Digital Communications
ELC_3070
Modulation Project

Individual Report:

Name: محمد حسام عثمان يسن

Sec.: 3

BN: 37

- Single Carrier System

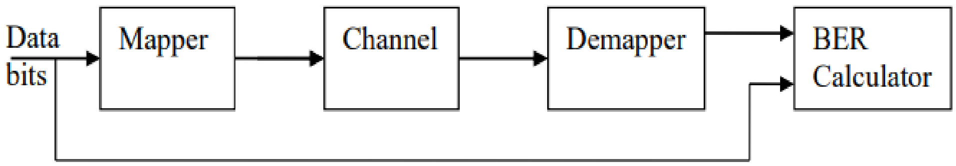
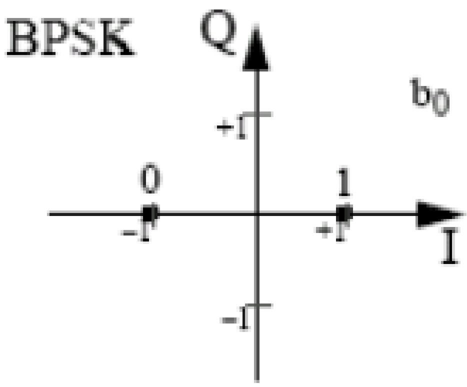


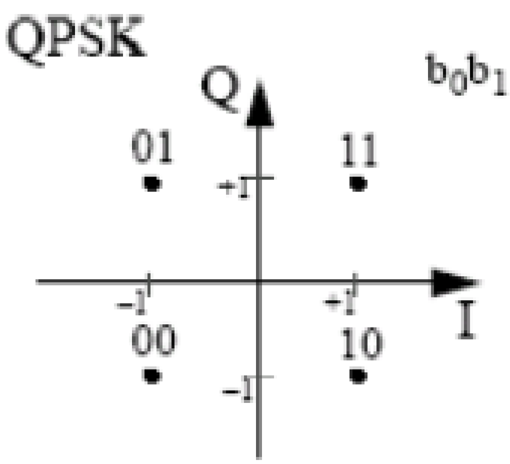
Figure 1 Single carrier communication system.

In this project, modulation has been done by the 4 modulation schemes under consideration which are the BPSK, QPSK, 8PSK, and 16QAM systems :

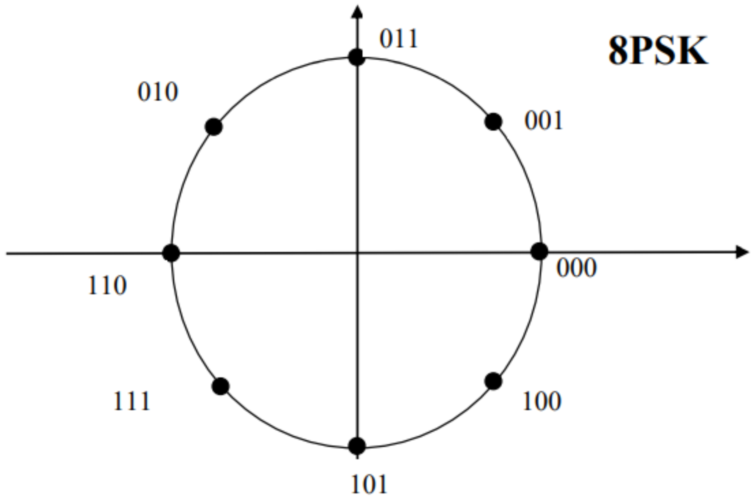
1. BPSK is done by modulating each single bit (0 or 1) into a symbol of 2 (1 or -1) as follows :



