Table of Contents

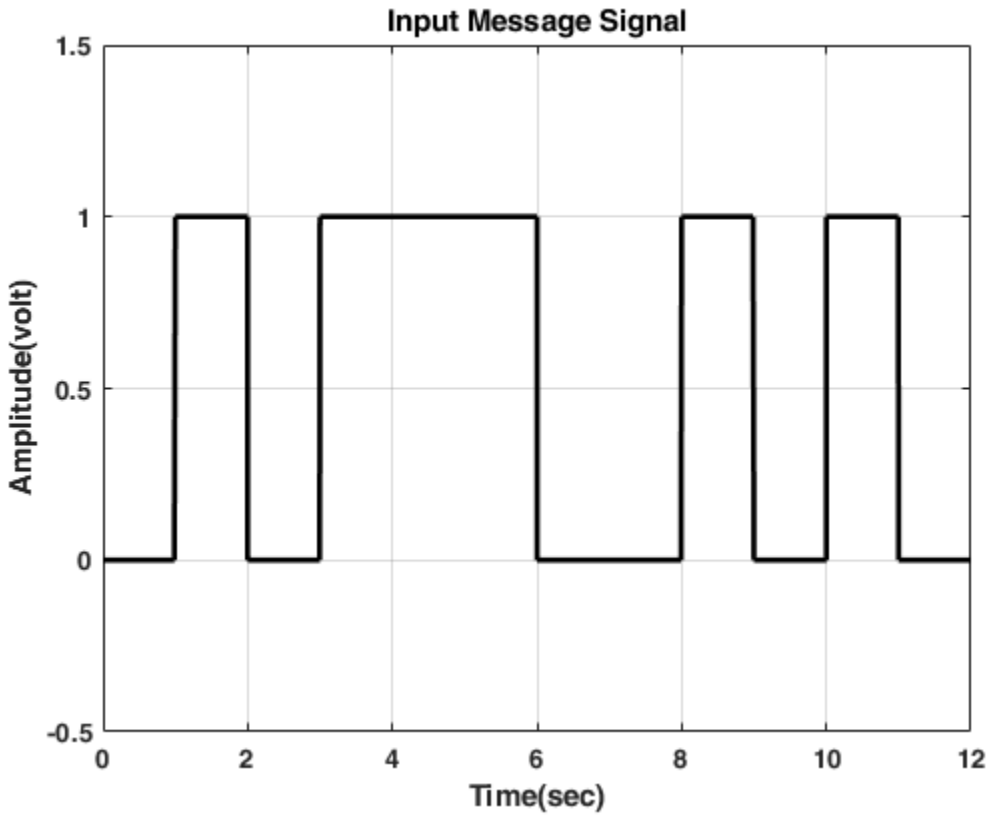

```
clc;
clear;
```

1) BPSK

Define transmitted signal (BPSK)

```
N=12; %Number of bits
X_input= randi([0, 1],1,N);  %Binary signal
Tb=1; %Bit duration = 1 sec
X_digit=[];
nb=10000;  %Number of points between two symbols (it's used to convert
 the symbols into continuous digital signal)
for i=1:N
    if X_input(i)==1
       x_temp=ones(1,nb);
    else
        x_temp=zeros(1,nb);
    end
    X_digit=[X_digit x_temp];
end
t_sig = Tb/nb : Tb/nb : N*Tb; %Time vector of continuous digital
 signal
```

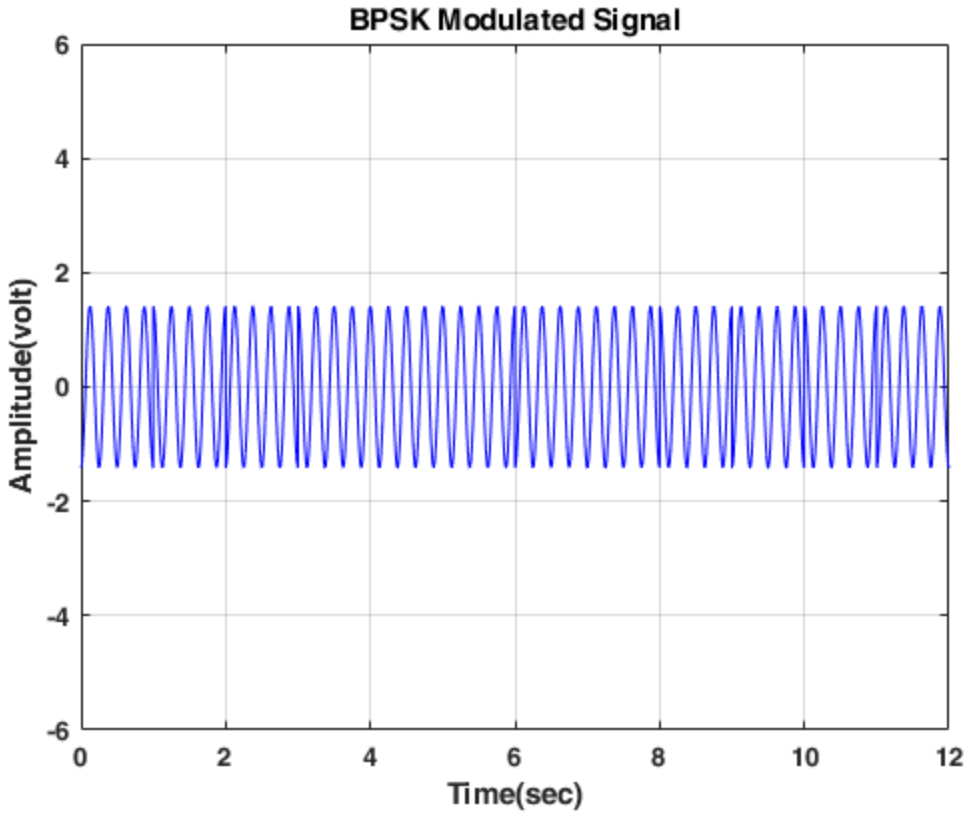
```
%Plotting the input message signal
figure();
plot(t_sig,X_digit, 'LineWidth',2,'Color','black'); grid on; xlim([0
 Tb*N]); ylim([-0.5 1.5]);
xlabel('Time(sec)'); ylabel('Amplitude(volt)'); title('Input Message
 Signal');
```



BPSK Modulation (BPSK)

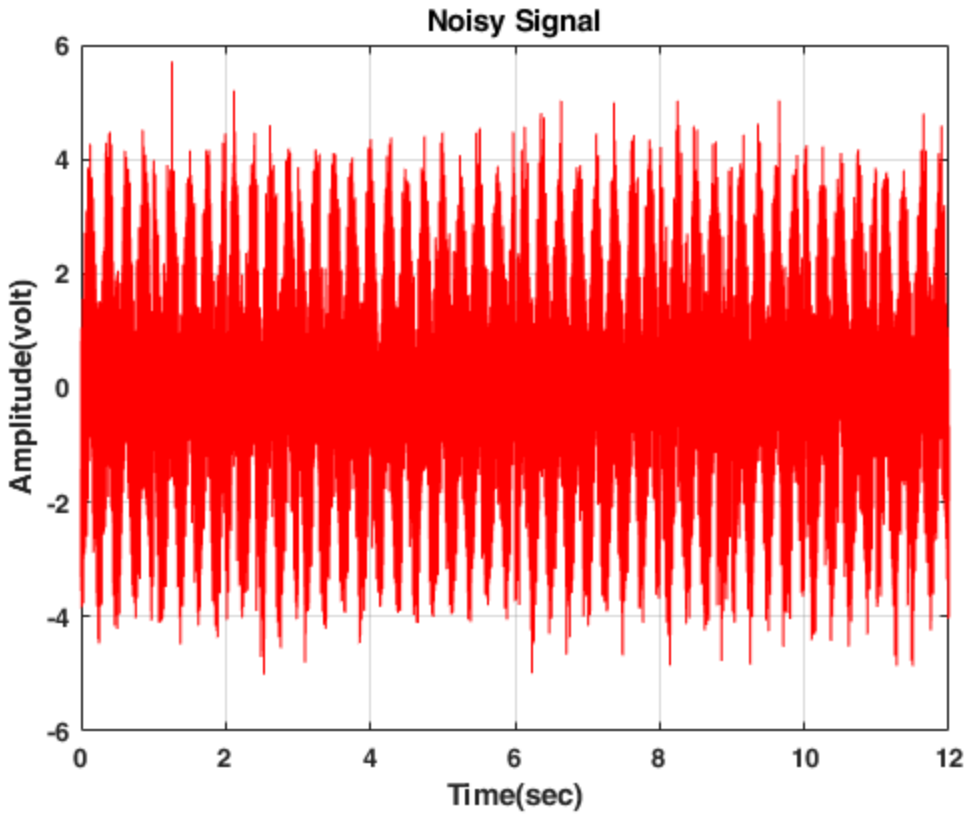
```
Ac=sqrt(2/Tb);   %Carrier Signal Amplitude
fc = 4*(1/Tb);  %Carrier Signal Frequency
phi_1 = 0;       %Phase Shift for bit '1'
phi_0 = pi;      %Phase Shift for bit '0'
t_cycle = Tb/nb : Tb/nb : Tb;  %Time of one symbol to be used to
 calculate the carrier signal to be multiplied by the input signal
X_BPSK=[];
x_con_mod = [];
%Multiplying the input message signal by the carrier based on the
 message symbol
for i=1:N
    if X_input(i)==1
        x_temp_mod = Ac*cos(2*pi*fc*t_cycle + phi_1);
        x_con_mod = [x_con_mod, 1];
    else
        x_temp_mod = Ac*cos(2*pi*fc*t_cycle + phi_0);
        x_con_mod = [x_con_mod, -1];
```

```
    end
    X_BPSK=[X_BPSK x_temp_mod];
end
t_mod = Tb/nb : Tb/nb : N*Tb; %Time of the modulated signal
%Plotting the BPSK Signal
figure();
plot(t_mod,X_BPSK,'Color','blue'); grid on;
xlabel('Time(sec)'); ylabel('Amplitude(volt)'); title('BPSK Modulated
 Signal'); ylim([-6 6]);
```



