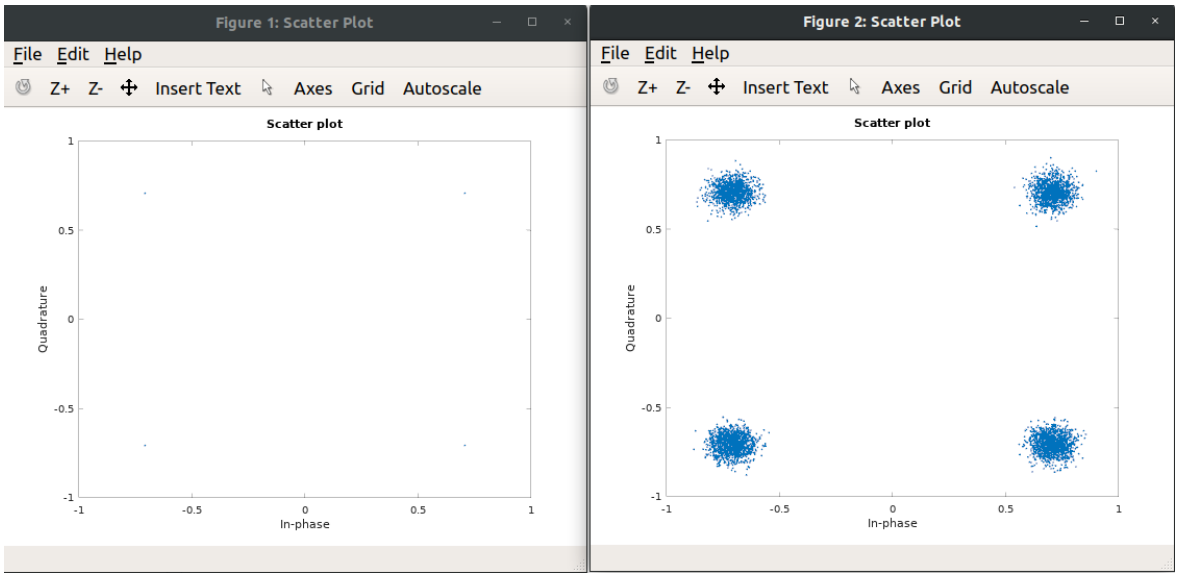
```
%                    <<Experiment-4 PART-A (QPSK)>>
%                        << Objective-1 >>


% Name: Rathod Chittaranjan
% Roll No:32457


% Aim: Simulation study of Performance of QPSK.
% Objective-1: Write a program to plot signal constellation diagram of received
%             QPSK signal in the presence of AWGN.
% Objective-2: Write a program to plot Practical and Theoretical BER vs SNR graph
%             of received QPSK signal in the presence of AWGN for ML receiver.

% Note: For objective-2, see separate octave files named <my_QPSK_ber_method1.m>
%       and <my_QPSK_ber_method2.m>



clc;
clear all;
close all;
pkg load communications
N = 10000; % Number of bits to be transmitted using QPSK
          % Too large value may slow down the program
x = randi([0,1],1,N); % Random input bits generation
M = 4;     % Number of Symbols in QPSK

% Symbol Generation
yy = [];
for i=1:2:length(x)
  if x(i)==0 && x(i+1)==0
    y = cosd(225)+1j*sind(225);
  elseif x(i)==0 && x(i+1)==1
    y = cosd(135)+1j*sind(135);
  elseif x(i)==1 && x(i+1)==0
    y = cosd(315)+1j*sind(315);
  elseif x(i)==1 && x(i+1)==1
    y = cosd(45)+1j*sind(45);
  endif
% Transmitted Symbols
yy = [yy y];
endfor
scatterplot(yy);    % Constellation Diagram without Noise
EbN0db = 20; % Change this value & run program to see the noisy constellation.
EbN0 = 10^(EbN0db/10);

% AWGN Channel
n = (1/sqrt(2))*[randn(1,length(yy)) + 1j*randn(1,length(yy))];
sigma = sqrt(1/((log2(M))*EbN0));

% Received Symbols
```

```
r = yy + sigma*n;
scatterplot(r);  % Constellation Diagram with Noise
```

