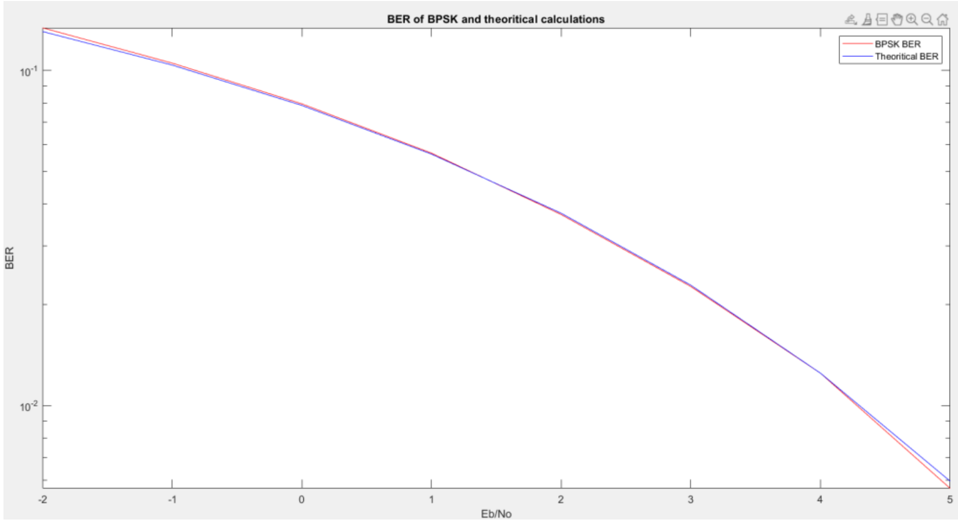
Digital Communication

Project 3

ID	Sec	Bn	Name
9203219	3	29	محمد حمدي عبد الحميد
9203753	4	49	يوسف ايمن رزق ابراهيم

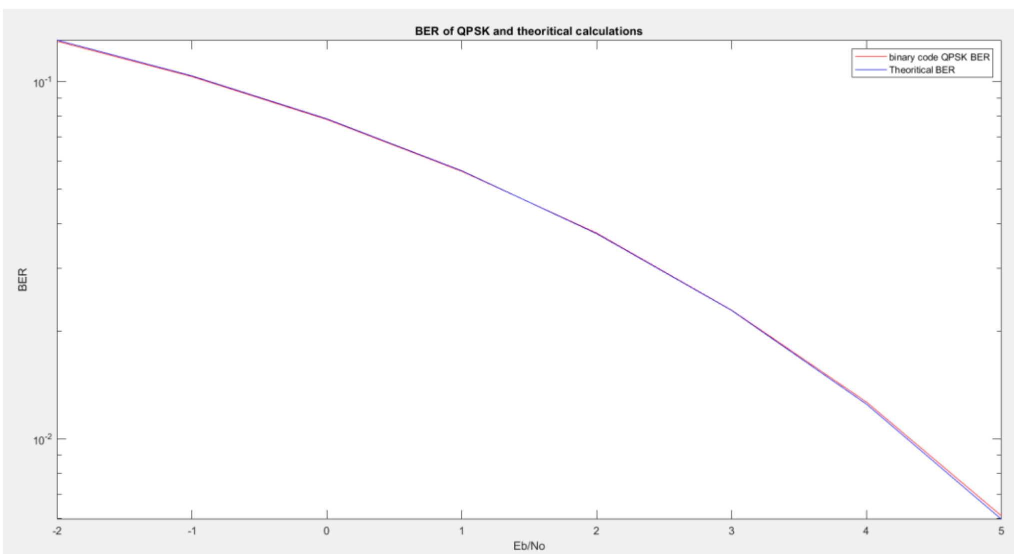
I. BPSK



Comment on the results:

- Energy per symbol $\rightarrow E_s = \frac{1^2+1^2}{2} = 1$
- Energy per bit $\rightarrow E_b = \frac{E_s}{\#bits} = 1$
- Theoretical BER $\rightarrow BER = 0.5erfc\left(\sqrt{\frac{E_b}{N_o}}\right)$

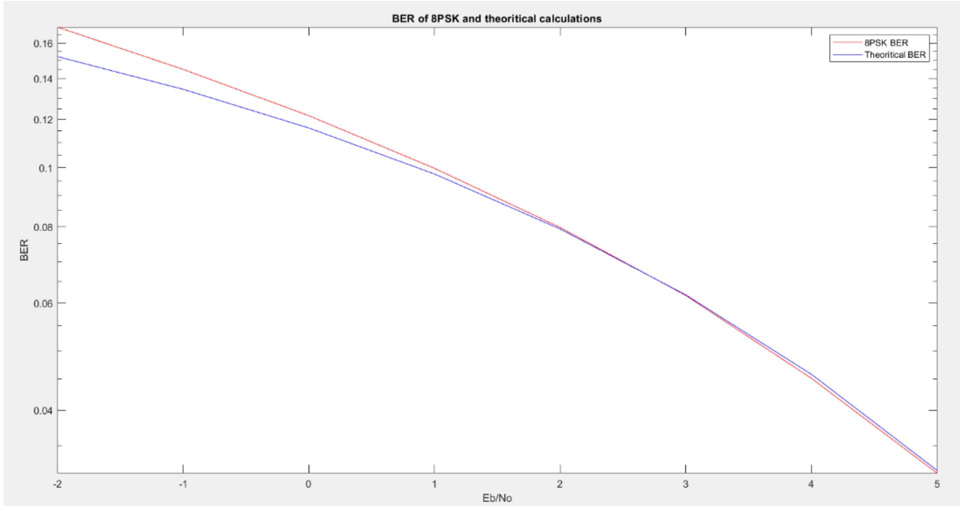
II. QPSK



Comment on the results:

- Energy per symbol $\rightarrow E_s = \frac{(\sqrt{2})^2 * 4}{4} = 2$
- Energy per bit $\rightarrow E_b = \frac{E_s}{\#bits} = \frac{2}{2} = 1$
- Theoretical BER $\rightarrow BER = 0.5erfc\left(\sqrt{\frac{E_b}{N_o}}\right)$

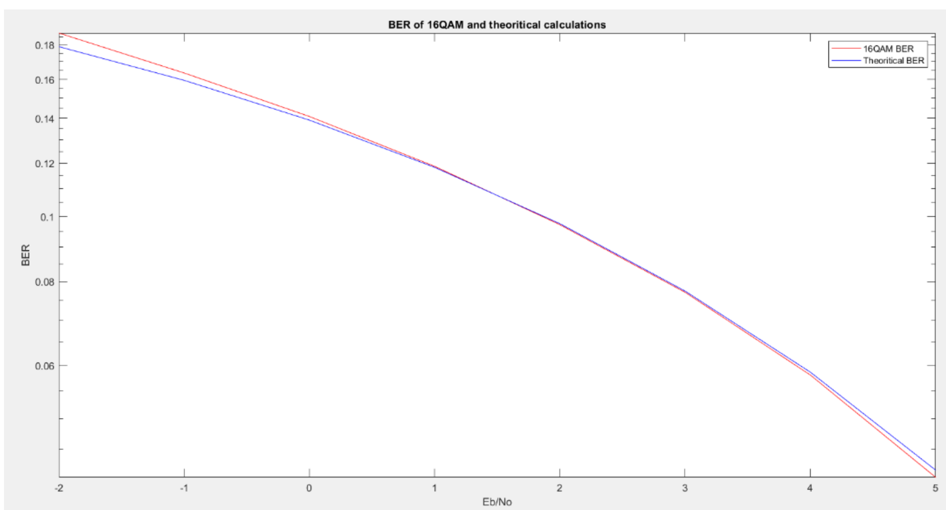
III. 8PSK



Comment on the results:

- Energy per symbol $\rightarrow E_s = \frac{1^2 * 8}{8} = 1$
- Energy per bit $\rightarrow E_b = \frac{E_s}{\#bits} = \frac{1}{3}$
- Theoretical BER $\rightarrow BER = \frac{1}{3} erfc\left(\sqrt{\frac{3E_b}{N_o}} \sin\left(\frac{\pi}{8}\right)\right)$
- There is a little difference between the theoretical BER and the practical BER as we have used a tight upper bound approximation to calculate the theoretical BER.

IV. 16 QAM

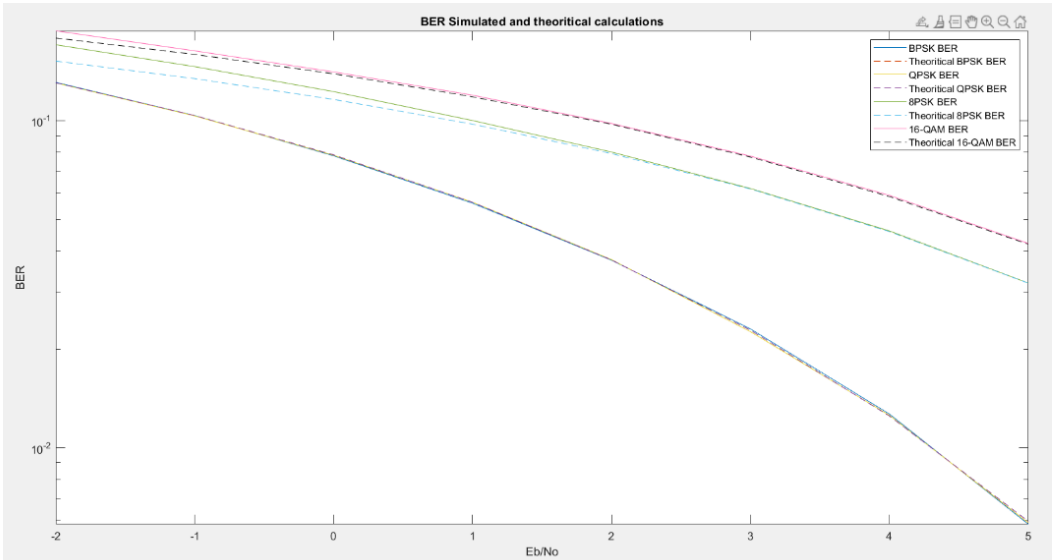


Comment on the results:

- Energy per symbol $\rightarrow E_s = \frac{4*(\sqrt{2})^2 + 4*(\sqrt{18})^2 + 8*(\sqrt{10})^2}{16} = 10$
- Energy per bit $\rightarrow E_b = \frac{E_s}{4} = 2.5$
- Theoretical BER $\rightarrow BER = \frac{3}{8} erfc\left(\sqrt{\frac{E_b}{2.5N_o}}\right)$