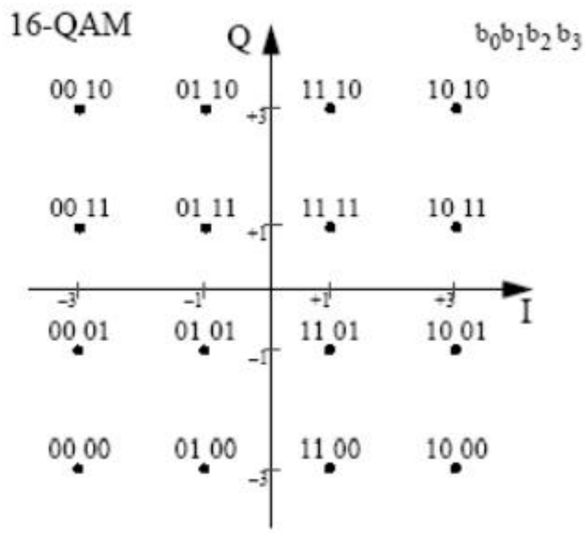
2. QPSK is done by modulating each two bits into a symbol of 4 as follows :



3. 8PSK is done by modulating each three bits into a symbol of 8 as follows :



4. 16QAM is done by modulating each four bits into a symbol of 16 as follows :