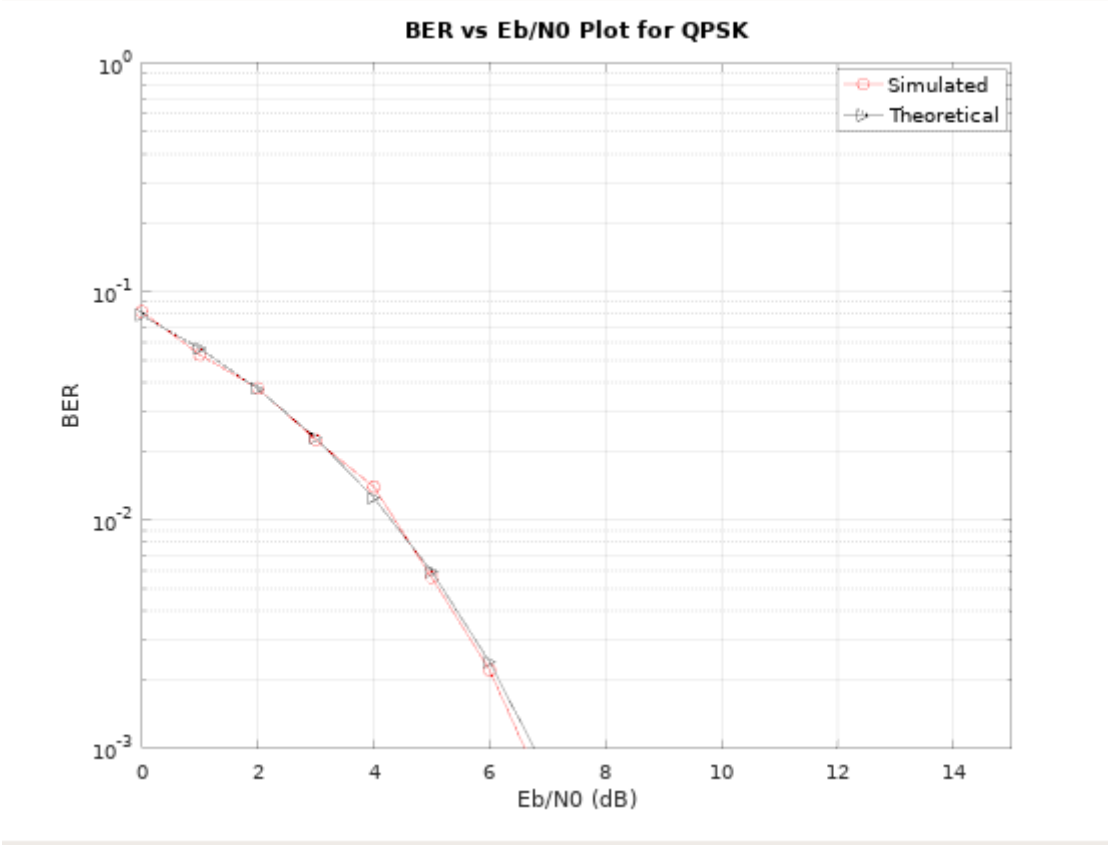
OUTPUT



```
%                    <<Experiment-4 PART-A (QPSK)>>
%                     << Objective-2 (Method-1) >>
% Name: Rathod Chittaranjan
% Roll No:32457
% Aim: Simulation study of Performance of QPSK.
% Objective-1: Write a program to plot signal constellation diagram of received
%           QPSK signal in the presence of AWGN.
% Objective-2: Write a program to plot Practical and Theoretical BER vs SNR graph
%           of received QPSK signal in the presence of AWGN for ML receiver.
% Note: For objective-1, see separate octave file named <my_QPSK_constellation.m>
%       and alternative method for objective-2 see file <my_QPSK_ber_method2.m>.
clc;c
lear all;
close all;
pkg load communications
N = 10000;
% Number of bits to be transmitted using QPSK
% Too large value may slow down the program
x = randi([0,1],1,N);
% Random input bits generation
M = 4;
% Number of Symbols in QPSK
% Symbol Generation
yy = [];
for i=1:2:length(x)
if x(i)==0 && x(i+1)==0
y = cosd(225)+1j*sind(225);
elseif x(i)==0 && x(i+1)==1
  y = cosd(135)+1j*sind(135);
 elseif x(i)==1 && x(i+1)==0
 y = cosd(315)+1j*sind(315);
elseif x(i)==1 && x(i+1)==1
y = cosd(45)+1j*sind(45);
endif
% Transmitted Symbolsyy = [yy y];
endfor
% Detection Method-1
ber_simulated = [];
ber_theoretical = [];
for EbN0db = 0:15
EbN0 = 10^(EbN0db/10);
n = (1/sqrt(2))*[randn(1,length(yy)) + 1j*randn(1,length(yy))];
 % AWGN
sigma = sqrt(1/((log2(M))*EbN0));
% Symbol energy is normalized to Unity
r = yy + sigma*n;
% Received symbols
I = (real(r)>0);
% In-phase component detection
Q = (imag(r)>0);
```

```
% Q-phase component detection
% Estimation of Bits
x_estimated = [];
for i=1:length(r)
x_estimated = [x_estimated I(i) Q(i)];
endfor
% BER Computation
ber_simulated =[ber_simulated sum(x~=x_estimated)/N];
ber_theoretical = [ber_theoretical 0.5*erfc(sqrt(EbN0))];
endfor
% BER Plotting
EbN0db = 0:15;
semilogy(EbN0db, ber_simulated, 'ro-', EbN0db, ber_theoretical, 'k>-');
title('BER vs Eb/N0 Plot for QPSK');
xlabel('Eb/N0 (dB)');
ylabel('BER');
grid on;
legend('Simulated', 'Theoretical');
axis([0 15 10^-3 10^0]);
```