Noise in the Communication Channel (BPSK)

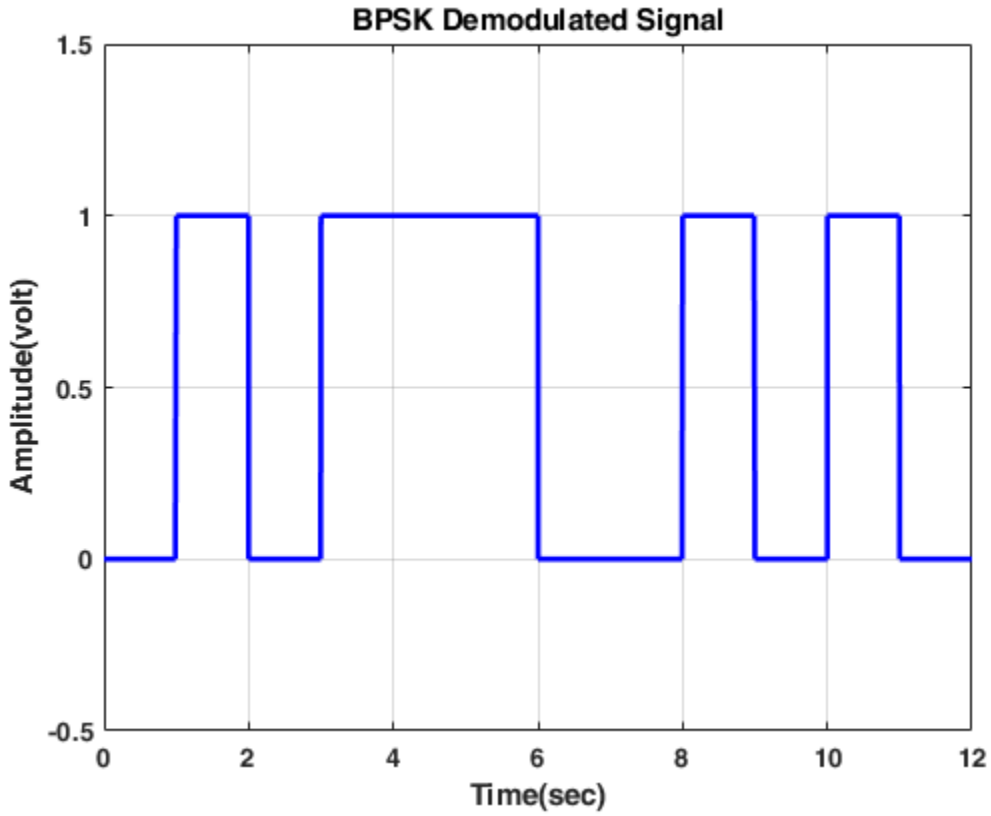
```
Y = awgn(X_BPSK,0.000001,'measured');  %Adds white Gaussian noise to
 the Modulated signal
%Plotting the Noisy Signal
figure();
plot(t_mod,Y,'Color','red'); grid on;
xlabel('Time(sec)'); ylabel('Amplitude(volt)'); title('Noisy Signal');
 ylim([-6 6]);
```



BPSK Demodulation (BPSK)

```
X_demod=[];
x_con_dem = [];
for i=nb:nb:length(Y)
  t1_dem = Tb/nb:Tb/nb:Tb;                  %Time of one symbol
  Ac_dem = sqrt(2/Tb);                      %Carrier Amplitude
  C_dem = Ac_dem*cos(2*pi*fc*t1_dem);       %Carrier that will be used
 in Coherent Detection
  %Correlator
  y_temp_dem = C_dem.*Y((i-(nb-1)):i);      %Multiply with the carrier
  y_corr=trapz(t1_dem,y_temp_dem);               %Integrate over the
 time period of the symbol
  x_con_dem = [x_con_dem y_corr];

  %Decision Making Device
  if(y_corr>0) %If the value > threshold (0)
    S=1;       %Then symbol = 1
  else         %If the value < threshold (0)
    S=0;       %Then symbol = 0
  end
  X_demod=[X_demod S];
end

%Convert the symbols into continuous digital signal
```

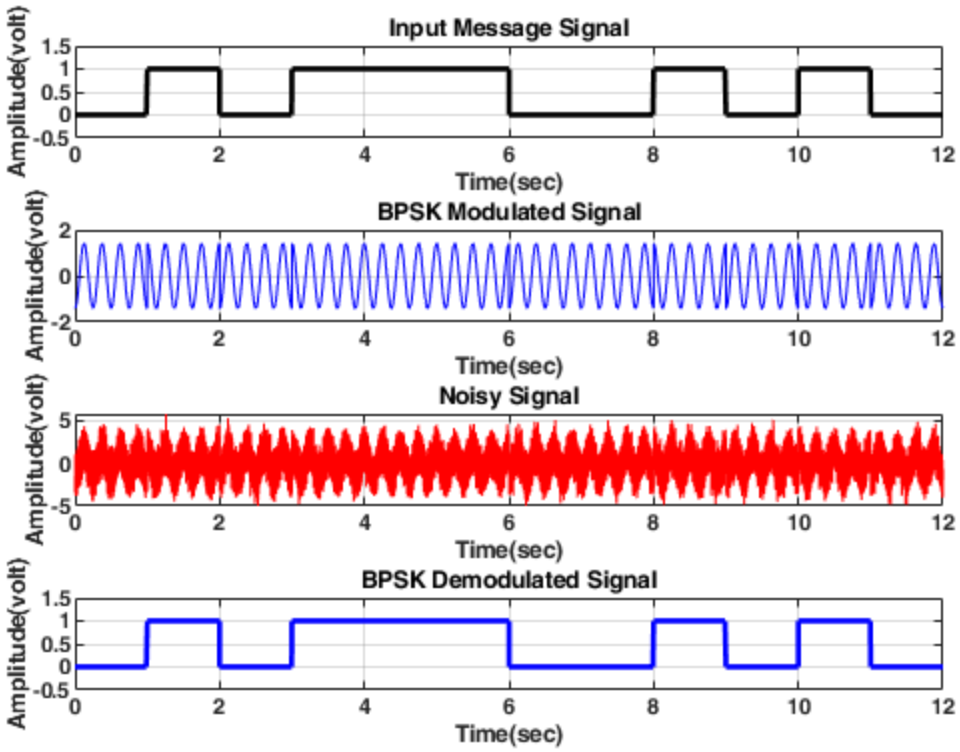
```
X_dem_sig = [];
for i=1:length(X_demod)
    if X_demod(i)==1
       x_temp_dem=ones(1,nb);
    else
        x_temp_dem=zeros(1,nb);
    end
    X_dem_sig=[X_dem_sig x_temp_dem];
end
t_sig_dem = Tb/nb : Tb/nb : length(X_demod)*Tb; %Time vector of
 continuous digital signal
figure();
plot(t_sig_dem,X_dem_sig, 'LineWidth',2,'Color','blue'); grid on;
 xlim([0 Tb*length(X_demod)]); ylim([-0.5 1.5]);
xlabel('Time(sec)'); ylabel('Amplitude(volt)'); title('BPSK
 Demodulated Signal');
```



Graphs in Subplot (BPSK)

```
%Plotting the input message signal
figure();
subplot(4,1,1);
plot(t_sig,X_digit, 'LineWidth',2,'Color','black'); grid on; xlim([0
 Tb*N]); ylim([-0.5 1.5]);
xlabel('Time(sec)'); ylabel('Amplitude(volt)'); title('Input Message
 Signal');
```