-Matlab Code:

```
N=120000 ;
Eb=1:30 ;
data_bits=randi([0 1],N,1) ;
s_bpsk=zeros(N,30) ;
s_qpsk=zeros(N/2,30) ;
s_8psk=zeros(N/3,30) ;
s_qam=zeros(N/4,30) ;
%%%%%%%%% Bpsk %%%%%%%%
for i=1:N
if data_bits(i,1)==0
    s_bpsk(i,:)=-sqrt(Eb) ;
else
    s_bpsk(i,:)=sqrt(Eb) ;
end
end
%%%%%%%%% Qpsk %%%%%%%%
for i=1:2:N
if data_bits(i,1)==0 && data_bits(i+1,1)==0
    s_qpsk((i+1)/2,:)=(-1-j)*sqrt(Eb) ;
elseif data_bits(i,1)==0 && data_bits(i+1,1)==1
```

```
    s_qpsk((i+1)/2,:)=(-1+j)*sqrt(Eb) ;
elseif data_bits(i,1)==1 && data_bits(i+1,1)==1
    s_qpsk((i+1)/2,:)=(1+j)*sqrt(Eb) ;
else
    s_qpsk((i+1)/2,:)=(1-j)*sqrt(Eb) ;
end
end
%%%%%%%%% 8psk %%%%%%%%
for i=1:3:N
if data_bits(i,1)==0 && data_bits(i+1,1)==0 && data_bits(i+2,1)==0
    s_8psk((i+2)/3,:)=sqrt(3*Eb) ;
elseif data_bits(i,1)==0 && data_bits(i+1,1)==0 && data_bits(i+2,1)==1
    s_8psk((i+2)/3,:)=sqrt(3*Eb)*exp(j*pi/4) ;
elseif data_bits(i,1)==0 && data_bits(i+1,1)==1 && data_bits(i+2,1)==1
    s_8psk((i+2)/3,:)=j*sqrt(3*Eb) ;
elseif data_bits(i,1)==0 && data_bits(i+1,1)==1 && data_bits(i+2,1)==0
    s_8psk((i+2)/3,:)=sqrt(3*Eb)*exp(j*3*pi/4) ;
elseif data_bits(i,1)==1 && data_bits(i+1,1)==1 && data_bits(i+2,1)==0
    s_8psk((i+2)/3,:)=-sqrt(3*Eb) ;
elseif data_bits(i,1)==1 && data_bits(i+1,1)==1 && data_bits(i+2,1)==1
    s_8psk((i+2)/3,:)=sqrt(3*Eb)*exp(j*5*pi/4) ;
elseif data_bits(i,1)==1 && data_bits(i+1,1)==0 && data_bits(i+2,1)==1
    s_8psk((i+2)/3,:)=-j*sqrt(3*Eb) ;
elseif data_bits(i,1)==1 && data_bits(i+1,1)==0 && data_bits(i+2,1)==0
    s_8psk((i+2)/3,:)=sqrt(3*Eb)*exp(j*-pi/4) ;
end
end
%%%%%%%%% QAM %%%%%%%%
for i=1:4:N
if (data_bits(i,1)==0 && data_bits(i+1,1)==0 && data_bits(i+2,1)==0 ...
    && data_bits(i+3,1)==0)
    s_qam((i+3)/4,:)=(-3-3j)*sqrt(Eb/2.5) ;
elseif (data_bits(i,1)==0 && data_bits(i+1,1)==0 && data_bits(i+2,1)==0 ...
    && data_bits(i+3,1)==1)
    s_qam((i+3)/4,:)=(-3-j)*sqrt(Eb/2.5) ;
elseif (data_bits(i,1)==0 && data_bits(i+1,1)==0 && data_bits(i+2,1)==1 ...
    && data_bits(i+3,1)==0)
    s_qam((i+3)/4,:)=(-3+3j)*sqrt(Eb/2.5) ;
elseif (data_bits(i,1)==0 && data_bits(i+1,1)==0 && data_bits(i+2,1)==1 ...
    && data_bits(i+3,1)==1)
    s_qam((i+3)/4,:)=(-3+j)*sqrt(Eb/2.5) ;
elseif (data_bits(i,1)==0 && data_bits(i+1,1)==1 && data_bits(i+2,1)==0 ...
    && data_bits(i+3,1)==0)
    s_qam((i+3)/4,:)=(-1-3j)*sqrt(Eb/2.5) ;
elseif (data_bits(i,1)==0 && data_bits(i+1,1)==1 && data_bits(i+2,1)==0 ...
    && data_bits(i+3,1)==1)
    s_qam((i+3)/4,:)=(-1-1j)*sqrt(Eb/2.5) ;
elseif (data_bits(i,1)==0 && data_bits(i+1,1)==1 && data_bits(i+2,1)==1 ...
    && data_bits(i+3,1)==0)
    s_qam((i+3)/4,:)=(-1+3j)*sqrt(Eb/2.5) ;
elseif (data_bits(i,1)==0 && data_bits(i+1,1)==1 && data_bits(i+2,1)==1 ...
