```
clc;
clear all;
close all;
%---------Input Fields------------------------
nSym=1500;%Number of symbols to transmit
EbN0dB = -10:1:10; % Eb/N0 range in dB for simulation

BER_sim = zeros(1,length(EbN0dB));%simulated Symbol error rates

M=2; %number of constellation points in BPSK
m = [0,1];%all possible input bits
A = 1; %amplitude
constellation = A*cos(m/M*2*pi);%constellation points

d=floor(M.*rand(1,nSym));%uniform random symbols from 1:M
s=constellation(d+1);%BPSK modulated symbols

for i=1:length(EbN0dB)
    r  = add_awgn_noise(s,EbN0dB(i));%add AWGN noise
    dCap = (r<=0);%threshold detector
    BER_sim(i) = sum((d~=dCap))/nSym;%SER computation
end

semilogy(EbN0dB,BER_sim,'o','linewidth',2.5);
grid on;
hold on;
SNR=10.^(EbN0dB/10);
BER_th=(1/2)*erfc(sqrt(SNR));
semilogy(EbN0dB,BER_th,'r','linewidth',2);
title(' curve for Bit Error Rate verses  SNR for Binary PSK modulation');
xlabel(' SNR(dB)');
ylabel('BER');
legend('simulation','theorytical')

%%function to generate additive white gaussian noise%%
function [r,n,N0] = add_awgn_noise(s,SNRdB,L)
 s_temp=s;
 if iscolumn(s), s=s.'; end %to return the result in same dim as 's'
 gamma = 10^(SNRdB/10); %SNR to linear scale

 if nargin==2, L=1; end %if third argument is not given, set it to 1

 if isvector(s)
  P=L*sum(abs(s).^2)/length(s);%Actual power in the vector
 else %for multi-dimensional signals like MFSK
  P=L*sum(sum(abs(s).^2))/length(s); %if s is a matrix [MxN]
 end

 N0=P/gamma; %Find the noise spectral density
 if(isreal(s))
  n = sqrt(N0/2)*randn(size(s));%computed noise
 else
  n = sqrt(N0/2)*(randn(size(s))+1i*randn(size(s)));%computed noise
```

```
 end

 r = s + n; %received signal

 if iscolumn(s_temp), r=r.'; end
 %return r in original format as s
end
```