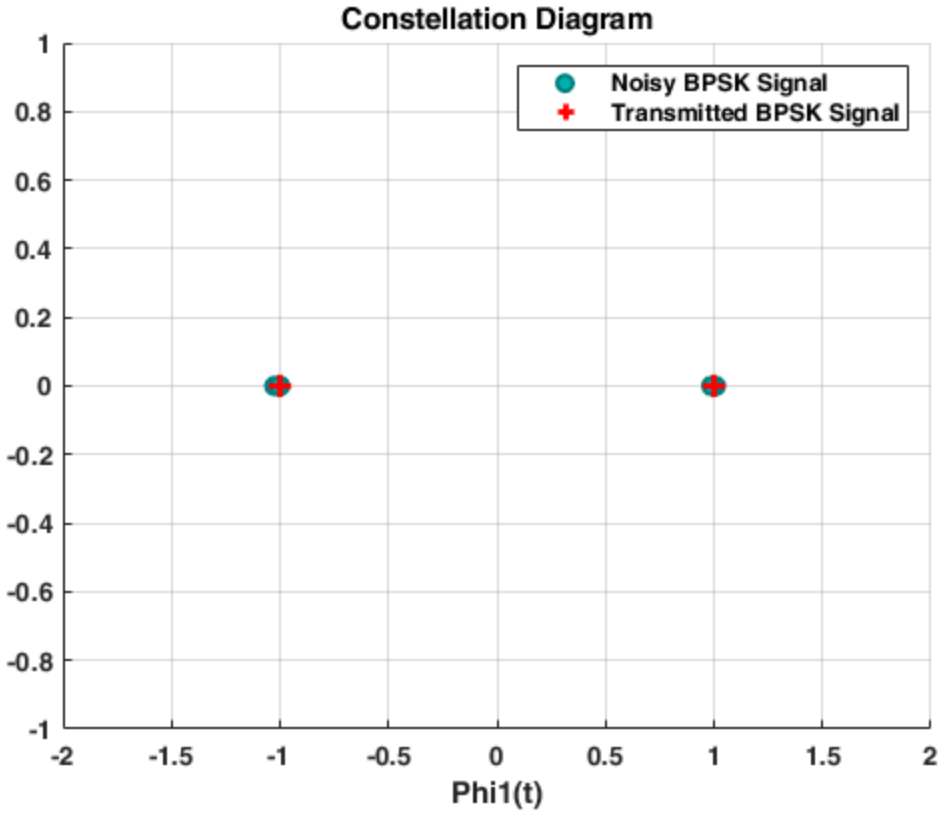
```
%Plotting the BPSK Signal
subplot(4,1,2);
plot(t_mod,X_BPSK,'Color','blue'); grid on;
xlabel('Time(sec)'); ylabel('Amplitude(volt)'); title('BPSK Modulated
 Signal');

%Plotting the Noisy Signal
subplot(4,1,3);
plot(t_mod,Y,'Color','red'); grid on;
xlabel('Time(sec)'); ylabel('Amplitude(volt)'); title('Noisy Signal');

%Plotting the Demodulated BPSK Signal
subplot(4,1,4);
plot(t_sig_dem,X_dem_sig, 'LineWidth',2,'Color','blue'); grid on;
 xlim([0 Tb*length(X_demod)]); ylim([-0.5 1.5]);
xlabel('Time(sec)'); ylabel('Amplitude(volt)'); title('BPSK
 Demodulated Signal');
```



Constellation Diagram (BPSK)

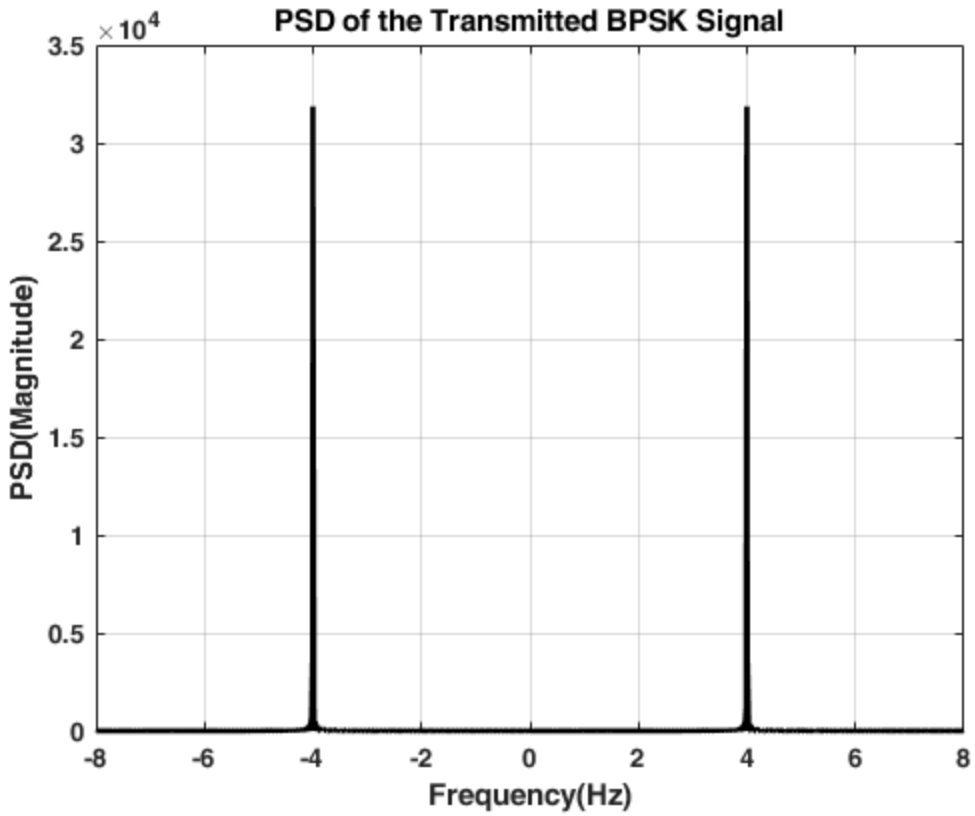
```
figure();
scatter(x_con_dem, zeros(1,length(x_con_dem)),40, 'MarkerEdgeColor',
 [0 .5 .5],'MarkerFaceColor',[0 .7 .7], 'LineWidth',1.5); grid on;
hold on
```

```
scatter(x_con_mod, zeros(1,length(x_con_mod)),
 70, 'red', '+', 'LineWidth',2);
hold off
ylim([-1 1]); xlim([-2 2]);
xlabel('Phi1(t)'); title('Constellation Diagram'); legend('Noisy BPSK
 Signal','Transmitted BPSK Signal')
```



PSD (BPSK)

```
psd_BPSK = fft(X_BPSK);  %FFT for the Modulated Signal
psds_BPSK = fftshift(psd_BPSK);  %FFT Shift
psd_neg = flip(psds_BPSK);   %The negative side frequencies
psd = [psd_neg psds_BPSK];   %Combining the frequencies of both sides
f = linspace(-2*fc, 2*fc, length(psd));  %The frequency Vector
%Plotting PSD
figure();
plot(f, abs(psd), 'LineWidth',2,'Color','black'); grid on;
xlabel('Frequency(Hz)'); ylabel('PSD(Magnitude)'); title('PSD of the
 Transmitted BPSK Signal');
```



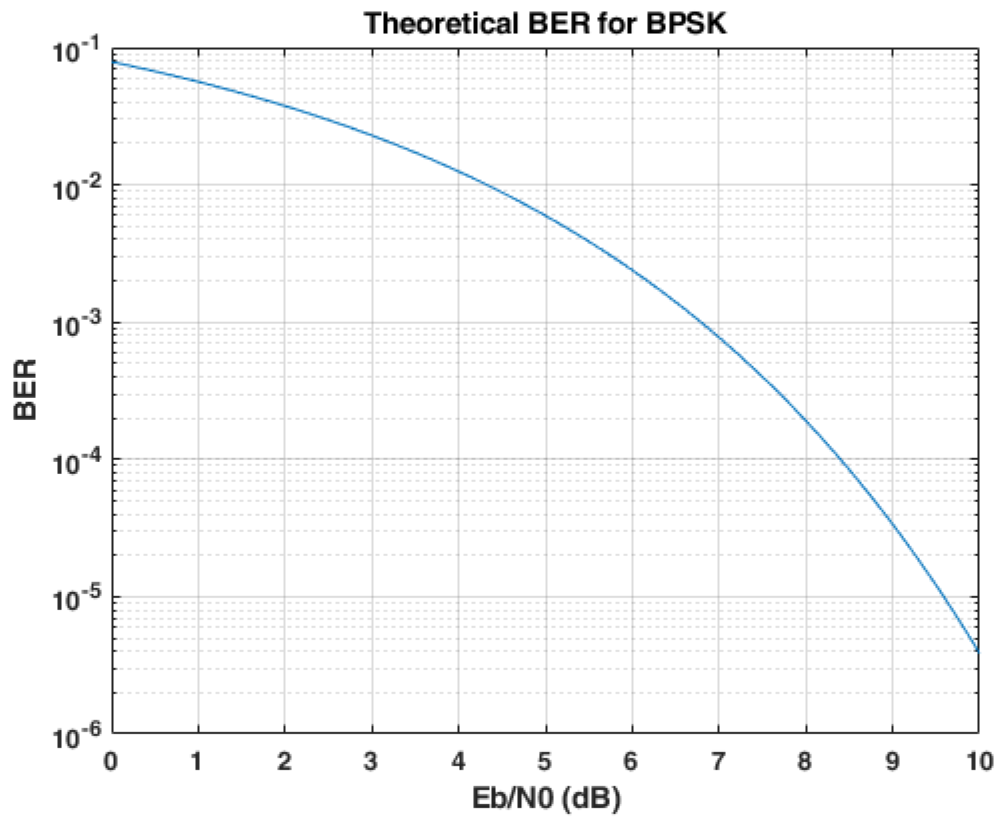
BER (BPSK)

```
Bit_Error_Rate = biterr(X_demod, X_input)
```

```
Bit_Error_Rate =

     0
```