    && data_bits(i+3,1)==1)
    s_qam((i+3)/4,:)=(-1+j)*sqrt(Eb/2.5) ;
elseif (data_bits(i,1)==1 && data_bits(i+1,1)==0 && data_bits(i+2,1)==0 ...
    && data_bits(i+3,1)==0)
    s_qam((i+3)/4,:)=(3-3j)*sqrt(Eb/2.5) ;
elseif (data_bits(i,1)==1 && data_bits(i+1,1)==0 && data_bits(i+2,1)==0 ...
    && data_bits(i+3,1)==1)
    s_qam((i+3)/4,:)=(3-j)*sqrt(Eb/2.5) ;
elseif (data_bits(i,1)==1 && data_bits(i+1,1)==0 && data_bits(i+2,1)==1 ...
    && data_bits(i+3,1)==0)
    s_qam((i+3)/4,:)=(3+3j)*sqrt(Eb/2.5) ;
elseif (data_bits(i,1)==1 && data_bits(i+1,1)==0 && data_bits(i+2,1)==1 ...
```

```
    && data_bits(i+3,1)==1)
    s_qam((i+3)/4,:)=(3+j)*sqrt(Eb/2.5) ;
elseif (data_bits(i,1)==1 && data_bits(i+1,1)==1 && data_bits(i+2,1)==0 ...
    && data_bits(i+3,1)==0)
    s_qam((i+3)/4,:)=(1-3j)*sqrt(Eb/2.5) ;
elseif (data_bits(i,1)==1 && data_bits(i+1,1)==1 && data_bits(i+2,1)==0 ...
    && data_bits(i+3,1)==1)
    s_qam((i+3)/4,:)=(1-j)*sqrt(Eb/2.5) ;
elseif (data_bits(i,1)==1 && data_bits(i+1,1)==1 && data_bits(i+2,1)==1 ...
    && data_bits(i+3,1)==0)
    s_qam((i+3)/4,:)=(1+3j)*sqrt(Eb/2.5) ;
elseif (data_bits(i,1)==1 && data_bits(i+1,1)==1 && data_bits(i+2,1)==1 ...
    && data_bits(i+3,1)==1)
    s_qam((i+3)/4,:)=(1+j)*sqrt(Eb/2.5) ;
end
end
%%%%% Signal after channel %%%%%
No=2 ;
noise_bpsk=(sqrt(No/2))*randn(N,1)+j*((sqrt(No/2))*randn(N,1)) ;
noise_qpsk=(sqrt(No/2))*randn(N/2,1)+j*((sqrt(No/2))*randn(N/2,1)) ;
noise_8psk=(sqrt(No/2))*randn(N/3,1)+j*((sqrt(No/2))*randn(N/3,1)) ;
noise_qam=(sqrt(No/2))*randn(N/4,1)+j*((sqrt(No/2))*randn(N/4,1)) ;
noise_bpsk=repmat(noise_bpsk,1,30) ;
noise_qpsk=repmat(noise_qpsk,1,30) ;
noise_8psk=repmat(noise_8psk,1,30) ;
noise_qam=repmat(noise_qam,1,30) ;
X_bpsk=s_bpsk+noise_bpsk ;
X_qpsk=s_qpsk+noise_qpsk ;
X_8psk=s_8psk+noise_8psk ;
X_qam=s_qam+noise_qam ;
%%%%% Signal after demapper %%%%%
rec_bits_bpsk=zeros(N,30) ;
rec_bits_qpsk=zeros(N,30) ;
rec_bits_8psk=zeros(N,30) ;
rec_bits_qam=zeros(N,30) ;
%%%%%%%%% Bpsk %%%%%%%%
for i=1:N
    for k=1:30
if X_bpsk(i,k)>0
    rec_bits_bpsk(i,k)=1 ;
else
    rec_bits_bpsk(i,k)=0 ;
end
    end
end
%%%%%%%%% Qpsk %%%%%%%%
for i=1:N/2
    for k=1:30
if real(X_qpsk(i,k))>0 && imag(X_qpsk(i,k))>0
    rec_bits_qpsk(2*i-1,k)=1 ; rec_bits_qpsk(2*i,k)=1 ;
elseif real(X_qpsk(i,k))<0 && imag(X_qpsk(i,k))>0
    rec_bits_qpsk(2*i-1,k)=0 ; rec_bits_qpsk(2*i,k)=1 ;
elseif real(X_qpsk(i,k))<0 && imag(X_qpsk(i,k))<0
    rec_bits_qpsk(2*i-1,k)=0 ; rec_bits_qpsk(2*i,k)=0 ;
elseif real(X_qpsk(i,k))>0 && imag(X_qpsk(i,k))<0
    rec_bits_qpsk(2*i-1,k)=1 ; rec_bits_qpsk(2*i,k)=0 ;
end
   end
end
%%%%%%%%% 8psk %%%%%%%%
for i=1:N/3
    for k=1:30
if angle(X_8psk(i,k))>(-pi/8) && angle(X_8psk(i,k))<(pi/8)
```