V. Tasks

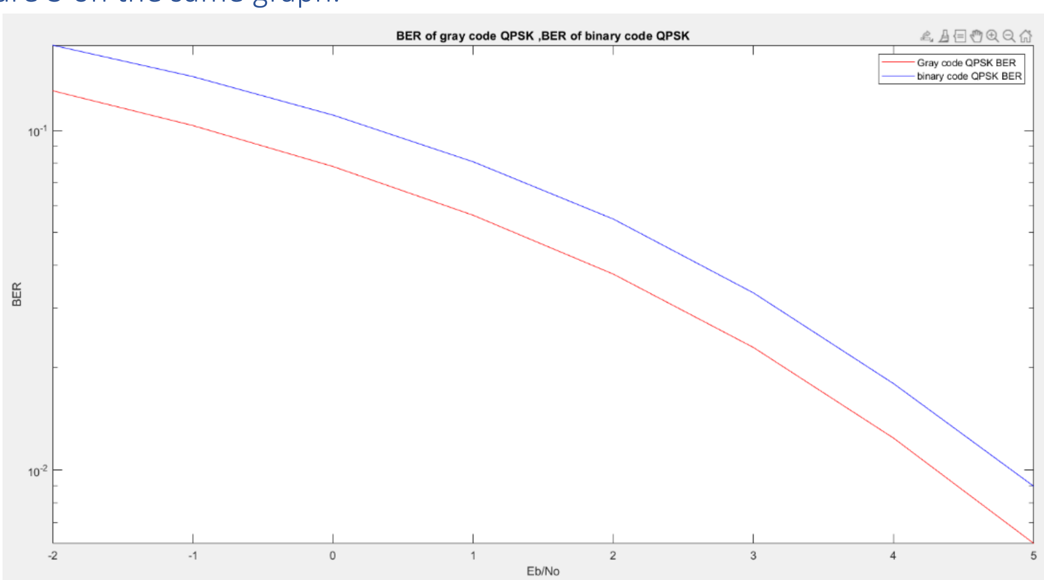
Plot curves for the BER vs. Eb/No for the four modulation schemes on the same graph. And plot the theoretical BER each one of the 4 modulation schemes on the same graph too using dashed lines.



Comment on the results:

- The BER increases when we increase the number of bits used for each symbol. So, the 16-QAM has the highest BER.
- BPSK and QPSK have approximately the same BER.

Plot the BER vs Eb/No for the QPSK case shown in Figure 2 and the case shown in Figure 3 on the same graph.



Comment on your findings:

- The BER of the binary code is higher than the grey code as in case of binary code the demapper can make mistake in 2 bits at the same time which doesn't exist in grey code as the difference between any 2 symbols is just one bit.

VI. BFSK

What are the basis functions of the signal set?

$$\emptyset_1 = \sqrt{\frac{2}{T_b}}\cos(2\pi f_1 t)$$

$$\emptyset_2 = \sqrt{\frac{2}{T_b}}\cos(2\pi f_2 t)$$

$$f_2 = f_1 + \frac{1}{T_b}$$

Write an expression for the baseband equivalent signals for this set, indicating the carrier frequency used.

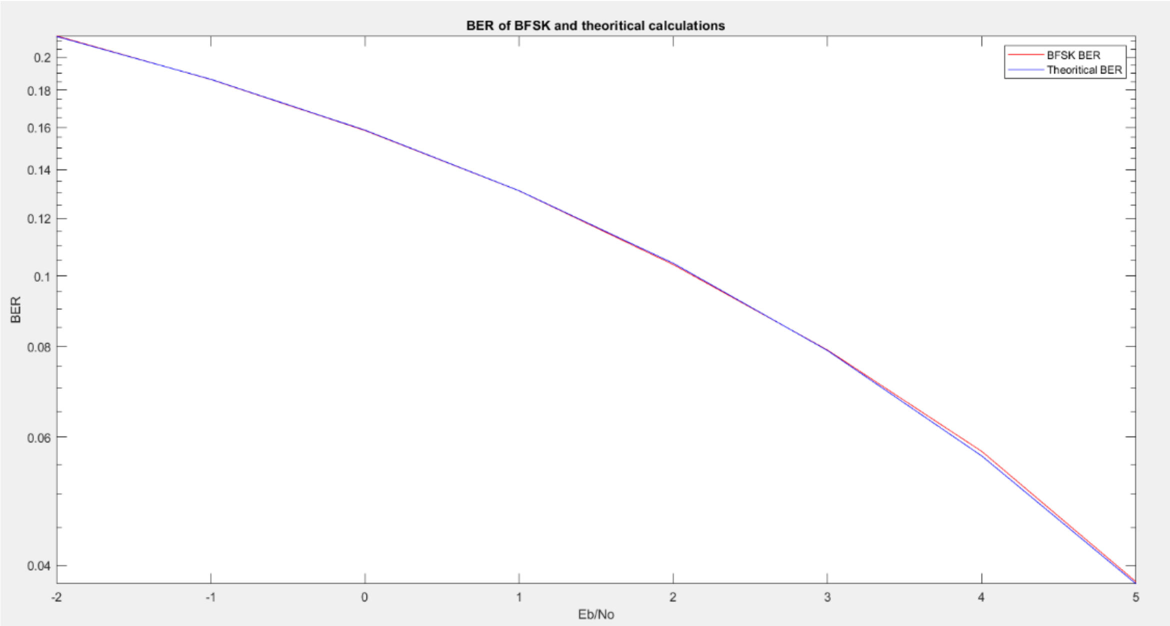
The carrier frequency → $f_c = f1$

$$S_{1BB} = \sqrt{\frac{2E_b}{T_b}}$$

$$S_{2BB} = \sqrt{\frac{2E_b}{T_b}}[\cos(2\pi\Delta f t) + \mathrm{j}\sin(2\pi\Delta f t)]$$

