Analog and Digital Communication Systems (ECE351s)

Under the supervision of:

Dr. Bassant Abdelhamid
Eng Mariz Ashraf

abdallah karim motwea	2000993	CONTRIBUTED WITH 33.33% OF THE CODE IN THE TWO PARTS
abdelrahman khaled amin mekkawy	1600784	CONTRIBUTED WITH 33.33% OF THE CODE IN THE TWO PARTS
Abdulaziz Hamad Albdowi	2002265	CONTRIBUTED WITH 33.33% OF THE CODE IN THE TWO PARTS

MATLAB CODE

```
ploting_signals_fuunction(S,T);
ploting_basis_function(S,T);
ploting_constellation_function(S,T);
Number_of_bits=10^5;   %Number of bits stream
In_stream_bits=randi([0 1],Number_of_bits,1); %random stream generation of zeroes
and ones
%Stream of bits encoding into binary Polar NRZ %
for i=1:Number_of_bits
    if In_stream_bits(i)==1
        In_stream_bits(i)=1;
    else
       In_stream_bits(i)=-1;
    end
end
P_NRZ_stream=In_stream_bits;
Eb_No_dB = [-10 -8 -6 -4 -2 0 2 4 6];  %Eb/No
Eb_No = 10.^(Eb_No_dB/10);
SNR = 2*Eb_No;
SNR_dB = pow2db(SNR); %SNR = 2*Eb/No
 %Transmit the encoded stream through AWGN channel
for i=1:length(Eb_No_dB)
  Received_bits(i,:) = awgn(P_NRZ_stream(:) ,SNR_dB(i),'measured');
end

S_ij=zeros(length(Eb_No_dB),length(P_NRZ_stream));
for i=1:length(Eb_No_dB)
    for j=1:Number_of_bits
    S_ij(i,j)=Received_bits(i,j);
    end
end
y = zeros(1,Number_of_bits);

for i=1:length(Eb_No_dB)
    figure(9+i)
    %subplot(length(Eb_No_dB),1,i);
    scatter(S_ij(i,:),y,'c x','LineWidth',0.5);
    xlim([min(S_ij(1,:))-1   max(S_ij(1,:))+1]);
    xlabel('Real');
    ylabel('Imagnary')
    grid on;
    title(sprintf('Eb/No = %d dB',Eb_No_dB(i)));
 end

P_NRZ_Decoded_Signal=zeros(length(Eb_No_dB),Number_of_bits);
for i=1:length(Eb_No_dB)
   for j=1:Number_of_bits
     if Received_bits(i,j)>0
       P_NRZ_Decoded_Signal(i,j)=1;
          else
       P_NRZ_Decoded_Signal(i,j)=-1;
     end
```

```
  end
end

for i =1:length(Eb_No_dB)
 No_error_bit(i) = 0;
  for j=1:Number_of_bits
    if P_NRZ_Decoded_Signal(i,j) ~= P_NRZ_stream(j) % checking for error bits
      No_error_bit(i) = No_error_bit(i)+1;
    end
  end
BER_measured(i) = No_error_bit(i)/Number_of_bits;
end
BER_measured    %Measured bit error rate
BER_theortical = 1/2.*erfc(sqrt(Eb_No))  %Theoretical bit error rate

figure
semilogy(Eb_No_dB,BER_measured,'k x -')
grid
hold
     ylabel('Bit error rate');
     xlabel('Eb/No (dB)');
     title('Measured &Theortical BER vs Eb/No')
semilogy(Eb_No_dB,BER_theortical,'g + --')
legend('Measured BER','Theortical BER')

function ploting_signals_fuunction(S,T)% plot the signals as pulses with pulse du-
ration T
M=size(S,1); %M is the number of signals
for i=1:1:M
    figure
plot_pulses(S(i,:),T) %plot all signals in case of pulse signals
ylabel(['S ',num2str(i)]);
xlabel('t');
end
end

function ploting_basis_function(S,T) % plot the basis functions as pulses with
pulse duration T
phi=basis_func(S,T);
N_basis(phi);
M=size(S,1);
for i=1:1:M
       %if (isnan(phi(i,:))~=true(1,length(phi(i,:))))
               figure
               plot_pulses(phi(i,:),T)
               ylabel(['phi ',num2str(i)]);
               xlabel('t');
       %end
end
end
```

```
function plot_pulses(S,T) % it takes a (vector) of pulse amplitudes, turns it into
a stream of pulses and plots them

    L=length(S); %L counts the number of the signals
    dt=T/1000;
    fs=T/dt;
    stream = zeros(1,L*fs); % stream of Zeros along the figure

    for i=1:1:L
        if i==1
            stream(1)=S(i);
        end
        stream((i-1)*fs+2:(i*fs))=S(i).*ones(1,fs-1);
        if i==L
            stream(i*fs+1)=0;
        else
            stream(i*fs+1)=(S(i)+S(i+1))/2;
        end
    end

    t=[0:dt:T*L];
    plot(t,stream) %plot all the signal as a stream of continuous pulses

end

function E=E_calc(V,T)
    E=sum((V.^2)*T);    %Calculate Energy of the pulse signal
end

function basiss_func=basis_func(S,T)% it takes a matrix of the signal (pulse) am-
plitudes, their pulse duration to generate a matrix of phi amplitudes with the same
pulse duration

L=length(S(1,:));
M= size (S,1);
N=0;
phi=zeros(M,L);   % initialize phi
g=zeros(1,L);   % initialize g
phi(1,:)=S(1,:)/sqrt(E_calc(S(1,:),T));

for i=2:1:M

    g=S(i,:);
    for j=1:1:i-1

        S_ij=sum(S(i,:).*phi(j,:)*T); % Sij=integration(S_i*phi_j)
        g=g-(S_ij.*phi(j,:));

    end

        phi(i,:)=g/sqrt(E_calc(g,T));
end
basiss_func=phi;
```