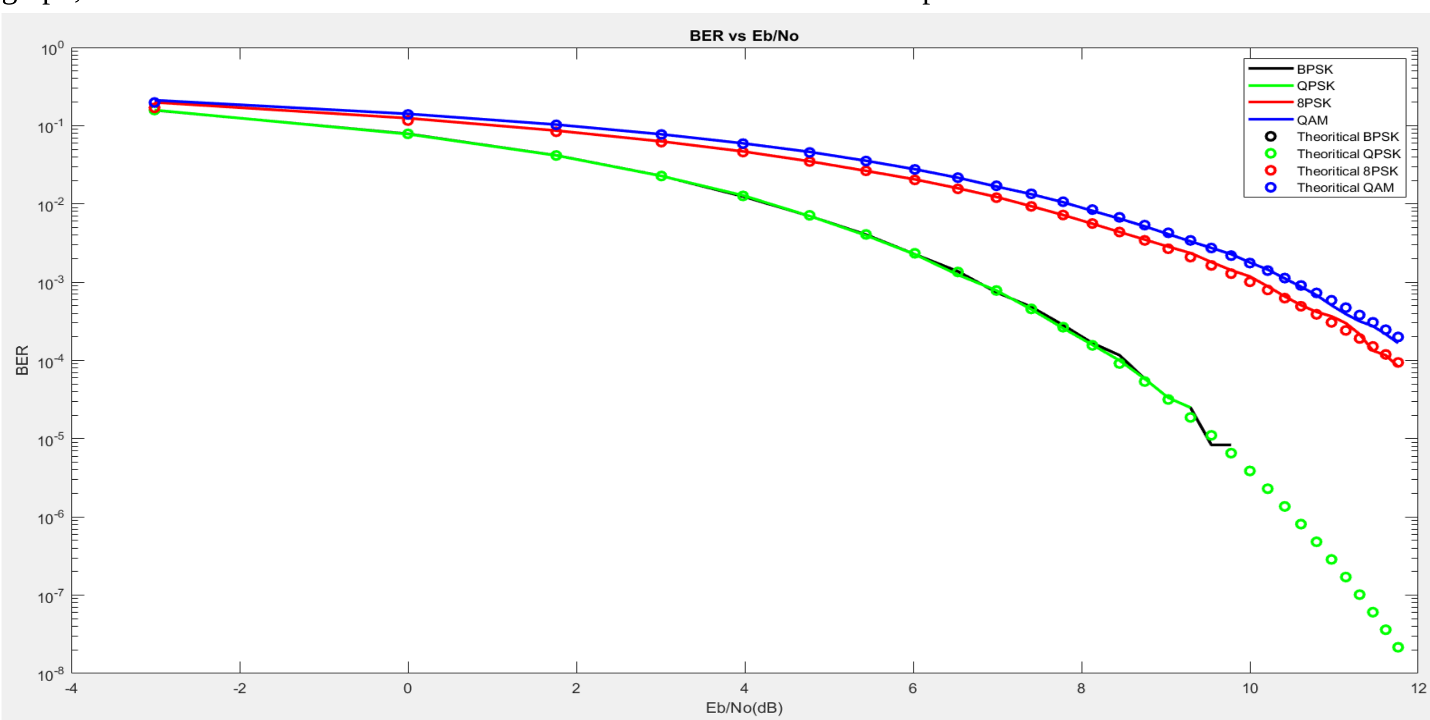
```
rec_bits_8psk(3*i-2,k)=0 ;rec_bits_8psk(3*i-1,k)=0 ;rec_bits_8psk(3*i,k)=0;
elseif angle(X_8psk(i,k))>(pi/8) && angle(X_8psk(i,k))<(3*pi/8)
rec_bits_8psk(3*i-2,k)=0 ;rec_bits_8psk(3*i-1,k)=0 ;rec_bits_8psk(3*i,k)=1;
elseif angle(X_8psk(i,k))>(3*pi/8) && angle(X_8psk(i,k))<(5*pi/8)
rec_bits_8psk(3*i-2,k)=0 ;rec_bits_8psk(3*i-1,k)=1 ;rec_bits_8psk(3*i,k)=1;
elseif angle(X_8psk(i,k))>(5*pi/8) && angle(X_8psk(i,k))<(7*pi/8)
rec_bits_8psk(3*i-2,k)=0 ;rec_bits_8psk(3*i-1,k)=1 ;rec_bits_8psk(3*i,k)=0;
elseif (angle(X_8psk(i,k))>(7*pi/8) && angle(X_8psk(i,k))<(pi)) || ...
        (angle(X_8psk(i,k))<(-7*pi/8) && angle(X_8psk(i,k))>(-pi))
rec_bits_8psk(3*i-2,k)=1 ;rec_bits_8psk(3*i-1,k)=1 ;rec_bits_8psk(3*i,k)=0;
elseif angle(X_8psk(i,k))>(-7*pi/8) && angle(X_8psk(i,k))<(-5*pi/8)
rec_bits_8psk(3*i-2,k)=1 ;rec_bits_8psk(3*i-1,k)=1 ;rec_bits_8psk(3*i,k)=1;
elseif angle(X_8psk(i,k))>(-5*pi/8) && angle(X_8psk(i,k))<(-3*pi/8)
rec_bits_8psk(3*i-2,k)=1 ;rec_bits_8psk(3*i-1,k)=0 ;rec_bits_8psk(3*i,k)=1;
elseif angle(X_8psk(i,k))>(-3*pi/8) && angle(X_8psk(i,k))<(-pi/8)
rec_bits_8psk(3*i-2,k)=1 ;rec_bits_8psk(3*i-1,k)=0 ;rec_bits_8psk(3*i,k)=0;
end
    end
end
%%%%%%%%% QAM %%%%%%%%
for i=1:N/4
    for k=1:30
if (real(X_qam(i,k))>0 && real(X_qam(i,k)<2*sqrt(Eb(1,k)/2.5)) && ...
 imag(X_qam(i,k))>0 && imag(X_qam(i,k))<2*sqrt(Eb(1,k)/2.5))
rec_bits_qam(4*i-3,k)=1;rec_bits_qam(4*i-2,k)=1;rec_bits_qam(4*i-1,k)=1;...
    rec_bits_qam(4*i,k)=1 ;
elseif (real(X_qam(i,k))>0 && real(X_qam(i,k))<2*sqrt(Eb(1,k)/2.5) && ...
        imag(X_qam(i,k))>2*sqrt(Eb(1,k)/2.5))
rec_bits_qam(4*i-3,k)=1;rec_bits_qam(4*i-2,k)=1;rec_bits_qam(4*i-1,k)=1;...
    rec_bits_qam(4*i,k)=0 ;
elseif (real(X_qam(i,k))>0 && real(X_qam(i,k))<2*sqrt(Eb(1,k)/2.5) && ...
    imag(X_qam(i,k))<0 && imag(X_qam(i,k))>-2*sqrt(Eb(1,k)/2.5))
rec_bits_qam(4*i-3,k)=1;rec_bits_qam(4*i-2,k)=1;rec_bits_qam(4*i-1,k)=0;...
    rec_bits_qam(4*i,k)=1 ;