$$\Delta f = f_2 - f_1 = \frac{1}{T_b}$$

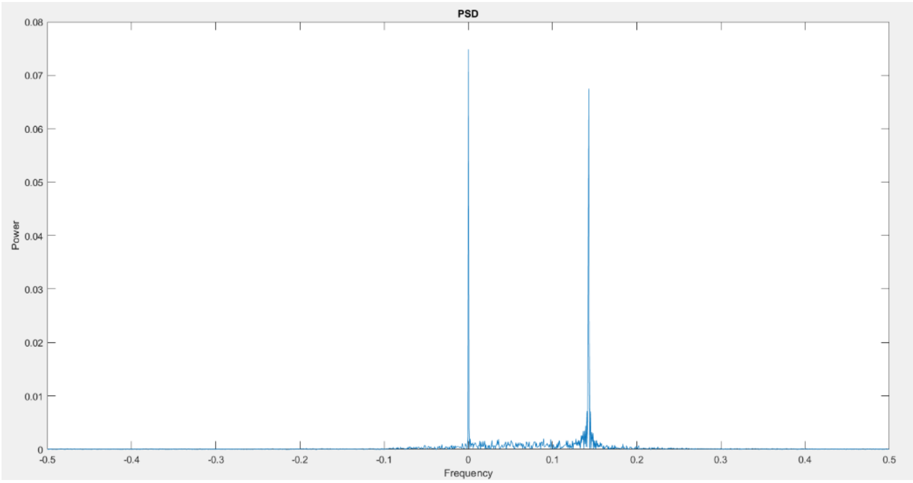
Use the baseband equivalent system to draw the BER curve vs. Eb/No. Also draw the theoretical BER curve on the same graph.



Comment on the results:

- Practical and theoretical BER are approximately the same.

Simulate the PSD of the signal set using the baseband equivalent signal.



Comment on the results:

- We have 2 deltas one at zero and the second at $\Delta f = \frac{1}{T_b} = \frac{1}{7}$.
- The position of the second delta depends on the number of samples and T_b.
- The amplitude of the deltas depends on E_b and T_b.