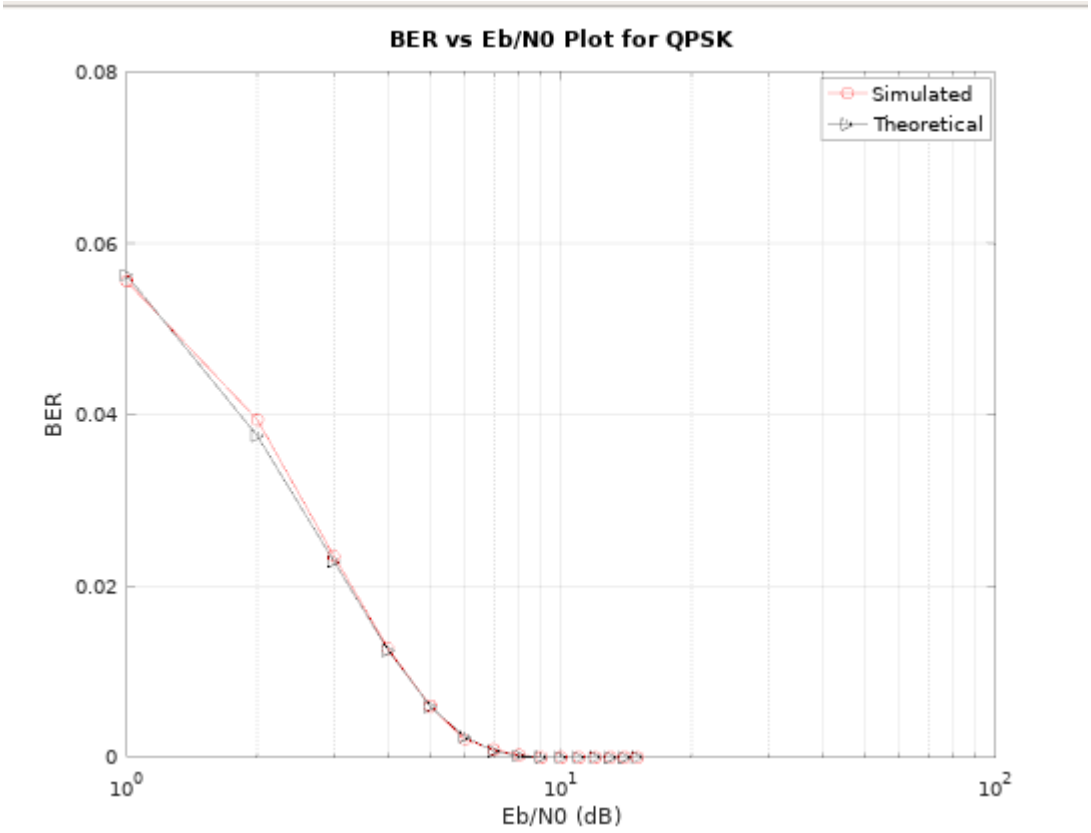
OUTPUT



```
%                    <<Experiment-4 PART-A (QPSK)>>
%                     << Objective-2 (Method-2) >>
% Name: Rathod Chittaranjan
% Roll No:32457

% Aim: Simulation study of Performance of QPSK.
% Objective-1: Write a program to plot signal constellation diagram of received
%           QPSK signal in the presence of AWGN.
% Objective-2: Write a program to plot Practical and Theoretical BER vs SNR graph
%           of received QPSK signal in the presence of AWGN for ML receiver.

% Note: For objective-1, see separate octave file named <my_QPSK_constellation.m>
%       and alternative method for objective-2 see file <my_QPSK_ber_method1.m>.




clc;
clear all;
close all;
pkg load communications
N = 10000; % Number of bits to be transmitted using QPSK
         % Too large value may slow down the program
x = randi([0,1],1,N); % Random input bits generation
M = 4;     % Number of Symbols in QPSK

% Symbol Generation
yy = [];
for i=1:2:length(x)
  if x(i)==0 && x(i+1)==0
    y = cosd(225)+1j*sind(225);
  elseif x(i)==0 && x(i+1)==1
    y = cosd(135)+1j*sind(135);
  elseif x(i)==1 && x(i+1)==0
    y = cosd(315)+1j*sind(315);
  elseif x(i)==1 && x(i+1)==1
    y = cosd(45)+1j*sind(45);
  endif
% Transmitted symbols
yy = [yy y];
endfor

% Detection based on euclidean distance
ber_simulated = [];
ber_theoretical = [];
ref_symbols = [cosd(45)+1j*sind(45), cosd(135)+1j*sind(135), cosd(225)+1j*sind(225),
cosd(315)+1j*sind(315)];
for EbN0db = 0:15
EbN0 = 10^(EbN0db/10);
n = (1/sqrt(2))*[randn(1,length(yy)) + 1j*randn(1,length(yy))]; % AWGN
sigma = sqrt(1/((log2(M))*EbN0)); % Symbol energy is normalized to Unity
```

```
r = yy + sigma*n;   % Received symbols

% Calculation of Euclidean Distances of received symbols from reference symobols
min_dist_index = [];
for i=1:length(r)
  Dist = [];
  for k=1:length(ref_symbols)
  dist = sqrt((real(r(i))-real(ref_symbols(k)))^2 + (imag(r(i))-imag(ref_symbols(k)))^2);
  Dist = [Dist dist];
  endfor
  min_dist_index = [min_dist_index find(Dist==min(Dist))];
endfor

% Estimation of Bits
x_estimated = [];
for i=1:length(r)
  if ref_symbols(min_dist_index(i))== cosd(45)+1j*sind(45);
  x_estimated = [x_estimated 1 1];
  elseif ref_symbols(min_dist_index(i))== cosd(135)+1j*sind(135);
  x_estimated = [x_estimated 0 1];
  elseif ref_symbols(min_dist_index(i))== cosd(225)+1j*sind(225);
  x_estimated = [x_estimated 0 0];
  elseif ref_symbols(min_dist_index(i))== cosd(315)+1j*sind(315);
  x_estimated = [x_estimated 1 0];
  endif
endfor

% BER Computation
ber_simulated =[ber_simulated sum(x~=x_estimated)/N];
ber_theoretical = [ber_theoretical 0.5*erfc(sqrt(EbN0))];
endfor

% BER Plotting
EbN0db = 0:15;
semilogx(EbN0db, ber_simulated, 'ro-', EbN0db, ber_theoretical, 'k>-');
title('BER vs Eb/N0 Plot for QPSK');
xlabel('Eb/N0 (dB)');
ylabel('BER');
grid on;
legend('Simulated', 'Theoretical');
axis([0 15 10^-3 10^0]);
```