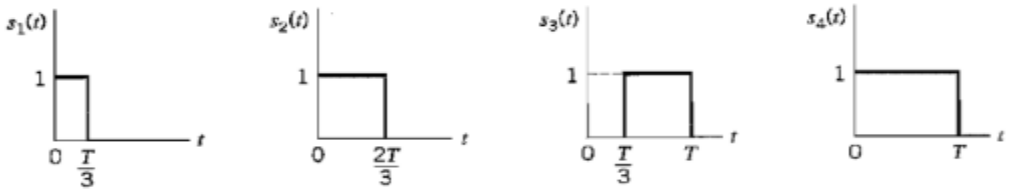
```
end

function phiess=phies(phi) % it produces a matrix of phies (row1=phi1,row2=phi2)
without any NAN (0/0) values by giving it the phi matrix that has NAN values
   M=size(phi,1);
   j=0;
   for i=1:1:M
       if (isnan(phi(i,:))~=logical(ones(1,length(phi(i,:))))) % don't count all
NAN (0/0) rows (unused phies)
              j=j+1;
              phiess(j,:)=phi(i,:);
       end
    end
end

function N_basiss=N_basis(phi) %calculates the number of basis functions N that
will be used in the constillation diagram
    N_basiss=size(phies(phi),1);
end

function S_ij=Sij(S,phi,T) % generates a matrix of S in terms of phies the 1st row
--> S11, the 2nd row --> S12 ,S22  etc

L=length(S(1,:));
N=N_basis(phi);   % the number of the matrix colomns = the number of basis funcs
M= size (S,1);    % the number of the matrix rows = the number of the signals
S_ij=zeros(M,N);   % initialise the matrix with zeros

for i=1:1:M

    for j=1:1:i

        if (isnan(phi(j,:))==logical(ones(1,length(phi(i,:)))))
            phi(j,:)=zeros(1,L);     %all NAN (0/0) values will be replaced with
zeros
        end

        S_ij(i,j)=sum(S(i,:).*phi(j,:)*T);  % Sij=integration(S_i*phi_j)
    end
end

end

function ploting_constellation_function(S,T)
phi=basis_func(S,T);%Basic functions as a matrix
s = Sij(S,phi,T);
columnNumbers = find(sum(abs(s)) == 0); %return the coulumn number which has a zero
in all rows of it
if columnNumbers >= 1 %if any column in the matrix of phi = 0
L = length(s)-1; %Number of colums will decrease by one
else L = length(s); % the length will be the same if no column in the matrix of phi
= 0
end
```

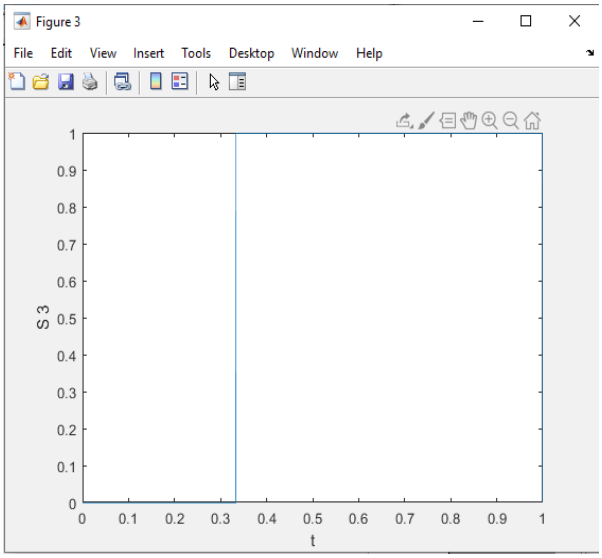
```
if L == 1 % if we have only one basic function
     scatter(s(1:end,1),s(1:end,columnNumbers));
     xlabel('phi1(t)');
     title('Ploting_constellation_function Diagram')
else if L == 2  % if we have Two basic functions
     scatter(s(1:end,1),s(1:end,2));
     ylabel('phi2(t)');
     xlabel('phi1(t)');
     title('Ploting_constellation_function Diagram')
else if L == 3   % if we have Three basic functions
     scatter3(s(1:end,1),s(1:end,2),s(1:end,3));
     zlabel('phi3(t)');
     ylabel('phi2(t)');
     xlabel('phi1(t)');
     title('Ploting_constellation_function Diagram')
    else
     scatter(s(1:end,1),zeros(length(s),1));
     title('Ploting_constellation_function Diagram')
end

M = length(s); % Number of M signals
E=zeros(1,M);
for i = 1:1:M
  E(i) = sum(s(i,1:end).^2); %Calculating Energy of each symbol from the Matrix of
the constillation Diagram
end
Energy=E
end

end
end
```

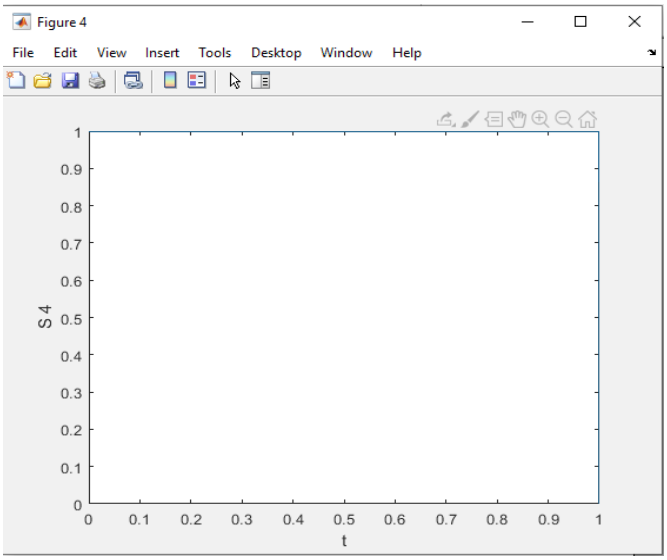
PART 1:

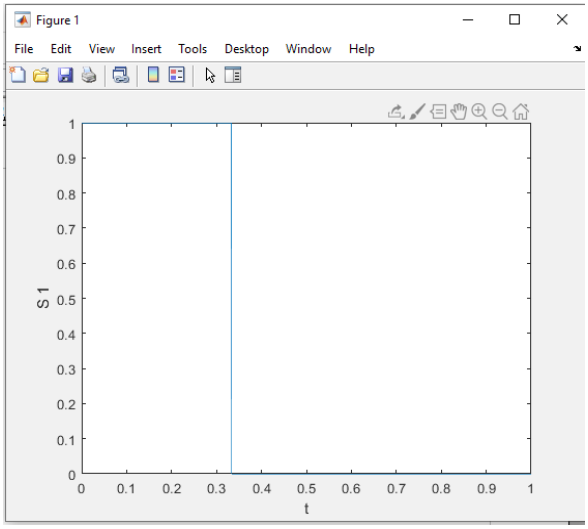
Test case 1:

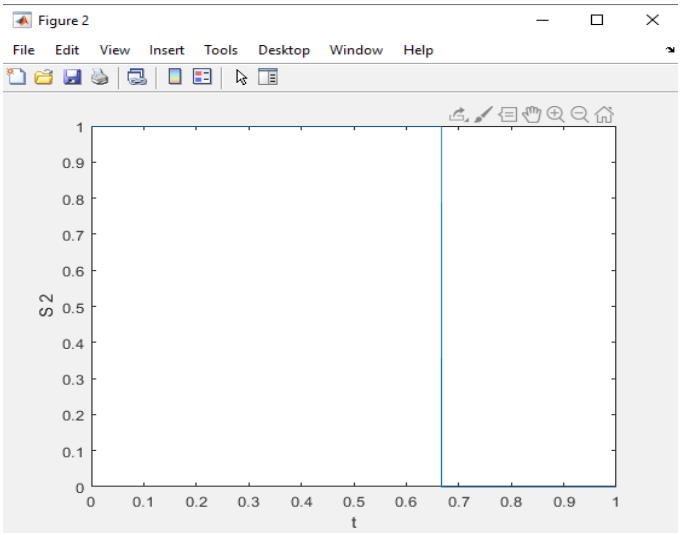
Inputs :



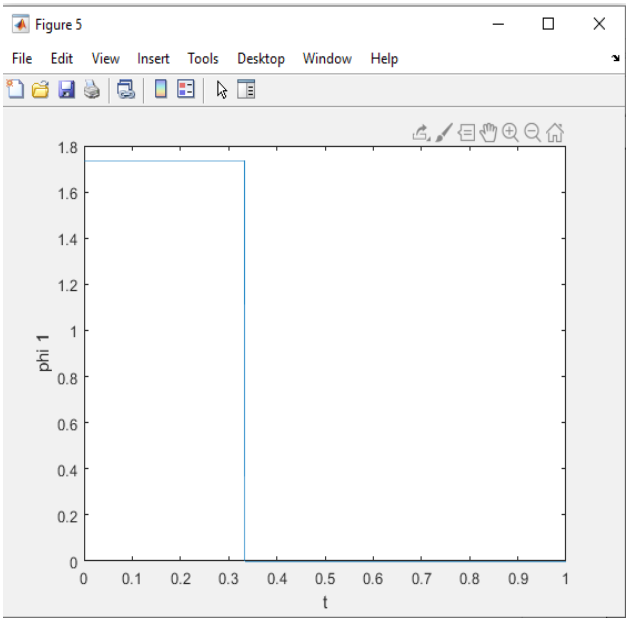


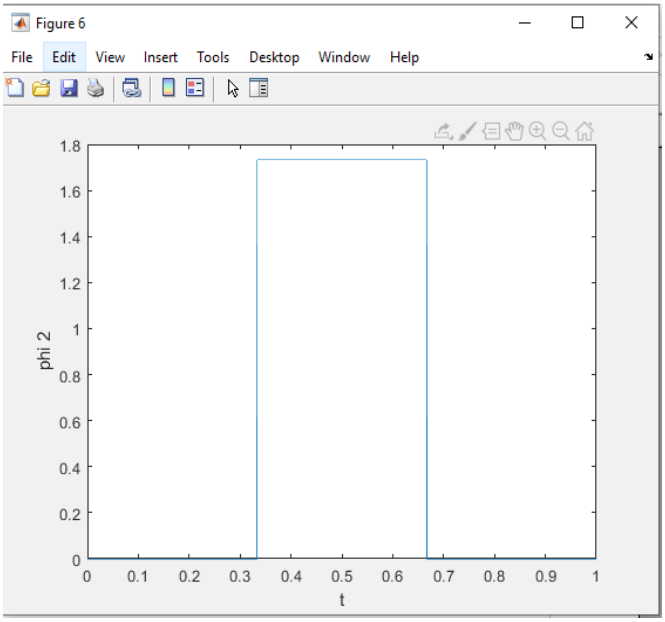


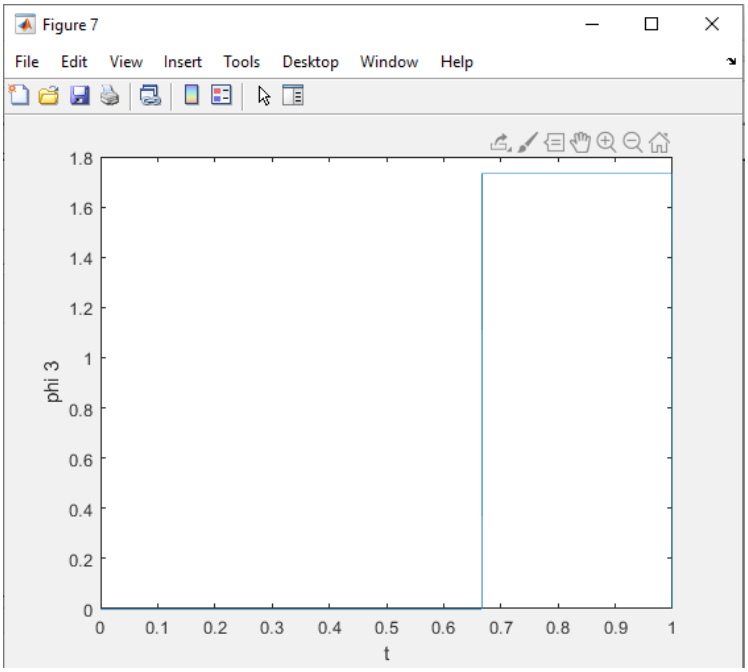




Basis functions:

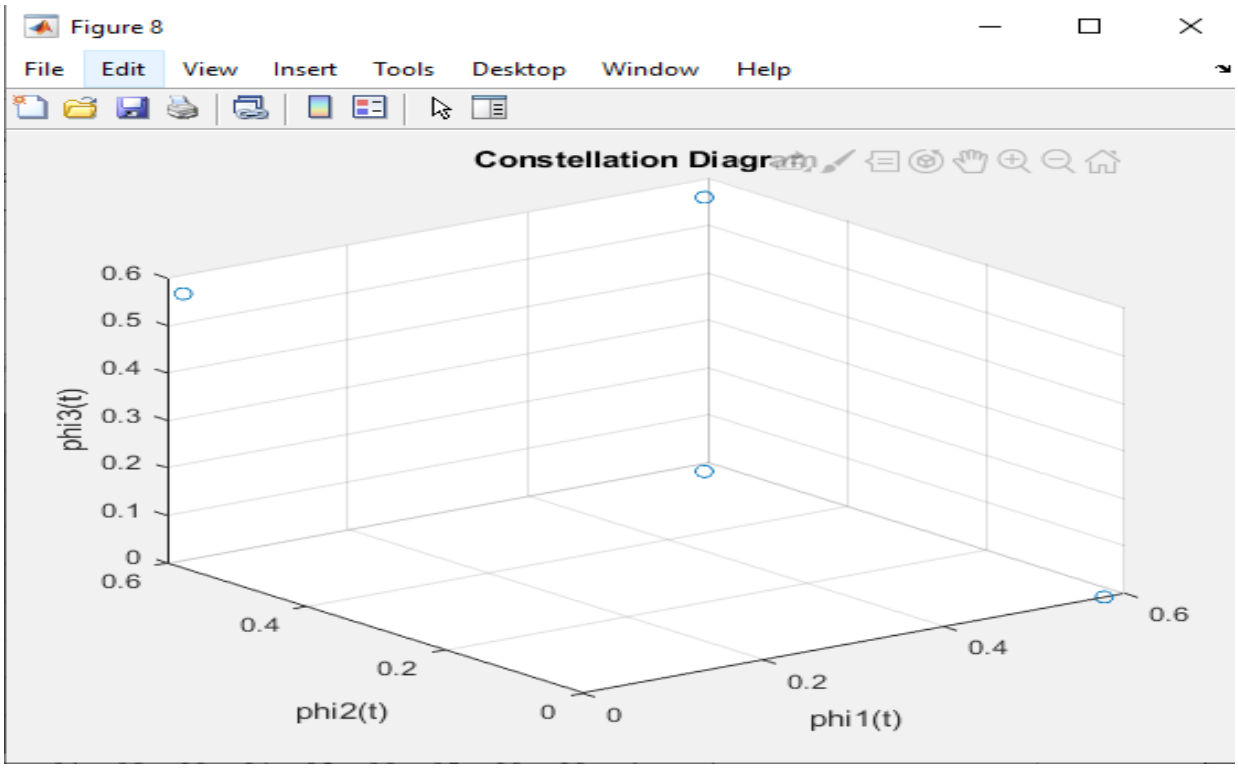






No phai4 is found, as symbols (S1,S3,S4) have linear relation between them.

Ploting_constellation_function diagram:

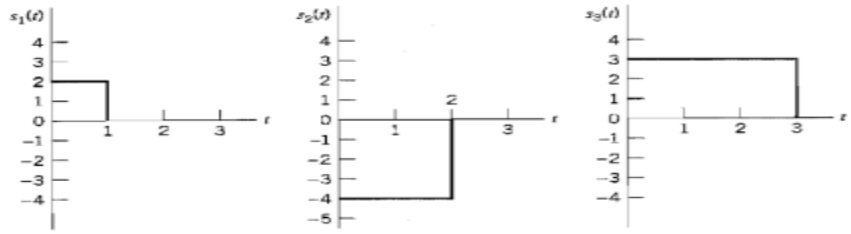


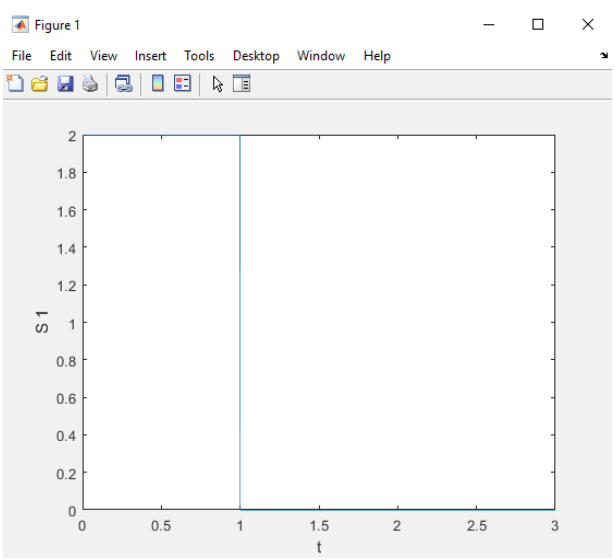
Signals energy:

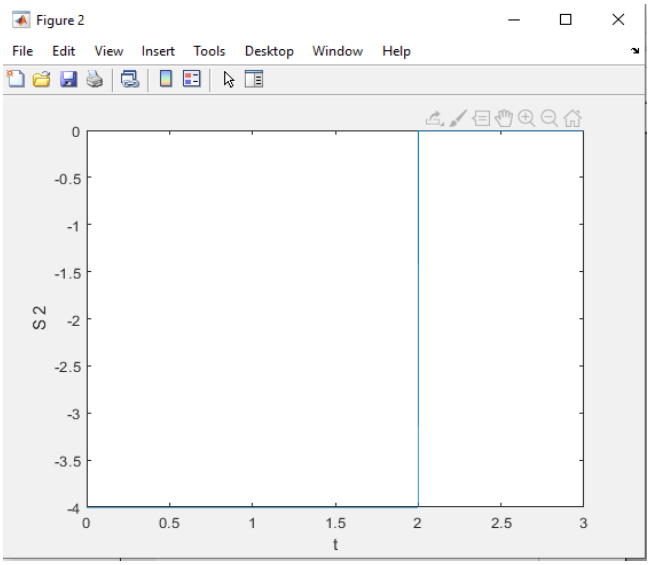
```
Energy =

   0.333333333333333   0.666666666666667   0.666666666666667   1.000000000000000
```

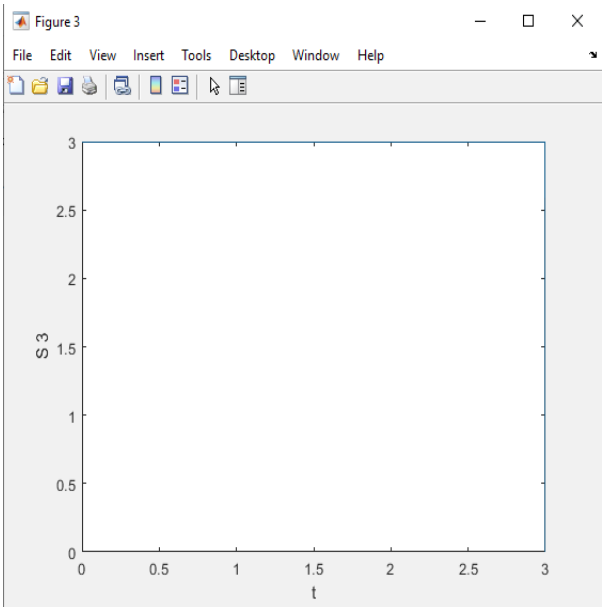
Test case 2:

Inputs :