elseif (real(X_qam(i,k))>0 && real(X_qam(i,k))<2*sqrt(Eb(1,k)/2.5) && ...
    imag(X_qam(i,k))<-2*sqrt(Eb(1,k)/2.5))
rec_bits_qam(4*i-3,k)=1;rec_bits_qam(4*i-2,k)=1;rec_bits_qam(4*i-1,k)=0;...
    rec_bits_qam(4*i,k)=0 ;
elseif (real(X_qam(i,k))>2*sqrt(Eb(1,k)/2.5) &&imag(X_qam(i,k))>0 && ...
    imag(X_qam(i,k))<2*sqrt(Eb(1,k)/2.5))
rec_bits_qam(4*i-3,k)=1;rec_bits_qam(4*i-2,k)=0;rec_bits_qam(4*i-1,k)=1;...
    rec_bits_qam(4*i,k)=1 ;
elseif (real(X_qam(i,k))>2*sqrt(Eb(1,k)/2.5) && ...
    imag(X_qam(i,k))>2*sqrt(Eb(1,k)/2.5))
rec_bits_qam(4*i-3,k)=1;rec_bits_qam(4*i-2,k)=0;rec_bits_qam(4*i-1,k)=1;...
    rec_bits_qam(4*i,k)=0 ;
elseif (real(X_qam(i,k))>2*sqrt(Eb(1,k)/2.5) &&imag(X_qam(i,k))<0 && ...
    imag(X_qam(i,k))>-2*sqrt(Eb(1,k)/2.5))
rec_bits_qam(4*i-3,k)=1;rec_bits_qam(4*i-2,k)=0;rec_bits_qam(4*i-1,k)=0;...
    rec_bits_qam(4*i,k)=1 ;
elseif (real(X_qam(i,k))>2*sqrt(Eb(1,k)/2.5) && ...
    imag(X_qam(i,k))<-2*sqrt(Eb(1,k)/2.5))
rec_bits_qam(4*i-3,k)=1;rec_bits_qam(4*i-2,k)=0;rec_bits_qam(4*i-1,k)=0;...
    rec_bits_qam(4*i,k)=0 ;
elseif (real(X_qam(i,k))<0 && real(X_qam(i,k))>-2*sqrt(Eb(1,k)/2.5) && ...
    imag(X_qam(i,k))>0&& imag(X_qam(i,k))<2*sqrt(Eb(1,k)/2.5))
rec_bits_qam(4*i-3,k)=0;rec_bits_qam(4*i-2,k)=1;rec_bits_qam(4*i-1,k)=1;...
    rec_bits_qam(4*i,k)=1 ;
elseif (real(X_qam(i,k))<0 && real(X_qam(i,k))>-2*sqrt(Eb(1,k)/2.5) && ...
    imag(X_qam(i,k))>2*sqrt(Eb(1,k)/2.5))
rec_bits_qam(4*i-3,k)=0;rec_bits_qam(4*i-2,k)=1;rec_bits_qam(4*i-1,k)=1;...
    rec_bits_qam(4*i,k)=0 ;
elseif (real(X_qam(i,k))<0 && real(X_qam(i,k))>-2*sqrt(Eb(1,k)/2.5) && ...
```

```
    imag(X_qam(i,k))<0&& imag(X_qam(i,k))>-2*sqrt(Eb(1,k)/2.5))
rec_bits_qam(4*i-3,k)=0;rec_bits_qam(4*i-2,k)=1;rec_bits_qam(4*i-1,k)=0;...
    rec_bits_qam(4*i,k)=1 ;
elseif (real(X_qam(i,k))<0 && real(X_qam(i,k))>-2*sqrt(Eb(1,k)/2.5) && ...
    imag(X_qam(i,k))<-2*sqrt(Eb(1,k)/2.5))
rec_bits_qam(4*i-3,k)=0;rec_bits_qam(4*i-2,k)=1;rec_bits_qam(4*i-1,k)=0;...
    rec_bits_qam(4*i,k)=0 ;