Theoretical BER (PBSK)

```
clc;
clear;
Eb_N0_dB = 0:0.1:10;
Eb_N0 = 10.^(Eb_N0_dB/10);
x= sqrt(Eb_N0);
BER = 1/2.*erfc(x);
figure();
semilogy(Eb_N0_dB,BER); grid on; ylabel('BER'); xlabel('Eb/N0 (dB)');
 title('Theoretical BER for BPSK');
```



%%

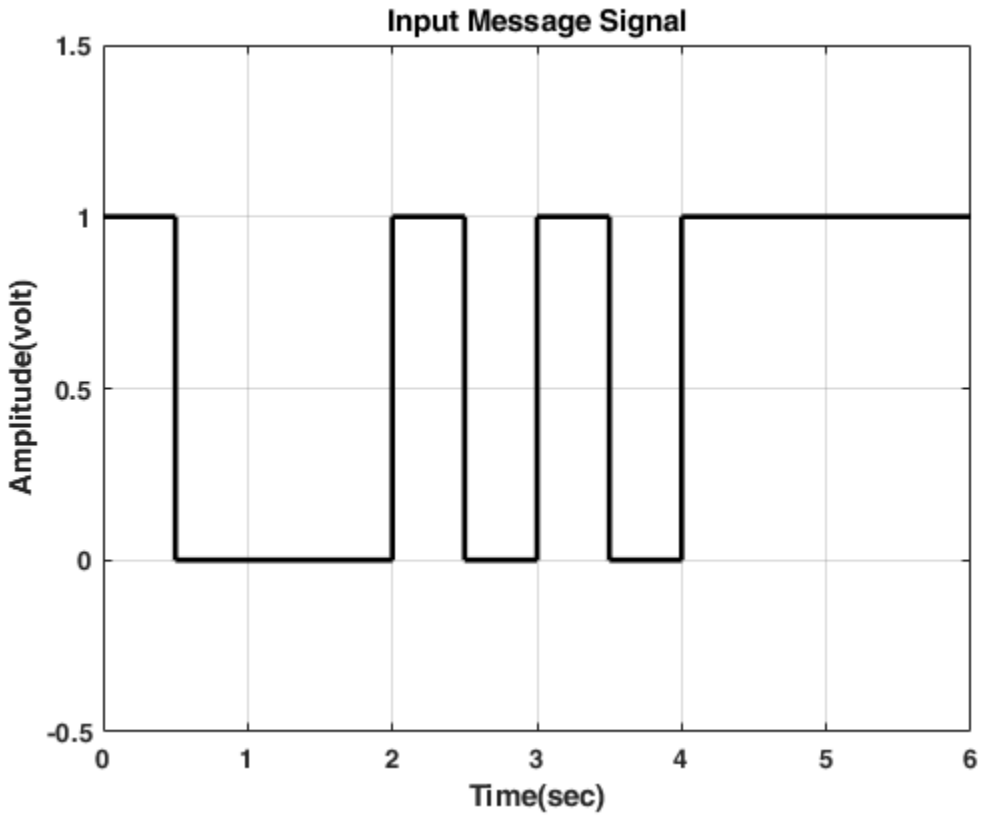
QPSK

```
clc;
clear;
```

Define transmitted signal (QPSK)

```
N=12; %Number of bits
X_input= randi([0, 1],1,N);  %Binary signal
Tb=0.5; %Bit duration = 0.5 sec
X_digit=[];
nb=1000;  %Number of points between two symbols (it's used to convert
 the symbols into continuous digital signal)
for i=1:N
    if X_input(i)==1
       x_temp=ones(1,nb);
```

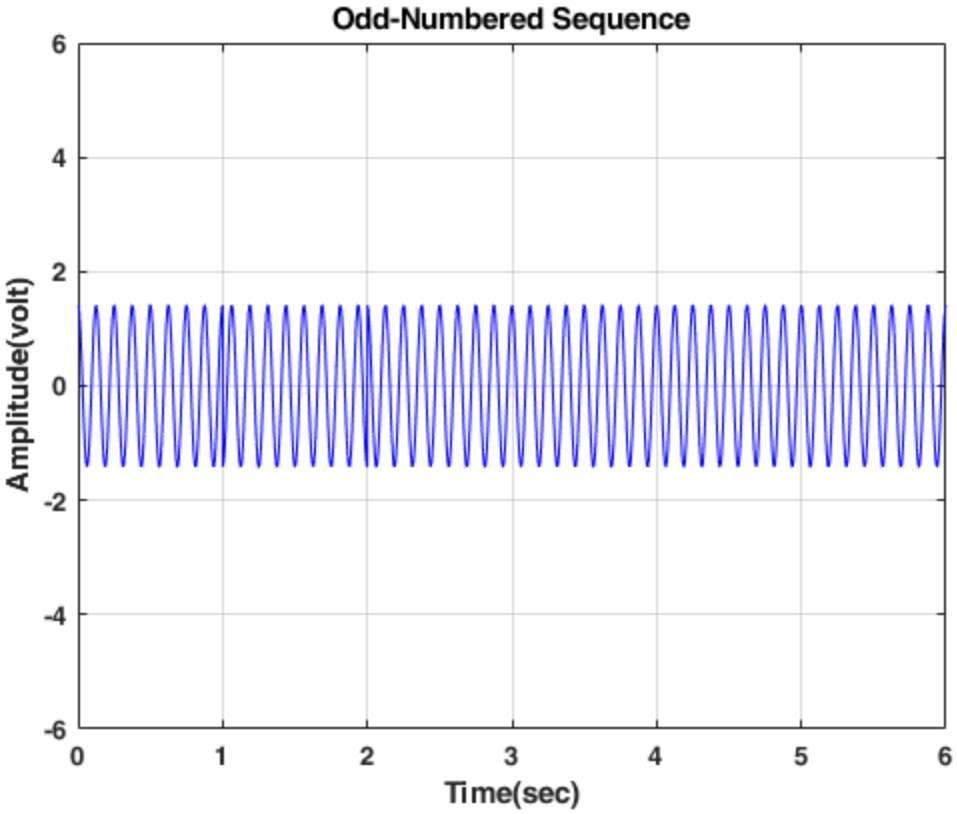
```
    else
        x_temp=zeros(1,nb);
    end
    X_digit=[X_digit x_temp];
end
t_sig = Tb/nb : Tb/nb : N*Tb; %Time vector of continuous digital
 signal
%Plotting the input message signal
figure();
plot(t_sig,X_digit, 'LineWidth',2,'Color','black'); grid on; xlim([0
 Tb*N]); ylim([-0.5 1.5]);
xlabel('Time(sec)'); ylabel('Amplitude(volt)'); title('Input Message
 Signal');
```