VII. Code

1. 8PSK, BPSK, QPSK, and 16-QAM

```
clc
clear all
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% BPSK %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
A = randi( [0,1] , 1 , 120000);
Es_BPSK = 1;
s1_BPSK = sqrt(Es_BPSK);
s2_BPSK = -sqrt(Es_BPSK);
mappeddata_BPSK = zeros(1,120000);
n_BPSK = randn(1,120000) + i*randn(1,120000);

for j = -2:5
    z = 10^(j/10);
    No = Es_BPSK/z;

%%%%%%%%%%%%%% Mapping %%%%%%%%%%%%%%%
for p = 1:120000
    if A(p) == 1
       mappeddata_BPSK(p) = s1_BPSK;
    else
        mappeddata_BPSK(p) = s2_BPSK;
    end
end

channeldata = mappeddata_BPSK + real(n_BPSK)*sqrt(No/2);
outputdata_BPSK = zeros(1,120000);
err = 0;
%%%%%%%%%%%%%% Demapping %%%%%%%%%%%%%%%
for p = 1:120000
    if channeldata(p) >= 0
        outputdata_BPSK(p) = 1;
    else
        outputdata_BPSK(p) = 0;
    end
    if outputdata_BPSK(p) ~= A(p)
        err = err + 1;
    end
end
%%%%%%%%%%%%%%%% BER %%%%%%%%%%%%%%%%%%%%%
BER_BPSK(j+3) = err/120000;

end
theoretical_BER_BPSK = 0.5 *erfc(sqrt(10.^((-2:5)./10)));

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% QPSK %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Es_QPSK = 2;
Eb_QPSK = Es_QPSK/2;
s00 = -sqrt(Es_QPSK/2)-i*sqrt(Es_QPSK/2);
s01 = -sqrt(Es_QPSK/2)+i*sqrt(Es_QPSK/2);
s10 = sqrt(Es_QPSK/2)-i*sqrt(Es_QPSK/2);
s11 = sqrt(Es_QPSK/2)+i*sqrt(Es_QPSK/2);

sn00 = -sqrt(Es_QPSK/2)-i*sqrt(Es_QPSK/2);
```

```
sn01 = -sqrt(Es_QPSK/2)+i*sqrt(Es_QPSK/2);
sn10 = sqrt(Es_QPSK/2)+i*sqrt(Es_QPSK/2);
sn11 = sqrt(Es_QPSK/2)-i*sqrt(Es_QPSK/2);

mappeddata_QPSK = zeros(1,60000);
mappeddata2_QPSK = zeros(1,60000);
n_QPSK = randn(1,60000) + i*randn(1,60000);

for j = -2:5
    z = 10^(j/10);
    No = Eb_QPSK/z;

k = 1;
%%%%%%%%%%%%%% Mapping %%%%%%%%%%%%%%
for p = 1:2:120000
    if A(p) == 0 && A(p+1) == 0
       mappeddata_QPSK(k) = s00;
       mappeddata2_QPSK(k) = sn00;
    elseif A(p) == 0 && A(p+1) == 1
       mappeddata_QPSK(k) = s01;
       mappeddata2_QPSK(k) = sn01;
    elseif A(p) == 1 && A(p+1) == 0
       mappeddata_QPSK(k) = s10;
       mappeddata2_QPSK(k) = sn10;
    elseif A(p) == 1 && A(p+1) == 1
       mappeddata_QPSK(k) = s11;
       mappeddata2_QPSK(k) = sn11;
    end
    k = k+1;
end

channeldata = mappeddata_QPSK + n_QPSK*sqrt(No/2);
channeldata2 = mappeddata2_QPSK + n_QPSK*sqrt(No/2);
outputdata_QPSK = zeros(1,120000);
outputdata2_QPSK = zeros(1,120000);
k = 1;

%%%%%%%%%%%%%% Demapping %%%%%%%%%%%%%%
for p = 1:2:120000
    if real(channeldata(k)) < 0 && imag(channeldata(k)) < 0
        outputdata_QPSK(p) = 0;
        outputdata_QPSK(p+1) = 0;
    elseif real(channeldata(k)) < 0 && imag(channeldata(k)) >= 0
        outputdata_QPSK(p) = 0;
        outputdata_QPSK(p+1) = 1;
    elseif real(channeldata(k)) >= 0 && imag(channeldata(k)) < 0
        outputdata_QPSK(p) = 1;
        outputdata_QPSK(p+1) = 0;
    elseif real(channeldata(k)) >= 0 && imag(channeldata(k)) >= 0
        outputdata_QPSK(p) = 1;
        outputdata_QPSK(p+1) = 1;
    end
```

```
if real(channeldata2(k)) < 0 && imag(channeldata2(k)) < 0
        outputdata2_QPSK(p) = 0;
        outputdata2_QPSK(p+1) = 0;
    elseif real(channeldata2(k)) < 0 && imag(channeldata2(k)) >= 0
        outputdata2_QPSK(p) = 0;

        outputdata2_QPSK(p+1) = 1;
    elseif real(channeldata2(k)) >= 0 && imag(channeldata2(k)) < 0
        outputdata2_QPSK(p) = 1;
        outputdata2_QPSK(p+1) = 1;
    elseif real(channeldata2(k)) >= 0 && imag(channeldata2(k)) >= 0
        outputdata2_QPSK(p) = 1;
        outputdata2_QPSK(p+1) = 0;
    end
    k = k+1;
end
err = 0;
err2 = 0;
for p = 1:120000
    if outputdata_QPSK(p) ~= A(p)
        err = err + 1;
    end
    if outputdata2_QPSK(p) ~= A(p)
        err2 = err2 + 1;
    end
end
%%%%%%%%%%%%%%%% BER %%%%%%%%%%%%%%%%%%%%%

BER_QPSK(j+3) = err/120000;
BER2_QPSK(j+3) = err2/120000;

end
theoretical_BER_QPSK = 0.5*erfc(sqrt(10.^((-2:5)./10)));

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% 8PSK %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Es_8psk = 1;
Eb_8psk = Es_8psk/3;
s000_8psk = sqrt(Es_8psk);
s001_8psk = (1/sqrt(2))*sqrt(Es_8psk)+i*(1/sqrt(2))*sqrt(Es_8psk);
s011_8psk = i*sqrt(Es_8psk);
s010_8psk = -(1/sqrt(2))*sqrt(Es_8psk)+i*(1/sqrt(2))*sqrt(Es_8psk);
s110_8psk = -sqrt(Es_8psk);
s111_8psk = -(1/sqrt(2))*sqrt(Es_8psk)-i*(1/sqrt(2))*sqrt(Es_8psk);
s101_8psk = -i*sqrt(Es_8psk);
s100_8psk = (1/sqrt(2))*sqrt(Es_8psk)-i*(1/sqrt(2))*sqrt(Es_8psk);
mappeddata_8psk = zeros(1,40000);
n_8psk = randn(1,40000) + i*randn(1,40000);

for j = -2:5
    z = 10^(j/10);
    No = Eb_8psk/z;

k = 1;
%%%%%%%%%%%%%%%% Mapping %%%%%%%%%%%%%%%%%%%%%
```

```
for p = 1:3:120000
    if A(p) == 0 && A(p+1) == 0 && A(p+2) == 0
       mappeddata_8psk(k) = s000_8psk;
    elseif A(p) == 0 && A(p+1) == 0 && A(p+2) == 1
       mappeddata_8psk(k) = s001_8psk;
    elseif A(p) == 0 && A(p+1) == 1 && A(p+2) == 1
       mappeddata_8psk(k) = s011_8psk;