elseif (real(X_qam(i,k))<-2*sqrt(Eb(1,k)/2.5) && imag(X_qam(i,k))>0 && ...
    imag(X_qam(i,k))<2*sqrt(Eb(1,k)/2.5))
rec_bits_qam(4*i-3,k)=0;rec_bits_qam(4*i-2,k)=0;rec_bits_qam(4*i-1,k)=1;...
    rec_bits_qam(4*i,k)=1 ;
elseif (real(X_qam(i,k))<-2*sqrt(Eb(1,k)/2.5) && ...
    imag(X_qam(i,k))>2*sqrt(Eb(1,k)/2.5))
rec_bits_qam(4*i-3,k)=0;rec_bits_qam(4*i-2,k)=0;rec_bits_qam(4*i-1,k)=1;...
    rec_bits_qam(4*i,k)=0 ;
elseif (real(X_qam(i,k))<-2*sqrt(Eb(1,k)/2.5) && imag(X_qam(i,k))<0 && ...
    imag(X_qam(i,k))>-2*sqrt(Eb(1,k)/2.5))
rec_bits_qam(4*i-3,k)=0;rec_bits_qam(4*i-2,k)=0;rec_bits_qam(4*i-1,k)=0;...
    rec_bits_qam(4*i,k)=1 ;
elseif (real(X_qam(i,k))<-2*sqrt(Eb(1,k)/2.5)&& ...
    imag(X_qam(i,k))<-2*sqrt(Eb(1,k)/2.5))
rec_bits_qam(4*i-3,k)=0;rec_bits_qam(4*i-2,k)=0;rec_bits_qam(4*i-1,k)=0;...
    rec_bits_qam(4*i,k)=0 ;
end
    end
end
%%%%% BER %%%%%%
errors_bpsk=zeros(1,30) ;
errors_qpsk=zeros(1,30) ;
errors_8psk=zeros(1,30) ;
errors_qam=zeros(1,30) ;
%%%%% Bpsk %%%%%
for i=1:N
    for k=1:30
    if rec_bits_bpsk(i,k)~=data_bits(i,1)
       errors_bpsk(1,k)=errors_bpsk(1,k)+1 ;
    end
    end
end
BER_bpsk=errors_bpsk/N ; BER_bpsk=BER_bpsk.' ;
%%%%% Qpsk %%%%%
for i=1:N
    for k=1:30
    if rec_bits_qpsk(i,k)~=data_bits(i,1)
       errors_qpsk(1,k)=errors_qpsk(1,k)+1 ;
    end
    end
end
BER_qpsk=errors_qpsk/N ; BER_qpsk=BER_qpsk.' ;
%%%%% 8psk %%%%%
for i=1:N
    for k=1:30
    if rec_bits_8psk(i,k)~=data_bits(i,1)
       errors_8psk(1,k)=errors_8psk(1,k)+1 ;
    end
    end
end
BER_8psk=errors_8psk/N ; BER_8psk=BER_8psk.' ;
%%%%% QAM %%%%%
for i=1:N
    for k=1:30
    if rec_bits_qam(i,k)~=data_bits(i,1)
       errors_qam(1,k)=errors_qam(1,k)+1 ;
```

```
    end
    end
end
BER_qam=errors_qam/N ; BER_qam=BER_qam.' ;
%%%%% Theoritical BER %%%%%%
BER_theo_bpsk=0.5*erfc(sqrt(Eb/No)) ; BER_theo_bpsk=BER_theo_bpsk.' ;
BER_theo_qpsk=0.5*erfc(sqrt(Eb/No)) ;BER_theo_qpsk=BER_theo_qpsk.' ;
BER_theo_8psk=erfc((sqrt(3*Eb/No))*sin(pi/8))/3 ;