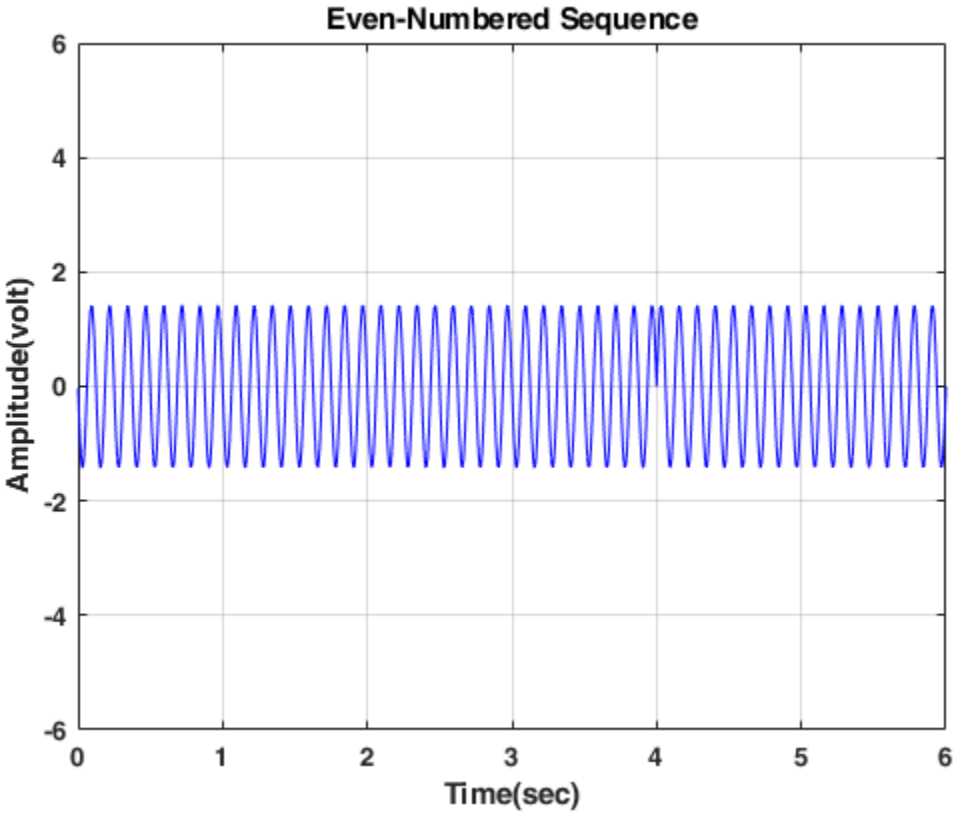


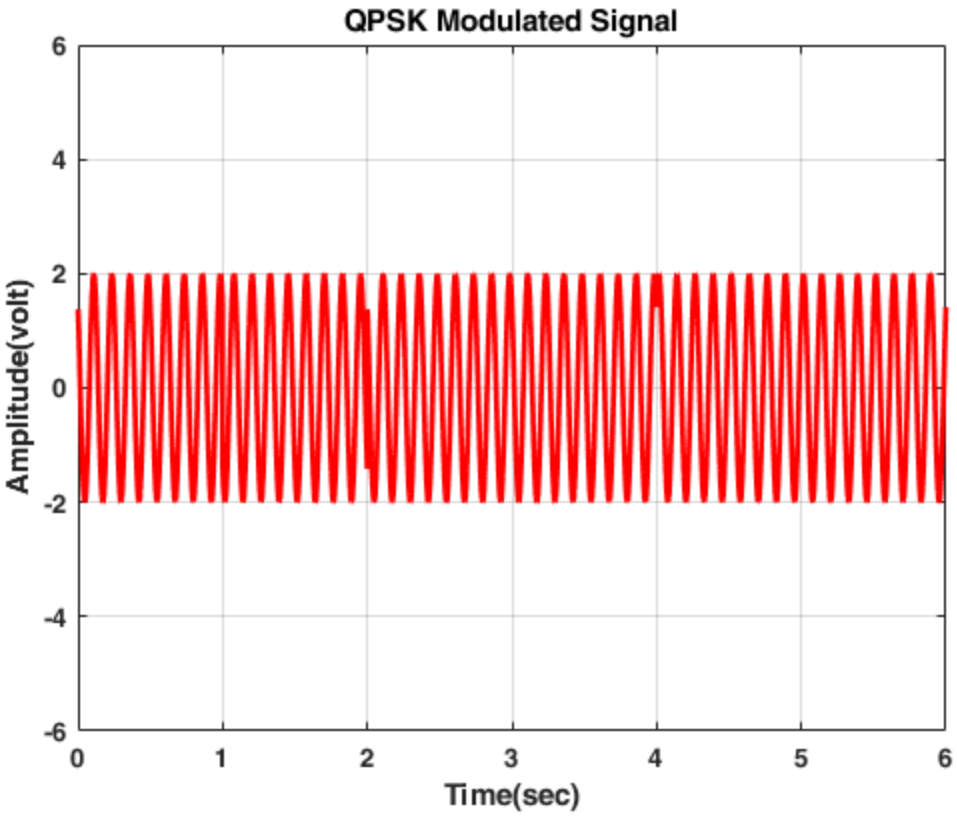
QPSK Modulation

```
Ac=sqrt(1/Tb);   %Carrier Signal Amplitude
fc = 4*(1/Tb);  %Carrier Signal Frequency
t_cycle = Tb/nb : Tb/nb : 2*Tb;  %Time of two symbol (1 Baud)
Q1 = Ac*cos(2*pi*fc*t_cycle);    %Phi_1 Basis function
Q2 = Ac*sin(2*pi*fc*t_cycle);    %Phi_2 Basis function

x1_con_mod = [];
x2_con_mod = [];

Odd_Sig=[];  %Odd Numbered Sequence
```

```
for i=0 : N/2 - 1
    if X_input((2*i+1))==1
        x_temp = Q1;
        x1_con_mod = [x1_con_mod, 1];
    else
        x_temp = -Q1;
        x1_con_mod = [x1_con_mod, -1];
    end
    Odd_Sig=[Odd_Sig x_temp];
end
Even_Sig=[];  %Even Numbered Sequence
for i=1 : N/2
    if X_input(2*i)==1
        x_temp = Q2;
        x2_con_mod = [x2_con_mod, 1];
    else
        x_temp = -Q2;
        x2_con_mod = [x2_con_mod, -1];
    end
    Even_Sig=[Even_Sig x_temp];
end

X_QPSK = Even_Sig + Odd_Sig;  %Adding Odd and Even Sequences to
 generate QPSK Signal
t_mod = Tb/nb : Tb/nb : N*Tb; %Time of the modulated signal

%Plotting the QPSK Signal
figure();
plot(t_mod,Odd_Sig,'Color','blue'); grid on;
xlabel('Time(sec)'); ylabel('Amplitude(volt)'); title('Odd-Numbered
 Sequence'); ylim([-6 6]);
figure();
plot(t_mod,Even_Sig,'Color','blue'); grid on;
xlabel('Time(sec)'); ylabel('Amplitude(volt)'); title('Even-Numbered
 Sequence'); ylim([-6 6]);
figure();
plot(t_mod,X_QPSK,'LineWidth',2, 'Color','red'); grid on;
xlabel('Time(sec)'); ylabel('Amplitude(volt)'); title('QPSK Modulated
 Signal'); ylim([-6 6]);
```