    elseif A(p) == 0 && A(p+1) == 1 && A(p+2) == 0
       mappeddata_8psk(k) = s010_8psk;
    elseif A(p) == 1 && A(p+1) == 1 && A(p+2) == 0
       mappeddata_8psk(k) = s110_8psk;
    elseif A(p) == 1 && A(p+1) == 1 && A(p+2) == 1
       mappeddata_8psk(k) = s111_8psk;
    elseif A(p) == 1 && A(p+1) == 0 && A(p+2) == 1
       mappeddata_8psk(k) = s101_8psk;
    elseif A(p) == 1 && A(p+1) == 0 && A(p+2) == 0
       mappeddata_8psk(k) = s100_8psk;
    end
    k = k+1;
end

channeldata = mappeddata_8psk + n_8psk*sqrt(No/2);
outputdata_8psk = zeros(1,120000);
err = 0;
k = 1;

%%%%%%%%%%%%%%%% Demapping %%%%%%%%%%%%%%%%%%%%%
for p = 1:3:120000
    if angle(channeldata(k)) <= pi/8 && angle(channeldata(k)) > -pi/8
        outputdata_8psk(p) = 0;
        outputdata_8psk(p+1) = 0;
        outputdata_8psk(p+2) = 0;
    elseif angle(channeldata(k)) <= 3*pi/8 && angle(channeldata(k)) > pi/8
        outputdata_8psk(p) = 0;
        outputdata_8psk(p+1) = 0;
        outputdata_8psk(p+2) = 1;
    elseif angle(channeldata(k)) <= 5*pi/8 && angle(channeldata(k)) > 3*pi/8
        outputdata_8psk(p) = 0;
        outputdata_8psk(p+1) = 1;
        outputdata_8psk(p+2) = 1;
    elseif angle(channeldata(k)) <= 7*pi/8 && angle(channeldata(k)) > 5*pi/8
        outputdata_8psk(p) = 0;
        outputdata_8psk(p+1) = 1;
        outputdata_8psk(p+2) = 0;
    elseif angle(channeldata(k)) > 7*pi/8 && angle(channeldata(k)) <= pi
        outputdata_8psk(p) = 1;
        outputdata_8psk(p+1) = 1;
        outputdata_8psk(p+2) = 0;
    elseif angle(channeldata(k)) <= -7*pi/8 && angle(channeldata(k)) >= -pi
        outputdata_8psk(p) = 1;
        outputdata_8psk(p+1) = 1;
        outputdata_8psk(p+2) = 0;
    elseif angle(channeldata(k)) <= -5*pi/8 && angle(channeldata(k)) > -7*pi/8
```

```
        outputdata_8psk(p) = 1;
        outputdata_8psk(p+1) = 1;
        outputdata_8psk(p+2) = 1;
    elseif angle(channeldata(k)) <= -3*pi/8 && angle(channeldata(k)) > -5*pi/8
        outputdata_8psk(p) = 1;
        outputdata_8psk(p+1) = 0;
        outputdata_8psk(p+2) = 1;
    elseif angle(channeldata(k)) <= -pi/8 && angle(channeldata(k)) > -3*pi/8
        outputdata_8psk(p) = 1;
        outputdata_8psk(p+1) = 0;

        outputdata_8psk(p+2) = 0;
    end
    k = k+1;
end

for p = 1:120000
    if outputdata_8psk(p) ~= A(p)
        err = err + 1;
    end
end

%%%%%%%%%%%%%%%% BER %%%%%%%%%%%%%%%%%%%%
BER_8psk(j+3) = err/120000;
end

theoretical_BER_8psk = 1/3 *erfc(sqrt(3*(10.^((-2:5)./10)))*sin(pi/8));
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% 16-QAM %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Es_QAM = 10;
Eb_QAM = Es_QAM/4;
s0000 = -3-3*i;
s0001 = -3-1*i;
s0011 = -3+1*i;
s0010 = -3+3*i;
s0100 = -1-3*i;
s0101 = -1-1*i;
s0111 = -1+1*i;
s0110 = -1+3*i;
s1100 = 1-3*i;
s1101 = 1-1*i;
s1111 = 1+1*i;
s1110 = 1+3*i;
s1000 = 3-3*i;
s1001 = 3-1*i;
s1011 = 3+1*i;
s1010 = 3+3*i;

mappeddata_QAM = zeros(1,30000);
n_QAM = randn(1,30000) + i*randn(1,30000);

for j = -2:5
    z = 10^(j/10);
    No = Eb_QAM/z;
k = 1;
```

```matlab
%%%%%%%%%%%%%%% Mapping %%%%%%%%%%%%%%%%%%%
for p = 1:4:120000
    if A(p) == 0 && A(p+1) == 0 && A(p+2) == 0 && A(p+3) == 0
       mappeddata_QAM(k) = s0000;
    elseif A(p) == 0 && A(p+1) == 0 && A(p+2) == 0 && A(p+3) == 1
       mappeddata_QAM(k) = s0001;
    elseif A(p) == 0 && A(p+1) == 0 && A(p+2) == 1 && A(p+3) == 0
       mappeddata_QAM(k) = s0010;
    elseif A(p) == 0 && A(p+1) == 0 && A(p+2) == 1 && A(p+3) == 1
       mappeddata_QAM(k) = s0011;
    elseif A(p) == 0 && A(p+1) == 1 && A(p+2) == 0 && A(p+3) == 0
       mappeddata_QAM(k) = s0100;
    elseif A(p) == 0 && A(p+1) == 1 && A(p+2) == 0 && A(p+3) == 1
       mappeddata_QAM(k) = s0101;
    elseif A(p) == 0 && A(p+1) == 1 && A(p+2) == 1 && A(p+3) == 0
       mappeddata_QAM(k) = s0110;
    elseif A(p) == 0 && A(p+1) == 1 && A(p+2) == 1 && A(p+3) == 1
       mappeddata_QAM(k) = s0111;
    elseif A(p) == 1 && A(p+1) == 0 && A(p+2) == 0 && A(p+3) == 0
       mappeddata_QAM(k) = s1000;
    elseif A(p) == 1 && A(p+1) == 0 && A(p+2) == 0 && A(p+3) == 1
       mappeddata_QAM(k) = s1001;
    elseif A(p) == 1 && A(p+1) == 0 && A(p+2) == 1 && A(p+3) == 0
       mappeddata_QAM(k) = s1010;
    elseif A(p) == 1 && A(p+1) == 0 && A(p+2) == 1 && A(p+3) == 1
       mappeddata_QAM(k) = s1011;
    elseif A(p) == 1 && A(p+1) == 1 && A(p+2) == 0 && A(p+3) == 0
       mappeddata_QAM(k) = s1100;
    elseif A(p) == 1 && A(p+1) == 1 && A(p+2) == 0 && A(p+3) == 1
       mappeddata_QAM(k) = s1101;
    elseif A(p) == 1 && A(p+1) == 1 && A(p+2) == 1 && A(p+3) == 0
       mappeddata_QAM(k) = s1110;
    elseif A(p) == 1 && A(p+1) == 1 && A(p+2) == 1 && A(p+3) == 1
       mappeddata_QAM(k) = s1111;
    end
    k = k+1;
end
channeldata = mappeddata_QAM + n_QAM*sqrt(No/2);
outputdata_QAM = zeros(1,120000);
err = 0;
k = 1;
%%%%%%%%%%%%%%% Demapping %%%%%%%%%%%%%%%%%%%
for p = 1:4:120000