BER_theo_8psk=BER_theo_8psk.' ;
BER_theo_qam=1.5*erfc(sqrt(Eb/(2.5*No)))/4 ; BER_theo_qam=BER_theo_qam.' ;
%%%%% Plotting %%%%%
Eb_No=10*log10(Eb/No); Eb_No=(Eb_No).' ;
slg=semilogy(Eb_No,BER_bpsk,'k',Eb_No,BER_qpsk,'g',Eb_No,BER_8psk,'r',...
Eb_No,BER_qam,'b') ;
slg(1).LineWidth=2;slg(2).LineWidth=2;slg(3).LineWidth=2;
slg(4).LineWidth=2;
hold on
slg2=semilogy(Eb_No,BER_theo_bpsk,'ok',...
Eb_No,BER_theo_qpsk,'og',...
Eb_No,BER_theo_8psk,'or',...
Eb_No,BER_theo_qam,'ob') ;
slg2(1).LineWidth=2;slg2(2).LineWidth=2;slg2(3).LineWidth=2;
slg2(4).LineWidth=2;
xlabel('Eb/No(dB)');
ylabel('BER') ;
legend('BPSK','QPSK','8PSK','QAM','Theoritical BPSK','Theoritical QPSK',...
    'Theoritical 8PSK','Theoritical QAM') ;
title('BER vs Eb/No') ;
hold off
```

-Plotting curves:

In the following figure, the BER for the four modulation schemes is plotted Vs Eb/No (SNR). On the same graph, the theoretical BER for each one of the 4 modulation schemes is plotted in circular markers.



-Comments:

From the previous figure, we see that the BER decreases as SNR increases (or as bit energy increases) and this happens as when the bit energy increases, the signal becomes more robust to noise, thus adding noise to the signal through the channel does not have much effects in retrieving the transmitted bits. We also observe from the previous figure that the BER in the case of BPSK and QPSK is much better than in 8PSK and 16QAM as in BPSK and QPSK the number of bits pers symbol is less than those in 8PSK and 16QAM. The BER of BPSK and QPSK is almost the same as we see in the previous figure the coincidence between their curves. The BER in 8PSK is better than that of the 16QAM as the number of bits pers symbol in 8PSK is less than those in 16QAM. Finally, the theoretical BER of each modulation scheme is almost the same as its practical BER and we can see this in the previous figure (any mismatch between practical and theoretical BER is due to finite number of bits).