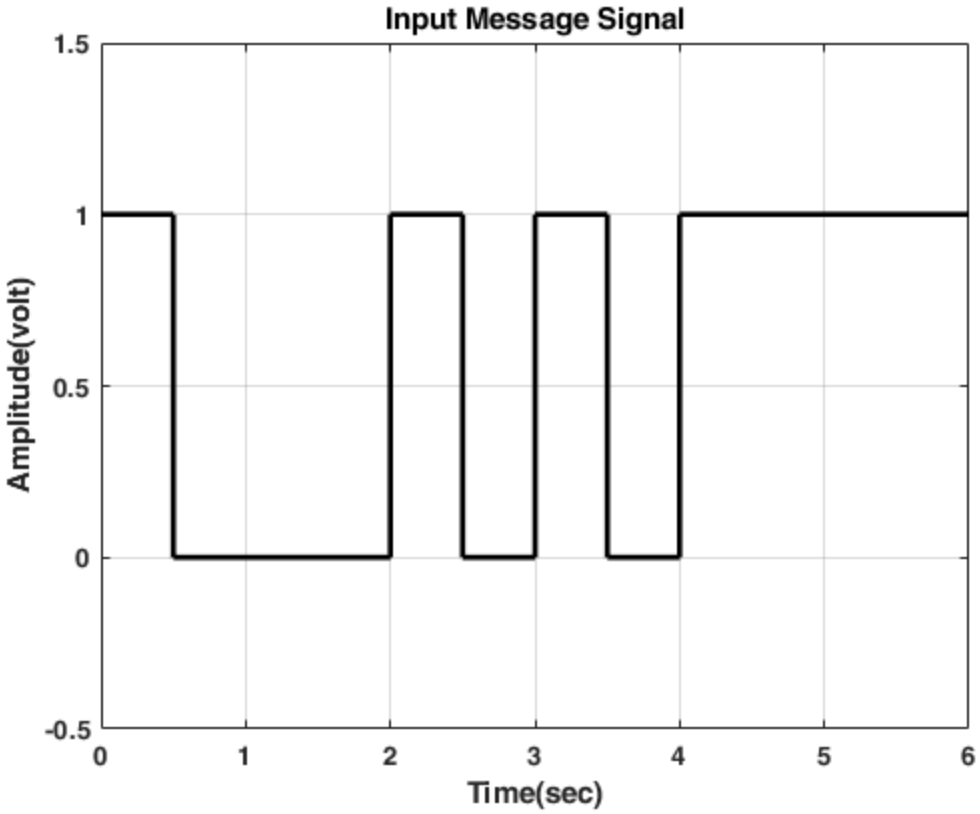

Odd-Numbered Sequence
Amplitude(volt)
Time(sec)
6
4
2
0
-2
-4
-6
0
1
2
3
4
5
6

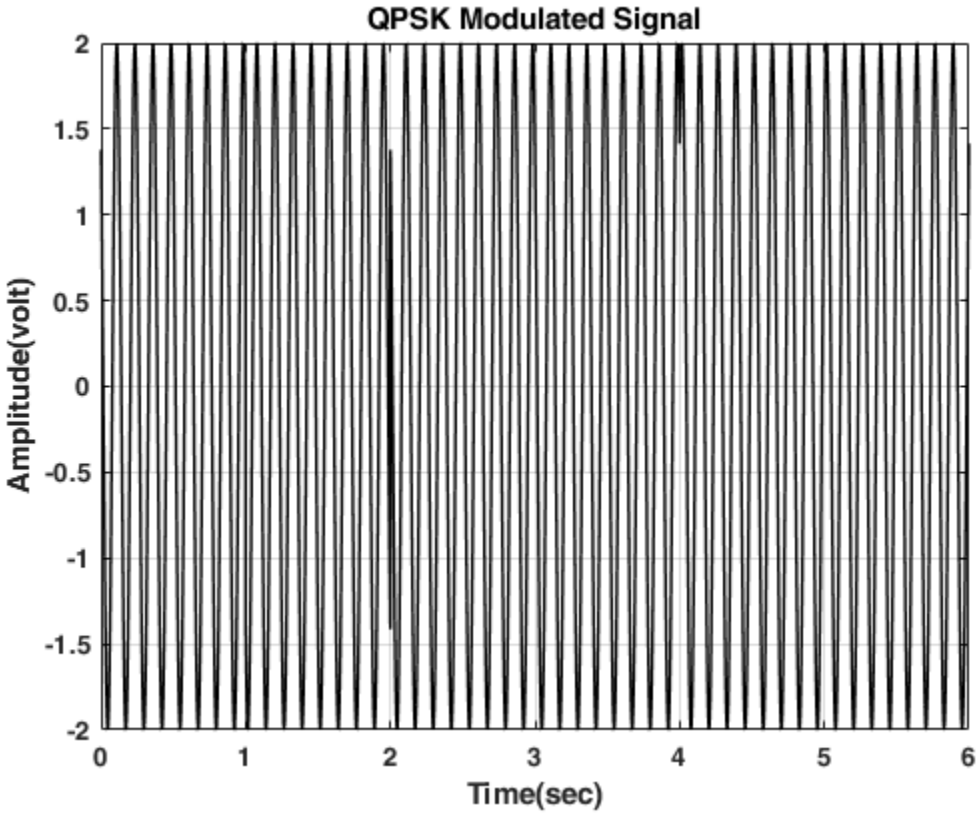

Even-Numbered Sequence
Amplitude(volt)
Time(sec)
6
4
2
0
-2
-4
-6
0
1
2
3
4
5
6

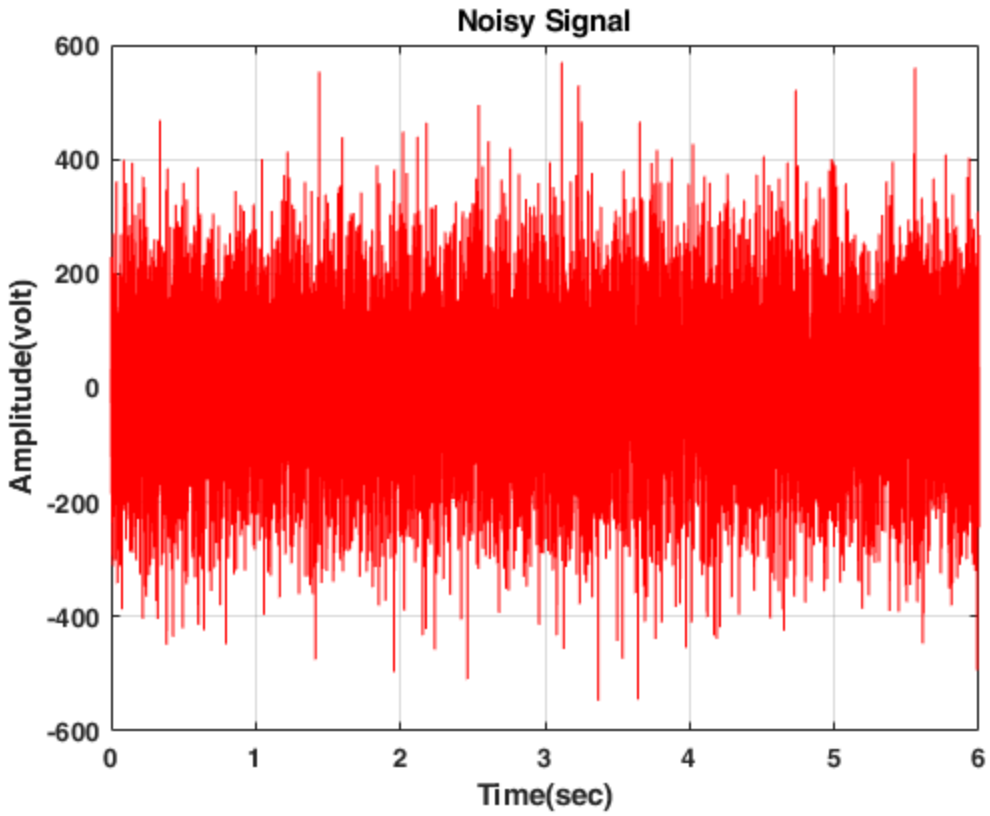


Noise in the Communication Channel (QPSK)

```
Y = awgn(X_QPSK,0.0001,'measured', 'linear');  %Adds white Gaussian
 noise to the Modulated signal

%Plotting the Modulated VS. Noisy Signal
figure();
plot(t_sig,X_digit, 'LineWidth',2,'Color','black'); grid on; xlim([0
 Tb*N]); ylim([-0.5 1.5]);
xlabel('Time(sec)'); ylabel('Amplitude(volt)'); title('Input Message
 Signal');

figure();
plot(t_mod,X_QPSK,'LineWidth',1, 'Color','Black'); grid on;
xlabel('Time(sec)'); ylabel('Amplitude(volt)'); title('QPSK Modulated
 Signal');

figure();
plot(t_mod,Y,'Color','red'); grid on;
xlabel('Time(sec)'); ylabel('Amplitude(volt)'); title('Noisy Signal');
```


Input Message Signal
Amplitude(volt)
Time(sec)
1.5
1
0.5
0
-0.5
0
1
2
3
4
5
6

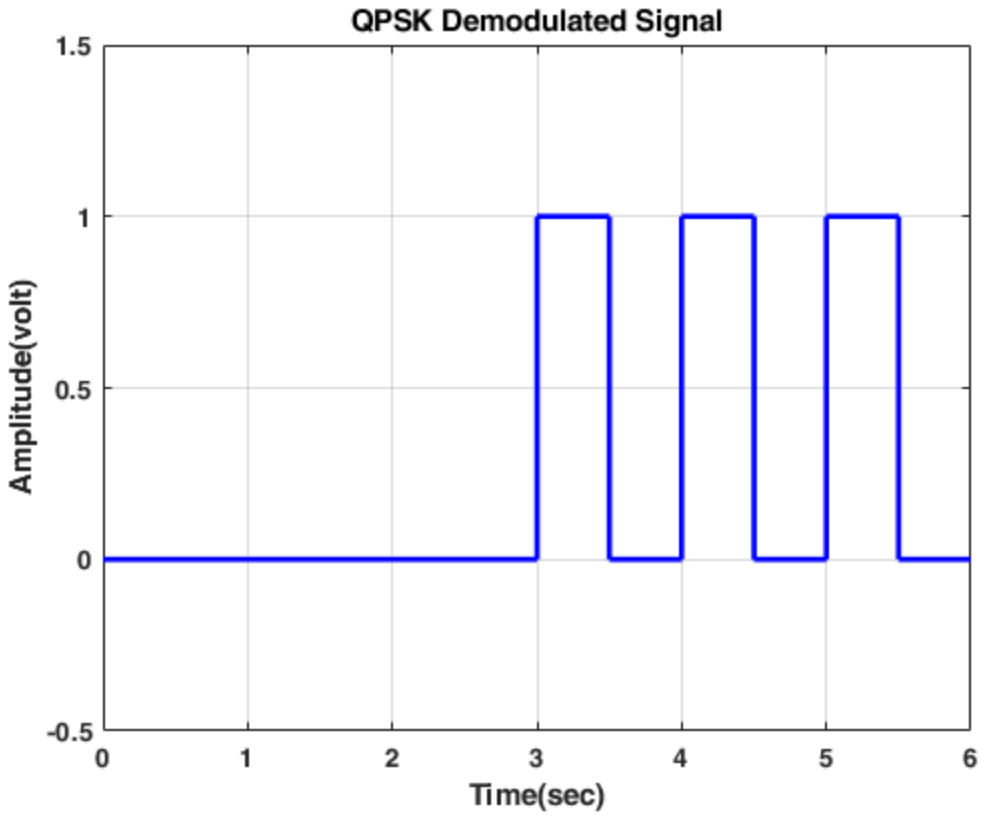

QPSK Modulated Signal
Amplitude(volt)
Time(sec)
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
0
1
2
3
4
5
6



QPSK Demodulation

```
X_demod=[];

t1_dem = Tb/nb:Tb/nb:2*Tb;                  %Time of one symbol
Ac_dem = sqrt(1/Tb);                       %Carrier Amplitude
Q1_dem = Ac_dem*cos(2*pi*fc*t1_dem);     %Phi_1 Basis function
Q2_dem = Ac_dem*sin(2*pi*fc*t1_dem);     %Phi_2 Basis function
x1_con_demQ = [];
x2_con_demQ = [];
for i=2*nb:2*nb:length(Y)
  %Correlator
  x1_temp = Q1_dem.*Y((i-(2*nb-1)):i);     %In Phase Channel
  x2_temp = Q2_dem.*Y((i-(2*nb-1)):i);     %Quadrature Channel
  x1_corr=trapz(t1_dem,x1_temp);           %Integrate the In Phase
 Component over the time period of the symbol
  x2_corr=trapz(t1_dem,x2_temp);           %Integrate the Quadrature
 Component over the time period of the symbol

  x1_con_demQ = [x1_con_demQ x1_corr];
  x2_con_demQ = [x2_con_demQ x2_corr];

  %Decision Making Device
  if(x1_corr>0) %If the value > threshold (0)
    S1=1;       %Then symbol = 1
```

