**//LBC**

clc;

clear all;

k=input('Enter the length of msg word:');

n=input('Enter the length of codeword:');

p=input('Enter the parity matrix:');

G=[eye(k);p;];m=input('Enter the length of msg word:');

H=[eye(n-k) p']

dtable=syndtable(H)

R=input('Enter the received code word');

%S=R\*H'

S\_B=rem(R\*H' ,2)

S\_D=bi2de(S\_B, 'left-msb')

if(S\_D==0)

disp('The recieved codeword is valid:')

else

disp('The corrected codeword is invalid:')

E=dtable(S\_D+1, :)

%CC=R+E

disp('The corrected codeword is: ')

cc=rem(R+E, 2)

msg=cc(1:k)

end

Number\_Symbols=**>** 16

**// M-ary PSK and M-ary QAM**

clc;

clear all;

M=input('Number\_Symbols=');

x=0:M-1;

N=1;

OFF=0;

z=pskmod(x,M);

figure(1)

scatterplot(z,N,OFF,"r+");

N=1;

OFF=0;

y=qammod(x,M);

figure(2)

scatterplot(y,N,OFF,"bo");

**//BPSK recerver**

clc;

close all;

data\_bits=10000;

b=(randn(1,data\_bits) > 5);

s=2\*b-1;

SNRdB=0:9;

for(k=1:length(SNRdB))

y=s+awgn(s,SNRdB(k));

error=0;

for(c=1:1:data\_bits)

if (y(c)>0&& s(c)==-1)||(y(c)<0&&s(c)==1)

error=error+1;

end

end

BER(k)=error/data\_bits;

end

figure(1);

semilogy(SNRdB, BER, 'r', 'Linewidth', 2);

grid on;

hold on;

SNR=10.^(SNRdB/10);

BER\_thBPSK=(1/2)\*erfc(sqrt(SNR));

semilogy(SNRdB,BER\_thBPSK,'k', 'linewidth', 2);

BER\_thQPSK=erfc(sqrt(SNR));

semilogy (SNRdB, BER\_thQPSK, 'b', 'LineWidth',2);

legend('PR-SNR','BPSK','QPSK')

**//Cyclic**

clc;

clear all;

close all;

n=input('Enter the value of n: ');

k=input('Enter the value of k: ');

m=n-k;

G=cyclpoly(n,k,'max')

poly2sym(G)

d1=[1 0 0 0];

poly2sym(d1)

c1=poly2sym(d1)\*poly2sym(G)

d2=[0 1 0 0];

poly2sym(d2)

c2=poly2sym(d2)\*poly2sym(G)

d3=[0 0 1 0];

poly2sym(d3)

c3=poly2sym(d3)\*poly2sym(G)

d4=[0 0 0 1];

poly2sym(d4)

c4=poly2sym(d4)\*poly2sym(G)

s=[c1;c2;c3;c4]

d=[d1;d2;d3;d4]

c=d\*s

parmat=hammgen(m)

trt=syndtable(parmat)

recd=[0 1 0 1 0 0 0]

syndrome=rem(recd\*parmat',2)

syndrome\_de=bi2de(syndrome,'left-msb')

disp([syndrome,'left-msb'])

disp(['Syndrome=',num2str(syndrome\_de), '(decimal)',num2str(syndrome\_de),'(binary)'])

Error=trt(1+syndrome\_de,:)

corrrctedcode= rem(Error+recd,2)

recd=[1 1 0 1 1 0 1]

syndrome=rem(recd\*parmat',2)

syndrome\_de=bi2de(syndrome,'left-msb');

disp(['Syndrome=',num2str(syndrome\_de), '(decimal)',

num2str(syndrome\_de),'(binary)'])

Error=trt(1+syndrome\_de,:)

correctedcode=rem(Error+recd,2)

**//Random process**

%Case 1:

m = 0;

sd = 1;

x = -6:0.1:6;

y = normpdf(x, m, sd);

figure(1)

subplot(2, 3, 1);

plot(x, y);

axis([-6 6 0 0.5]);

title('CASE-1, mean=0, sd=1');

xlabel('x values --->');

ylabel('pdf --->');

grid on;

%Case 2:

m = -1;

sd = 1;

x = -6:0.1:6;

y = normpdf(x, m, sd);

figure(1)

subplot(2, 3, 2);

plot(x, y);

axis([-6 6 0 0.5]);

title('CASE-2, mean=-1, sd=1');

xlabel('x values --->');

ylabel('pdf --->');

grid on;

%Case 3:

m = 1;

sd = 1;

x = -6:0.1:6;

y = normpdf(x, m, sd);

figure(1)

subplot(2, 3, 3);

plot(x, y);

axis([-6 6 0 0.5]);

title('CASE-3, mean=1, sd=1');

xlabel('x values --->');

ylabel('pdf --->');

grid on;

// **Experiment 10-Huffman Coding**

clc;

clear all;

close all;

code\_length=0;

x=input('Enter number of symbols: ');

for m=1:x

symbols(m)=input('Enter the symbol number: ');

p(m)=input('Enter the probability: ');

end

Hx=0

for m=1:x

[dict,avglen]=huffmandict(symbols,p)

hcode=huffmanenco(m,dict)

dsig = huffmandeco(hcode,dict)

code\_length=length(hcode)

Hx=Hx+(p(m)\*(-log(p(m)))/(log(2)));

end

display(Hx);

Efficiency=(Hx/avglen)\*100

Disp(Efficiency)

**OUTPUT🡪**

Enter number of symbols: 6

Enter the symbol number: 1

Enter the probability: 0.3

Enter the symbol number: 2

Enter the probability: 0.25

Enter the symbol number: 3

Enter the probability: 0.2

Enter the symbol number: 4

Enter the probability: 0.12

Enter the symbol number: 5

Enter the probability: 0.05

Enter the symbol number: 6

Enter the probability: 0.08