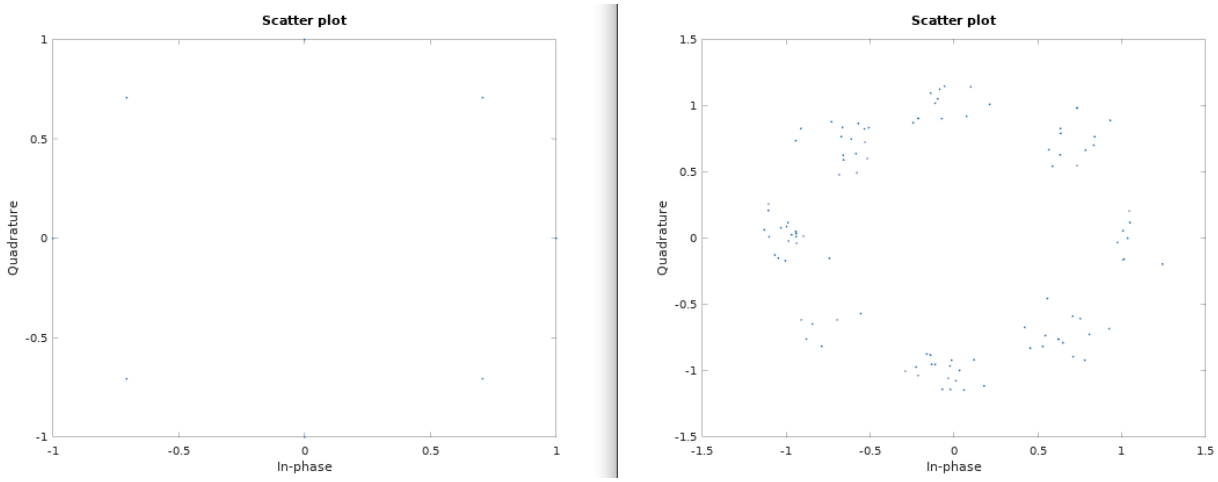
OUTPUT :