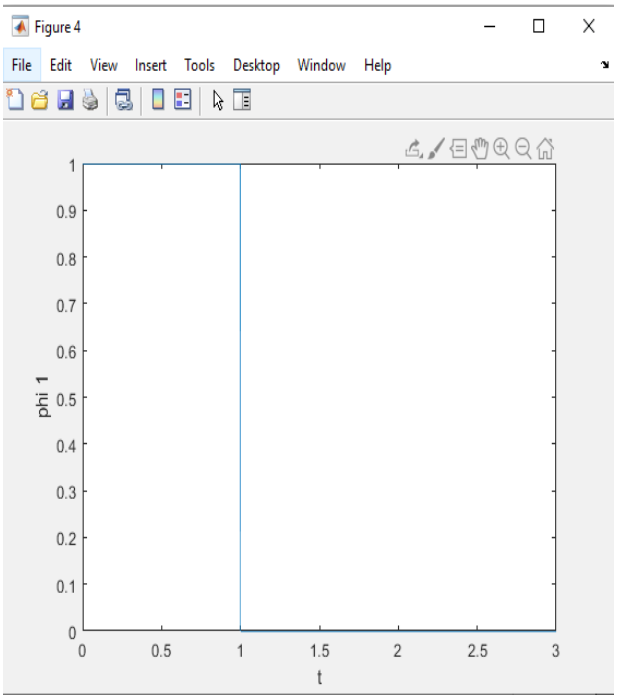


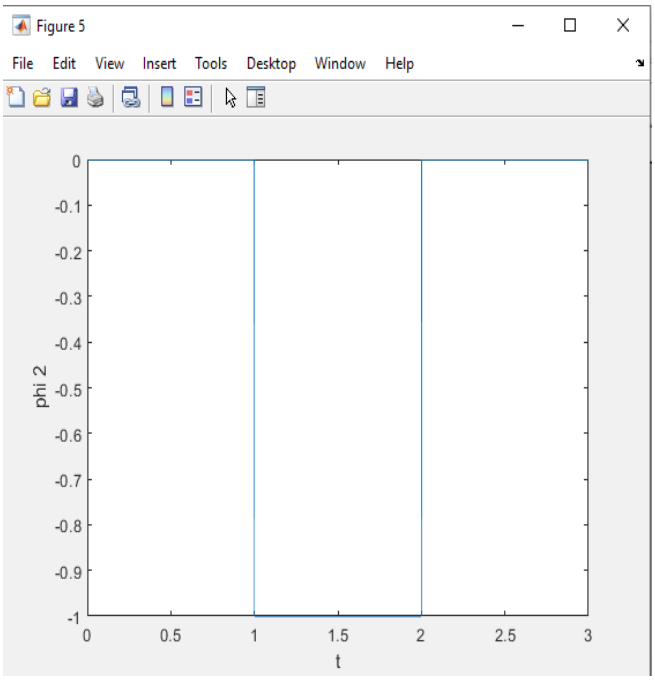


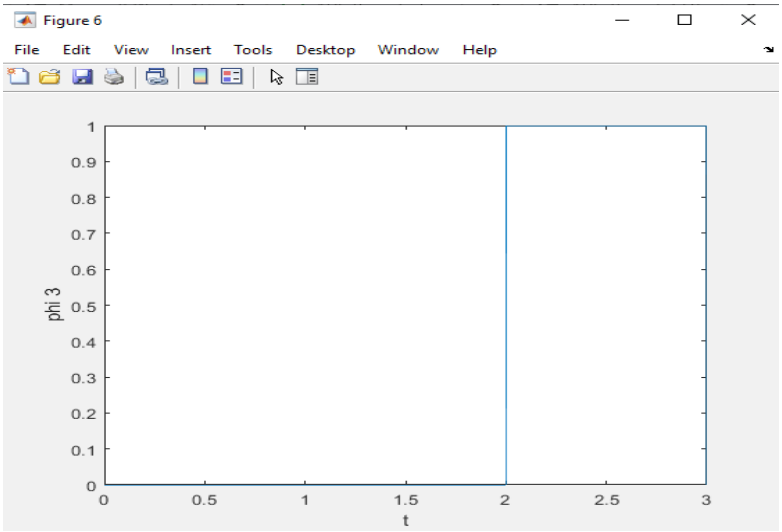




Basis functions:

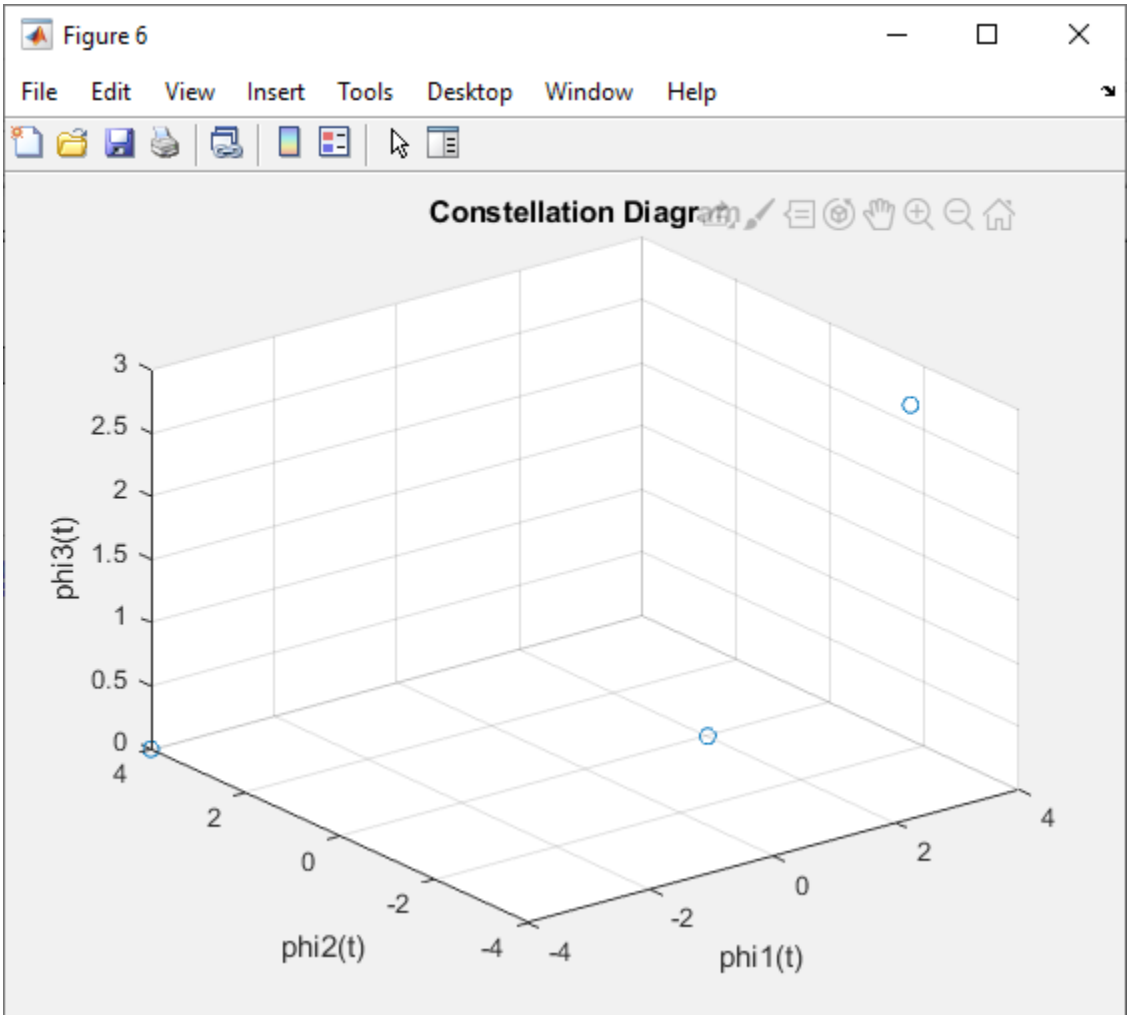






All phai's exist as there's no linear relation between symbols.

Ploting_constellation_function diagram:

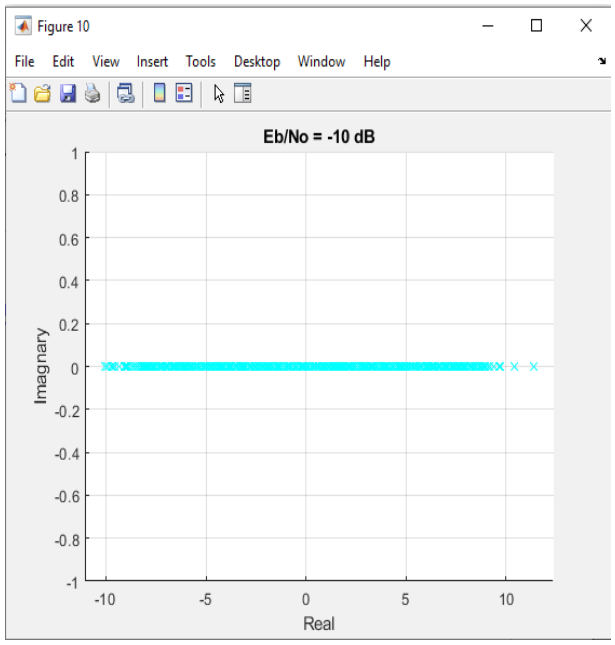


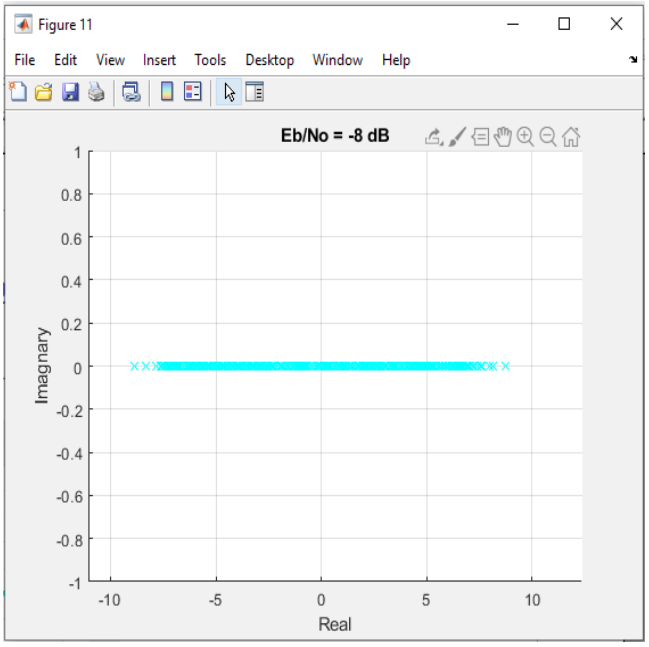
Signals energy:

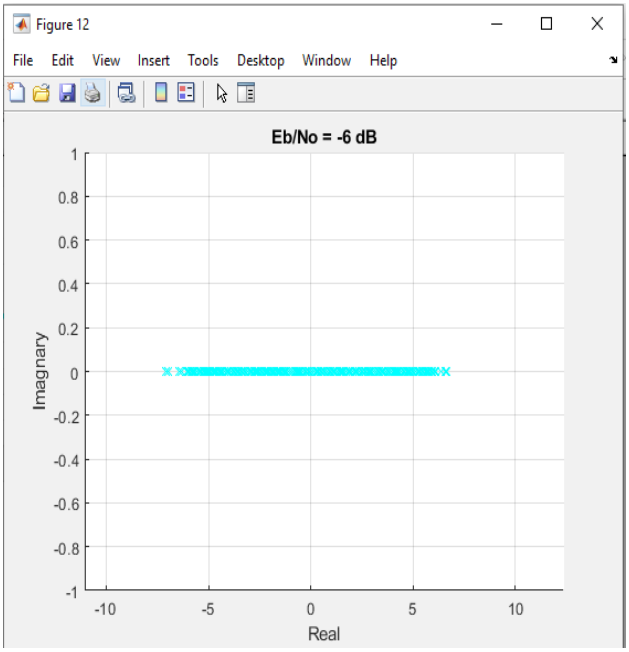
```
Energy =

     4    32    27
```

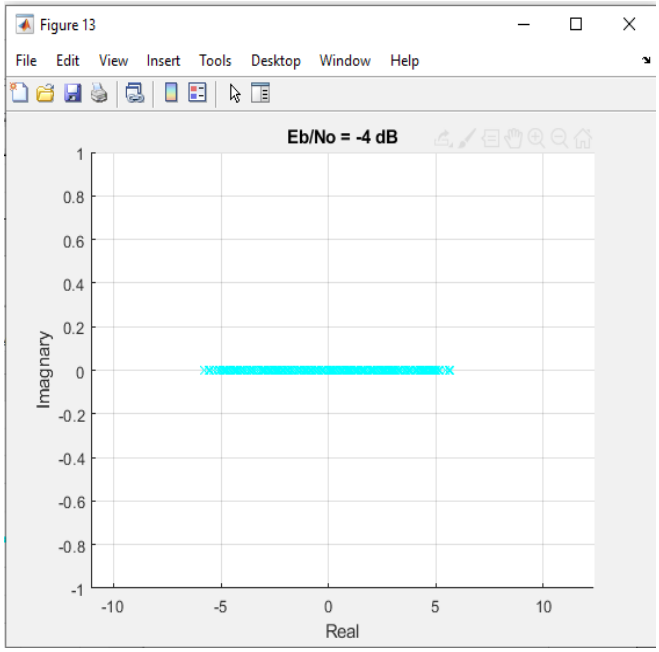
PART 2:

The plot of relation between the BER of decoded data and Eb/No:









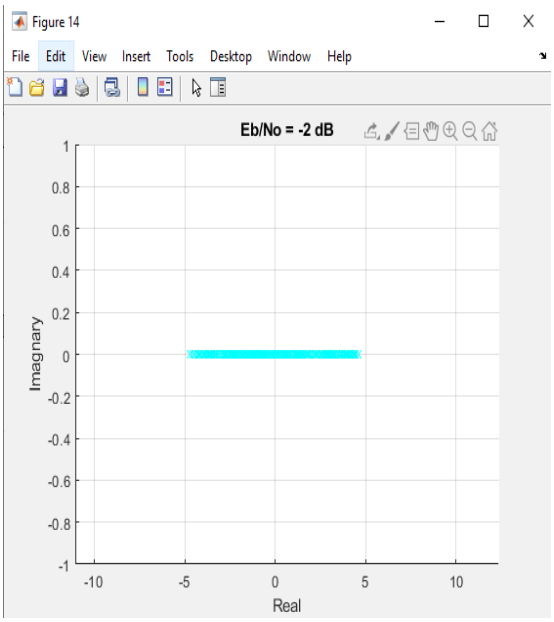


Figure 14
File Edit View Insert Tools Desktop Window Help
Eb/No = -2 dB
Imagnary
Real

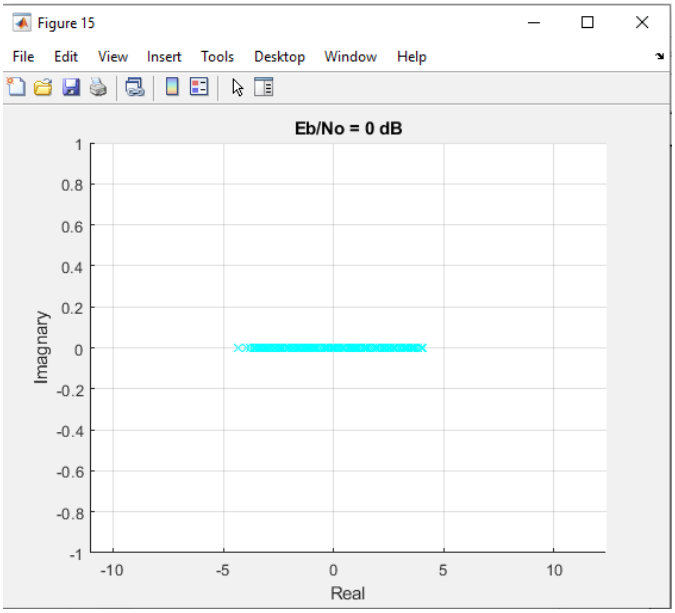


Figure 15
File Edit View Insert Tools Desktop Window Help
Eb/No = 0 dB
Imagnary
Real

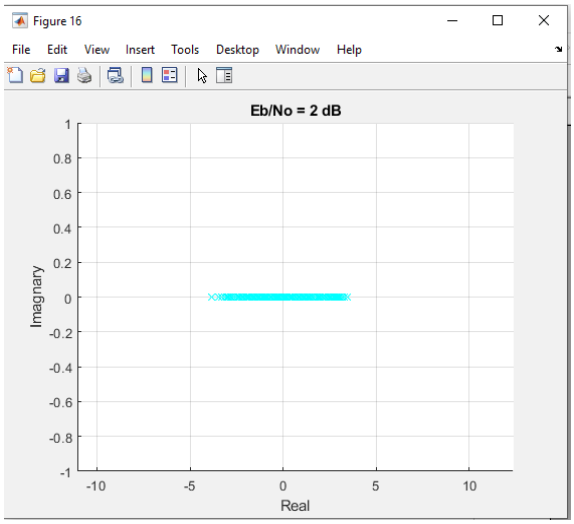


Figure 16
File Edit View Insert Tools Desktop Window Help
Eb/No = 2 dB
Imagnary
Real

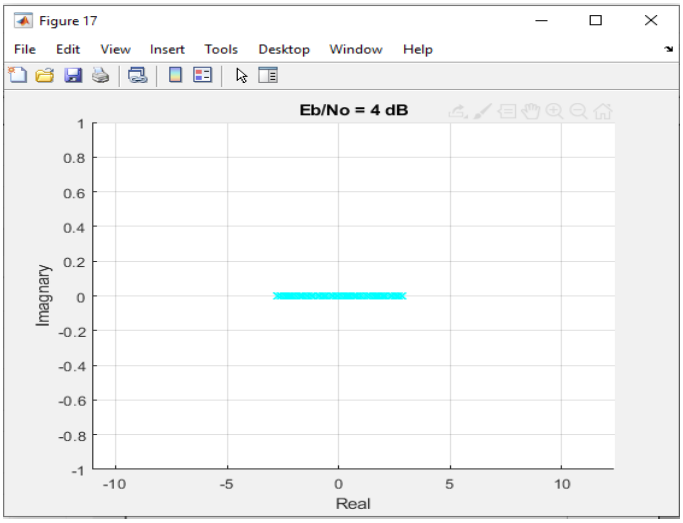


Figure 17
File Edit View Insert Tools Desktop Window Help
Eb/No = 4 dB
Imagnary
Real