    if real(channeldata(k)) <= -2 && imag(channeldata(k)) <= -2
        outputdata_QAM(p) = 0;
        outputdata_QAM(p+1) = 0;
        outputdata_QAM(p+2) = 0;
        outputdata_QAM(p+3) = 0;
    elseif real(channeldata(k)) <= -2 && imag(channeldata(k)) > -2 &&
imag(channeldata(k)) < 0
        outputdata_QAM(p) = 0;
        outputdata_QAM(p+1) = 0;
        outputdata_QAM(p+2) = 0;
        outputdata_QAM(p+3) = 1;
```

```
    elseif real(channeldata(k)) <= -2 && imag(channeldata(k)) > 0 &&
imag(channeldata(k)) < 2
        outputdata_QAM(p) = 0;
        outputdata_QAM(p+1) = 0;
        outputdata_QAM(p+2) = 1;
        outputdata_QAM(p+3) = 1;
    elseif real(channeldata(k)) <= -2 && imag(channeldata(k)) > 2
        outputdata_QAM(p) = 0;
        outputdata_QAM(p+1) = 0;
        outputdata_QAM(p+2) = 1;
        outputdata_QAM(p+3) = 0;
    elseif real(channeldata(k)) > -2 && real(channeldata(k)) <= 0 &&
imag(channeldata(k)) <= -2
        outputdata_QAM(p) = 0;

        outputdata_QAM(p+1) = 1;
        outputdata_QAM(p+2) = 0;
        outputdata_QAM(p+3) = 0;
    elseif real(channeldata(k)) > -2 && real(channeldata(k)) <= 0 &&
imag(channeldata(k)) > -2 && imag(channeldata(k)) < 0
        outputdata_QAM(p) = 0;
        outputdata_QAM(p+1) = 1;
        outputdata_QAM(p+2) = 0;
        outputdata_QAM(p+3) = 1;
    elseif real(channeldata(k)) > -2 && real(channeldata(k)) <= 0 &&
imag(channeldata(k)) > 0 && imag(channeldata(k)) < 2
        outputdata_QAM(p) = 0;
        outputdata_QAM(p+1) = 1;
        outputdata_QAM(p+2) = 1;
        outputdata_QAM(p+3) = 1;
    elseif real(channeldata(k)) > -2 && real(channeldata(k)) <= 0 &&
imag(channeldata(k)) > 2
        outputdata_QAM(p) = 0;
        outputdata_QAM(p+1) = 1;
        outputdata_QAM(p+2) = 1;
        outputdata_QAM(p+3) = 0;
    elseif real(channeldata(k)) > 0 && real(channeldata(k)) <= 2 &&
imag(channeldata(k)) <= -2
        outputdata_QAM(p) = 1;
        outputdata_QAM(p+1) = 1;
        outputdata_QAM(p+2) = 0;
        outputdata_QAM(p+3) = 0;
    elseif real(channeldata(k)) > 0 && real(channeldata(k)) <= 2 &&
imag(channeldata(k)) > -2 && imag(channeldata(k)) < 0
        outputdata_QAM(p) = 1;
        outputdata_QAM(p+1) = 1;
        outputdata_QAM(p+2) = 0;
        outputdata_QAM(p+3) = 1;
    elseif real(channeldata(k)) > 0 && real(channeldata(k)) <= 2 &&
imag(channeldata(k)) > 0 && imag(channeldata(k)) < 2
        outputdata_QAM(p) = 1;
        outputdata_QAM(p+1) = 1;
```

```
        outputdata_QAM(p+2) = 1;
        outputdata_QAM(p+3) = 1;
    elseif real(channeldata(k)) > 0 && real(channeldata(k)) <= 2 &&
imag(channeldata(k)) > 2
        outputdata_QAM(p) = 1;
        outputdata_QAM(p+1) = 1;
        outputdata_QAM(p+2) = 1;
        outputdata_QAM(p+3) = 0;
    elseif real(channeldata(k)) > 2 && imag(channeldata(k)) <= -2
        outputdata_QAM(p) = 1;
        outputdata_QAM(p+1) = 0;
        outputdata_QAM(p+2) = 0;
        outputdata_QAM(p+3) = 0;
    elseif real(channeldata(k)) > 2 && imag(channeldata(k)) > -2 &&
imag(channeldata(k)) < 0
        outputdata_QAM(p) = 1;
        outputdata_QAM(p+1) = 0;
        outputdata_QAM(p+2) = 0;
        outputdata_QAM(p+3) = 1;

    elseif real(channeldata(k)) >2 && imag(channeldata(k)) > 0 &&
imag(channeldata(k)) < 2
        outputdata_QAM(p) = 1;
        outputdata_QAM(p+1) = 0;
        outputdata_QAM(p+2) = 1;
        outputdata_QAM(p+3) = 1;
    elseif real(channeldata(k)) > 2 && imag(channeldata(k)) > 2
        outputdata_QAM(p) = 1;
        outputdata_QAM(p+1) = 0;
        outputdata_QAM(p+2) = 1;
        outputdata_QAM(p+3) = 0;
    end
    k = k+1;
end
for p = 1:120000
    if outputdata_QAM(p) ~= A(p)
        err = err + 1;
    end
end
%%%%%%%%%%%%%%%% BER %%%%%%%%%%%%%%%%%%%%
BER_QAM(j+3) = err/120000;
end
theoretical_BER_QAM = (3/8)*erfc(sqrt((10.^((-2:5)./10))/2.5));


%%%%%%%%%%%%%%%% Plotting %%%%%%%%%%%%%%%%%%%%
figure
semilogy((-2:5) , BER_BPSK,'color','#0072BD');
hold on
semilogy((-2:5) , theoretical_BER_BPSK,'--','color','#D95319');
hold on
semilogy((-2:5) , BER_QPSK,'color','#EBD120');
```

```
hold on
semilogy((-2:5) , theoretical_BER_QPSK,'--','color','#7E2F8E');
hold on
semilogy((-2:5) , BER_8psk,'color','#77AC30');
hold on
semilogy((-2:5) , theoretical_BER_8psk,'--','color','#4DBEEE');
hold on
semilogy((-2:5) , BER_QAM,'color','#FF69B4');
hold on
semilogy((-2:5) , theoretical_BER_QAM,'--','color','#000000');
title('BER Simulated and theoritical calculations');
legend('BPSK BER','Theoritical BPSK BER','QPSK BER','Theoritical QPSK BER','8PSK
BER','Theoritical 8PSK BER','16-QAM BER','Theoritical 16-QAM BER');
xlabel('Eb/No');
ylabel('BER');

figure
semilogy((-2:5) , BER_QPSK,'color','r');
hold on
semilogy((-2:5) , BER2_QPSK,'color','b');
title('BER of gray code QPSK ,BER of binary code QPSK');
legend('Gray code QPSK BER','binary code QPSK BER');
xlabel('Eb/No');
ylabel('BER');
```