BER vs Eb/N0 Plot for QPSK
Simulated
Theoretical
BER
Eb/N0 (dB)
0.08
0.06
0.04
0.02
0
10^0
10^1
10^2

```
%                        Experiment-5
%                    << PART-B (8-PSK)>>


% Name: Rathod Chittaranjan
% Roll No:32457

% << Objective-1 >>


% Aim: Simulation study of Performance of 8-PSK.
% Objective-1: Write a program to plot signal constellation
diagram of received % 8-PSK

% signal in the presence of AWGN.
% Objective-2: Write a program to plot Practical and Theoretical
BER vs SNR graph
% of received 8-PSK signal in the presence of AWGN for ML
receiver.



clc;
clear all;
close all;
pkg load communications
N = 300; % Number of bits to be transmitted using *-PSK
% Too large value may slow down the program
x = randi([0,1],1,N); % Random input bits generation
M = 8; % Number of Symbols in 8-PSK
% Symbol Generation
yy = [];
for i=1:3:length(x)
if x(i)==0 && x(i+1)==0 & x(i+2)==0
y = cosd(0)+1j*sind(0);
elseif x(i)==0 && x(i+1)==0 & x(i+2)==1
y = cosd(45)+1j*sind(45);
elseif x(i)==0 && x(i+1)==1 & x(i+2)==1
y = cosd(90)+1j*sind(90);
elseif x(i)==0 && x(i+1)==1 & x(i+2)==0
y = cosd(135)+1j*sind(135);
elseif x(i)==1 && x(i+1)==1 & x(i+2)==0
y = cosd(180)+1j*sind(180);
elseif x(i)==1 && x(i+1)==1 & x(i+2)==1
y = cosd(225)+1j*sind(225);
elseif x(i)==1 && x(i+1)==0 & x(i+2)==1
y = cosd(270)+1j*sind(270);
elseif x(i)==1 && x(i+1)==0 & x(i+2)==0
y = cosd(315)+1j*sind(315);
endif % Transmitted Symbols
```

```
yy = [yy y];
endfor
scatterplot(yy); % Constellation Diagram without Noise
EbN0db = 10; % Change this value & run program to see the noisy
constellation.
EbN0 = 10^(EbN0db/10); % AWGN Channel
n = (1/sqrt(2))*[randn(1,length(yy)) + 1j*randn(1,length(yy))];
sigma = sqrt(1/((log2(M))*EbN0));
% Received Symbols
r = yy + sigma*n;
scatterplot(r);
```

OUTPUT



Conclusion:

• M-ary PSK are, as with BPSK, there are phase ambiguity problems at the receiver and differentially encoded M-ary PSK is more normally used in practice. It is use of differential M-ary PSK where the phase shift is not relation to a reference signal but only compares multiple signal to rebuild data.

This program is used to calculate the Bit Error Rate (BER) of QPSK in an Additive

White Gaussian Noise (AWGN) channel. The modulation and demodulation is done at

baseband. Complex numbers are used to model the in-phase and quadrature components

of a QPSK signal.

Quadrature Phase Shift Keying (QPSK) is a form of Phase Shift Keying in which two bits are modulated at once, selecting one of four possible carrier phase shifts (0, 90, 180, or 270 degrees). QPSK allows the signal to carry twice as much information as ordinary PSK using the same bandwidth.

In QPSK, the carrier phase can change only once every 2T seconds. If from one 2T interval to the next, neither bit stream changes sign, the carrier phase remains the same. If one component a(t) or ag(t) changes sign, a phase shift of x/2 occurs

8PSK have more than eight phases and it transmit 3 bits per symbol.