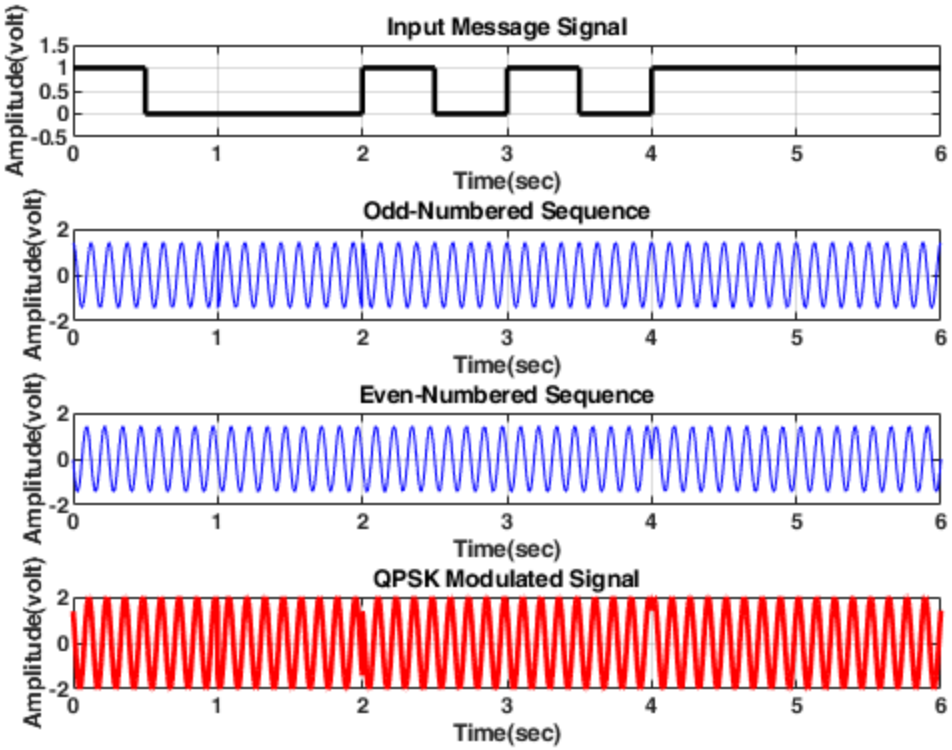
```
  else          %If the value < threshold (0)
    S1=0;        %Then symbol = 0
  end
  if(x2_corr>0) %If the value > threshold (0)
    S2=1;        %Then symbol = 1
  else          %If the value < threshold (0)
    S2=0;        %Then symbol = 0
  end
  X_demod=[X_demod S1 S2];
end


%Convert the symbols into continuous digital signal
X_dem_sig = [];
for i=1:length(X_demod)
    if X_demod(i)==1
       x_temp_dem=ones(1,nb);
    else
        x_temp_dem=zeros(1,nb);
    end
    X_dem_sig=[X_dem_sig x_temp_dem];
end
t_sig_dem = Tb/nb : Tb/nb : length(X_demod)*Tb; %Time vector of
 continuous digital signal
%Plotting the demodulated signal
figure();
plot(t_sig_dem,X_dem_sig, 'LineWidth',2,'Color','blue'); grid on;
 xlim([0 Tb*length(X_demod)]); ylim([-0.5 1.5]);
xlabel('Time(sec)'); ylabel('Amplitude(volt)'); title('QPSK
 Demodulated Signal');
```

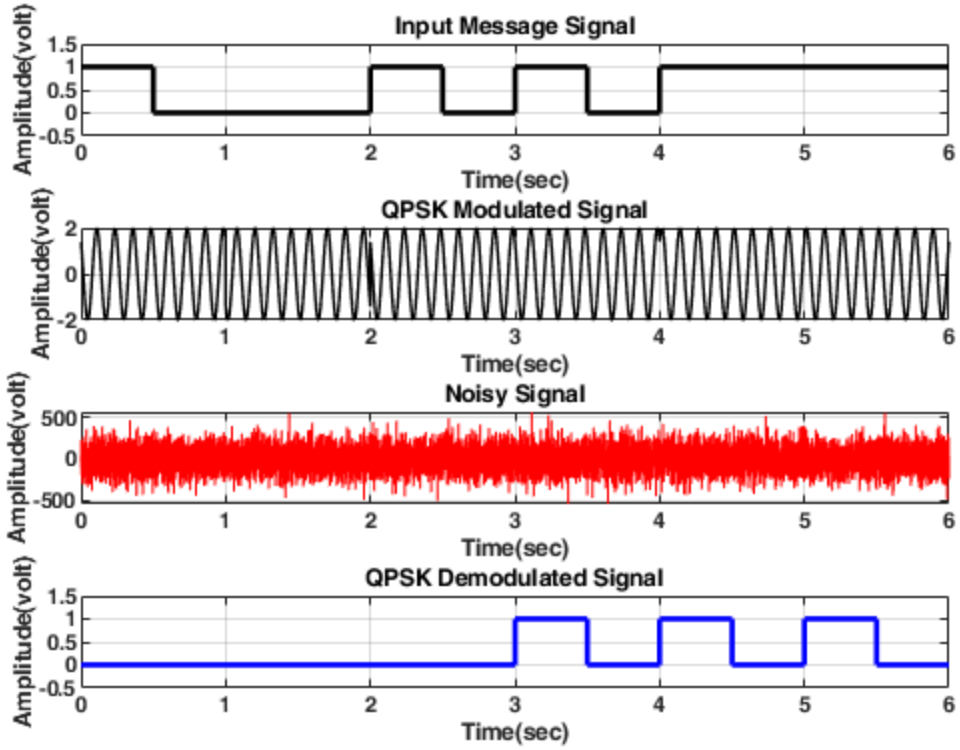


Graphs in Subplot (BPSK)

```
figure();
subplot(4,1,1);
plot(t_sig,X_digit, 'LineWidth',2,'Color','black'); grid on; xlim([0
 Tb*N]); ylim([-0.5 1.5]);
xlabel('Time(sec)'); ylabel('Amplitude(volt)'); title('Input Message
 Signal');

%Plotting the BPSK Signal
subplot(4,1,2);
plot(t_mod,Odd_Sig,'Color','blue'); grid on;
xlabel('Time(sec)'); ylabel('Amplitude(volt)'); title('Odd-Numbered
 Sequence');
subplot(4,1,3);
plot(t_mod,Even_Sig,'Color','blue'); grid on;
xlabel('Time(sec)'); ylabel('Amplitude(volt)'); title('Even-Numbered
 Sequence');
subplot(4,1,4);
plot(t_mod,X_QPSK,'LineWidth',2, 'Color','red'); grid on;
xlabel('Time(sec)'); ylabel('Amplitude(volt)'); title('QPSK Modulated
 Signal');


%Plotting the Modulated VS. Noisy Signal
```