2. BFSK

```
clc
clear all
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% BFSK BER %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
A = randi( [0,1] , 1 , 120000);
Es = 1;
Eb = Es;
s1 = sqrt(Es);
s2 = sqrt(Es)*1i;
mappeddata = zeros(1,120000);
n = randn(1,120000) + 1i*randn(1,120000);

for j = -2:5
    z = 10^(j/10);
    No = Es/z;

for k = 1:120000
    if A(k) == 1
       mappeddata(k) = s1;
    else
       mappeddata(k) = s2;
    end
end

channeldata = mappeddata + n*sqrt(No/2);
outputdata = zeros(1,120000);
err = 0;
for k = 1:120000
    if angle(channeldata(k)) >= -3*pi/4 && angle(channeldata(k)) < pi/4
```

```
        outputdata(k) = 1;
    else
        outputdata(k) = 0;
    end
    if outputdata(k) ~= A(k)
        err = err + 1;
    end
end

BER(j+3) = err/120000;

end

%%%%%%%%%%% Plotting BER %%%%%%%%%%%%%%
theoretical_BER = 0.5 *erfc(sqrt((10.^((-2:5)./10))/2));
figure
semilogy((-2:5) , BER,'r');
hold on
semilogy((-2:5) , theoretical_BER,'b');
title('BER of BFSK and theoritical calculations');
legend('BFSK BER','Theoritical BER');
xlabel('Eb/No');
ylabel('BER');
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% BFSK PSD %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Tb = 7;
F = 1/Tb;
samples_num = 7;
t = 0:Tb/samples_num:Tb;
A = randi([0,1],500,101);
Eb = 1;

s1bb(1:samples_num) = sqrt(2*Eb/Tb);
s2bb = sqrt(2*Eb/Tb)*(cos(2*pi*F*t)+i*sin(2*pi*F*t));

Data = zeros(500, 101*samples_num);
 for i = 1:500
     mappeddata =0;
     for j = 1:101
        if A(i,j) == 1
           mappeddata = [mappeddata s1bb];
        else
           mappeddata = [mappeddata s2bb(1:samples_num)];
        end
     end
     Data(i,:) = mappeddata(2:101*samples_num+1);
 end

  td = randi([1,samples_num],500,1);
  B = Data(1, td(1):td(1)+100*samples_num-1);
  Data2 = B;
  for i=2:500
      B = Data(i, td(i):td(i)+100*samples_num-1);
```

```
      Data2 = [Data2; B];
  end

  Auto_corr_real(1:100*samples_num) = 0;
for j=1:100*samples_num
    for i=1:500
        Auto_corr_real(j) = Auto_corr_real(j) + (conj(Data2(i,1)) * Data2(i,j));
    end
    Auto_corr_real(j) = Auto_corr_real(j)/500;
end
y = [fliplr(conj(Auto_corr_real(1:end))) Auto_corr_real];

%%%%%%%%%%%% Plotting PSD %%%%%%%%%%%%%%%
DataL = length(y(1,:));
Dataf = F*samples_num*(-(DataL/2):(DataL/2)-1)/DataL;
x = fftshift(fft(y(1,:)));
figure
plot(Dataf,abs(x)/DataL);
title('PSD')
xlabel('Frequency')
ylabel('Power')
```