=EbN0dB\_vector(snr\_i); %one by one access the vector

EbN0=10.^(EbN0dB/10);

N0=Eb/EbN0; %db to decimal

sym\_cnt=0;

err\_cnt=0; %reset the both counter

while err\_cnt<100 % If you increase err\_cnt (currently 100), the accuracy increases and time also increase.

b=sign(rand-0.5); %BPSK symbol {1,-1}

s=sqrt(Eb/L)\*b;

% for k=1:L

% h(k)=sqrt(1/2)\*(randn+j\*randn);

% n(k)=sqrt(N0/2)\*(randn+j\*randn);

% r(k)=h(k)\*s + n(k);

% end

hk=sqrt(1/2)\*(randn+j\*randn);

for k=1:L

h(k)=hk;

n(k)=sqrt(N0/2)\*(randn+j\*randn);

r(k)=h(k)\*s+n(k);

end

[T1 T2]=max(r); % Refer to (25.9). To see how to use max( ).

D=r(T2)\*exp(-j\*angle(h(T2))); %Refer to (25.9).

b\_hat=sign(real(D));

if b\_hat~=b;

err\_cnt=err\_cnt+1;

end

sym\_cnt=sym\_cnt+1;

end

BER(snr\_i)=err\_cnt/sym\_cnt

end

semilogy(EbN0dB\_vector, BER,'b')

xlabel('E\_b/N\_0 [dB]')

ylabel('BER')

grid

### Part E-3

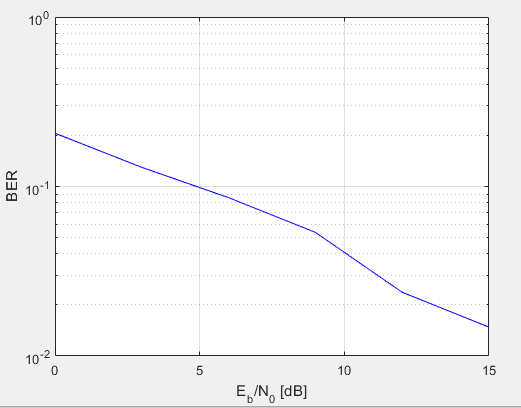


Figure : Q04 E Graph

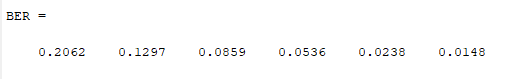


Figure : Q04 E

### Part E-4

A)

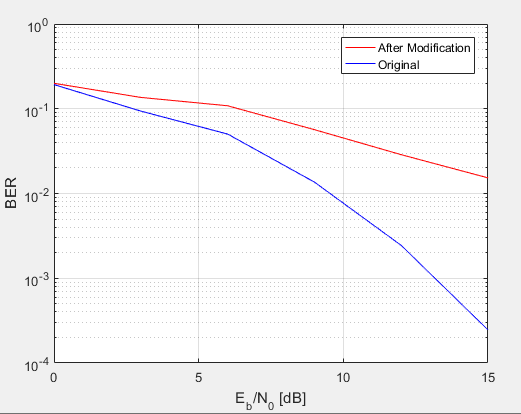


Figure : Compare of modified and original

B)

Both cure is different because h(k) is not same. In original file h(k) compute every time and in the modify code h(k) assign with same value that is compute one time.

### Part E-5

The modify code have the same coefficients because we reuse the only one value. In the given sample code every time coefficient executes and this one is different. The diversity move toward maximize if increase the order and same if we increase the order then able to find the minimize diversity gain.