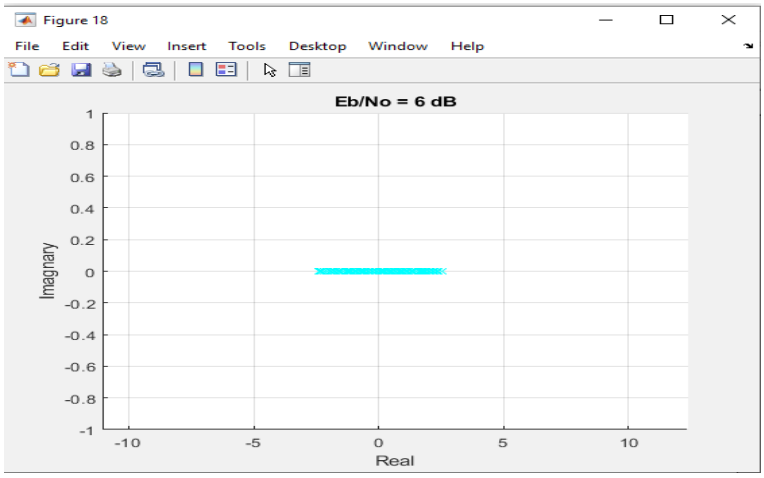
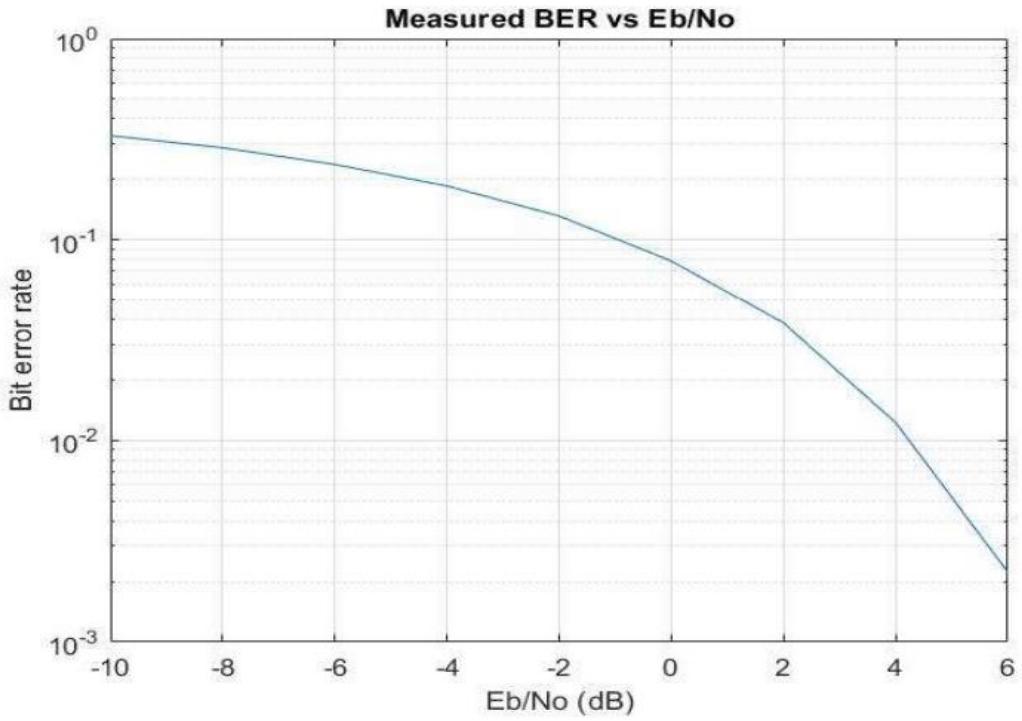


Figure 18
File Edit View Insert Tools Desktop Window Help
Eb/No = 6 dB
Imagnary
Real

Decode the received data plot the relation between the BER of decoded



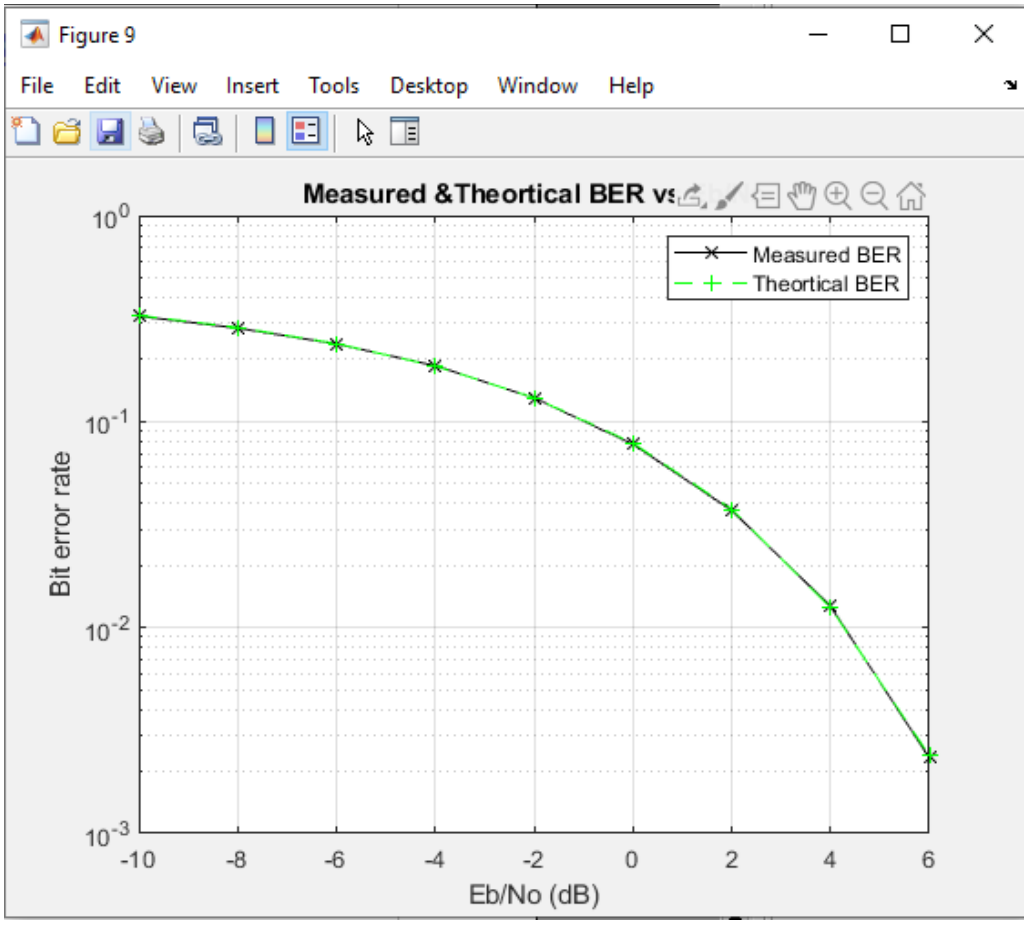
data and $\frac{E_b}{N_o}$

BER_measured

1x9 double

	1	2	3	4	5	6	7	8	9	10	11	12	1
1	0.3276	0.2866	0.2397	0.1843	0.1291	0.0785	0.0385	0.0125	0.0023				

Derive theoretically the BER of polar NRZ in terms of Eb/No and plot the theoretical BER vs Eb/No in dB and compare it with the results from part 4 on the same curve. The plot of relation between the BER of decoded data and Eb/No:



BER_measured | BER_theortical

1x9 double

	1	2	3	4	5	6	7	8	9	10	11	12
1	0.3274	0.2867	0.2392	0.1861	0.1306	0.0786	0.0375	0.0125	0.0024			

From the numbers and the curves we can see that measured and theoretical BER are very nearly the same which is because we have big number of symbols Also, by increasing SNR the BER decreases which is predicted as the signal power increases, the noise effect decreases and also error decreases.