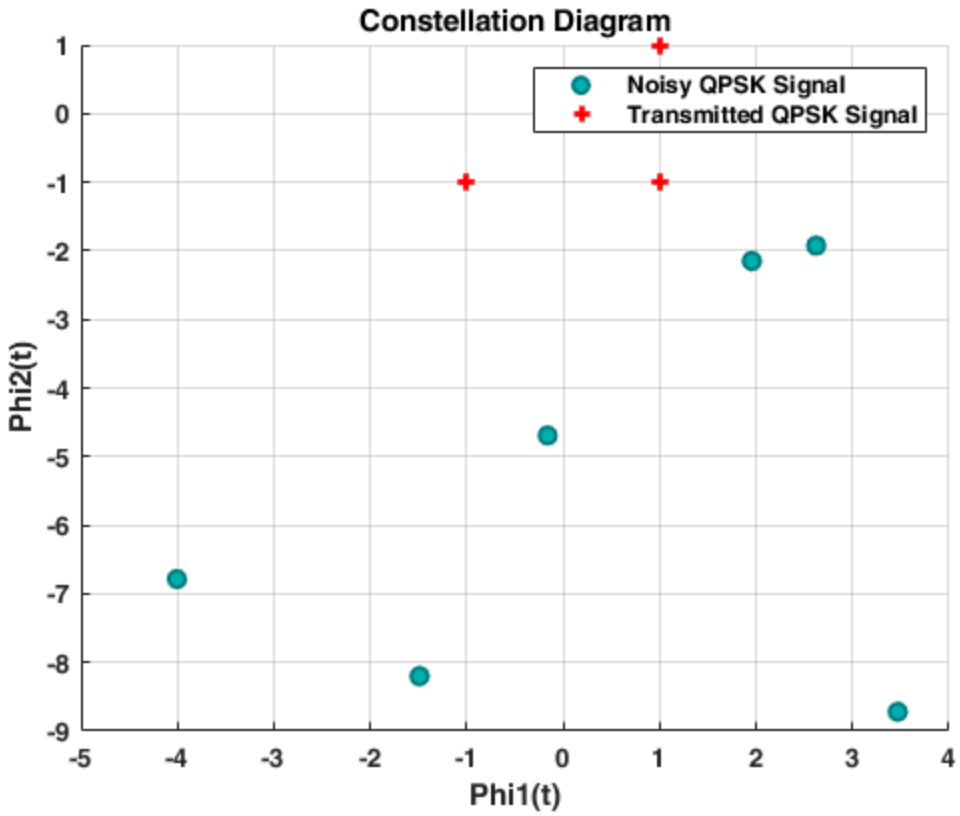
```
figure();
subplot(4,1,1);
plot(t_sig,X_digit, 'LineWidth',2,'Color','black'); grid on; xlim([0
 Tb*N]); ylim([-0.5 1.5]);
xlabel('Time(sec)'); ylabel('Amplitude(volt)'); title('Input Message
 Signal');

subplot(4,1,2);
plot(t_mod,X_QPSK,'LineWidth',1, 'Color','Black'); grid on;
xlabel('Time(sec)'); ylabel('Amplitude(volt)'); title('QPSK Modulated
 Signal');

subplot(4,1,3);
plot(t_mod,Y,'Color','red'); grid on;
xlabel('Time(sec)'); ylabel('Amplitude(volt)'); title('Noisy Signal');

subplot(4,1,4);
plot(t_sig_dem,X_dem_sig, 'LineWidth',2,'Color','blue'); grid on;
 xlim([0 Tb*length(X_demod)]); ylim([-0.5 1.5]);
xlabel('Time(sec)'); ylabel('Amplitude(volt)'); title('QPSK
 Demodulated Signal');
```





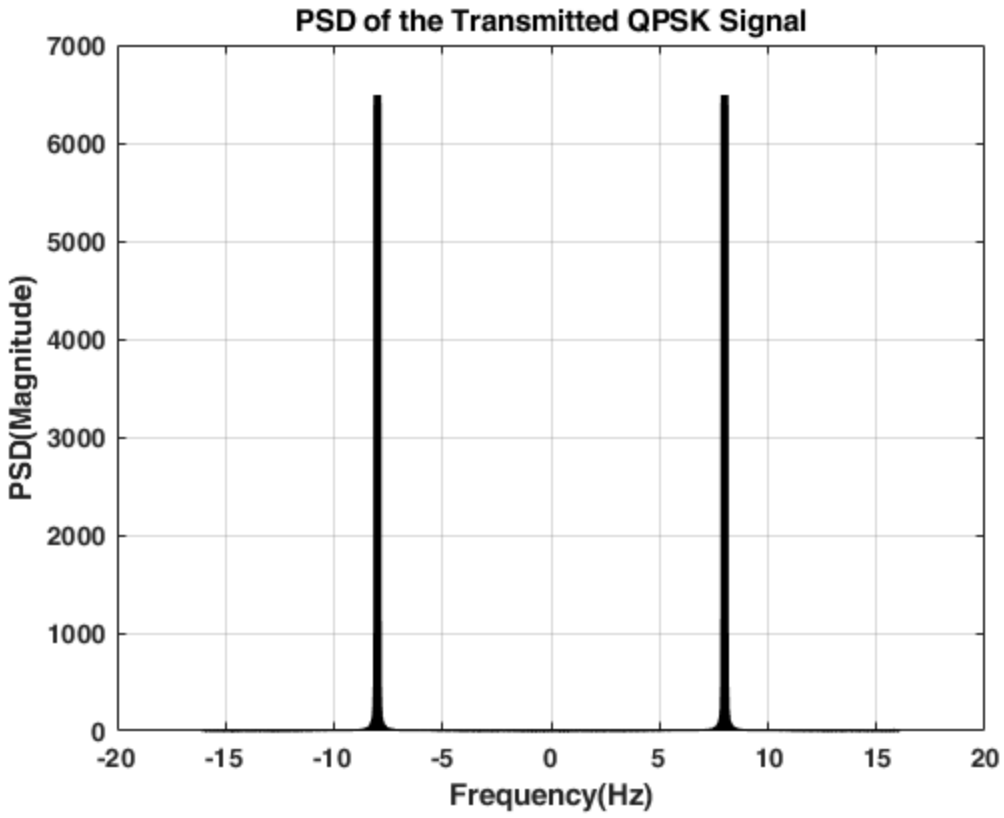
Constellation Diagram (QPSK)

```
figure();
scatter(x1_con_demQ, x2_con_demQ, 40, 'MarkerEdgeColor',
 [0 .5 .5],'MarkerFaceColor',[0 .7 .7], 'LineWidth',1.5); grid on;
hold on;
scatter(x1_con_mod, x2_con_mod, 40, 'red', '+', 'LineWidth',2);
hold off
xlabel('Phi1(t)'); ylabel('Phi2(t)'); title('Constellation Diagram');
 legend('Noisy QPSK Signal','Transmitted QPSK Signal')
```



PSD (QPSK)

```
psd_QPSK = fft(X_QPSK);  %FFT for the Modulated Signal
psds_QPSK = fftshift(psd_QPSK);  %FFT Shift
psd_neg = flip(psds_QPSK);   %The negative side frequencies
psdQ = [psd_neg psds_QPSK];   %Combining the frequencies of both sides
f = linspace(-2*fc, 2*fc, length(psdQ));  %The frequency Vector
%Plotting PSD
figure();
plot(f, abs(psdQ), 'LineWidth',2,'Color','black'); grid on;
xlabel('Frequency(Hz)'); ylabel('PSD(Magnitude)'); title('PSD of the
 Transmitted QPSK Signal');
```



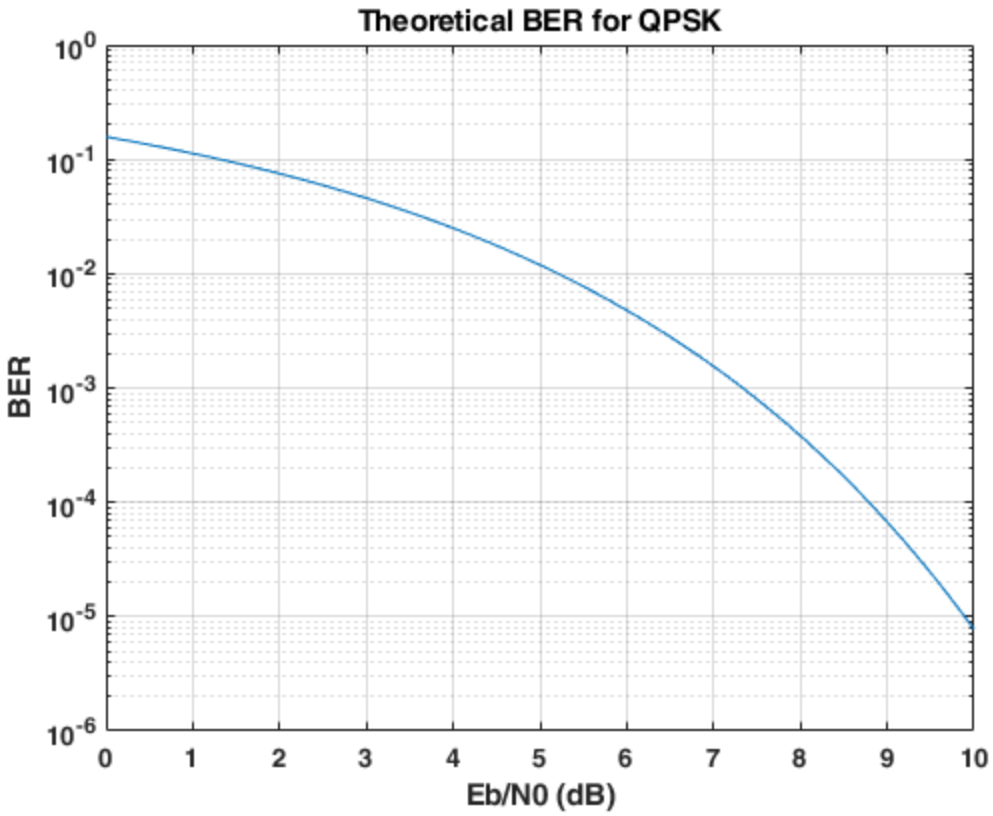
BER (QPSK)

```
Bit_Error_Rate = biterr(X_demod, X_input)
```

```
Bit_Error_Rate =

     4
```

Theoretical BER (QPSK)

```
clc;
clear;
Eb_N0_dB = 0:0.1:10;
Eb_N0 = 10.^(Eb_N0_dB/10);
x= sqrt(Eb_N0);
BER = erfc(x);
figure();
semilogy(Eb_N0_dB,BER); grid on; ylabel('BER'); xlabel('Eb/N0 (dB)');
 title('Theoretical BER for QPSK');
```



Published with MATLAB® R2021a