% <<Experiment-6 (16-Square QAM)>>

% << Objective-1 >>

% Aim: Simulation study of Performance of 16-Square QAM.

% Objective-1:Write a program to plot signal constellation diagram of received

% 16-Square QAM signal in the presence of AWGN.

% Objective-2:Write a program to plot Practical and Theoretical BER vs SNR graph

% of received 16-Square QAM in the presence of AWGN for ML receiver.

% Note: For objective-2, see separate octave file named <my\_16QAM\_ber.m>

clc;

clear all;

close all;

pkg load communications

N = 16000; % Number of bits to be transmitted using 16-Square QAM

% Too large value may slow down the program

x = randi([0,1],1,N); % Random input bits generation

M = 16; % Number of Symbols in 16-Square QAM

d = sqrt(2/5); % Average symbol energy is normalised to unity

% Symbol Generation

yy = [];

for i=1:4:length(x)

if x(i)==0 && x(i+1)==0 & x(i+2)==0 & x(i+3)==0

y = -3\*d/2+j\*(-3\*d/2);

elseif x(i)==0 && x(i+1)==0 & x(i+2)==0 & x(i+3)==1

y = -3\*d/2+j\*(-d/2);

elseif x(i)==0 && x(i+1)==0 & x(i+2)==1 & x(i+3)==1

y = -3\*d/2+j\*(d/2);

elseif x(i)==0 && x(i+1)==0 & x(i+2)==1 & x(i+3)==0

y = -3\*d/2+j\*(3\*d/2);

elseif x(i)==0 && x(i+1)==1 & x(i+2)==0 & x(i+3)==0

y = -d/2+j\*(-3\*d/2);

elseif x(i)==0 && x(i+1)==1 & x(i+2)==0 & x(i+3)==1

y = -d/2+j\*(-d/2);

elseif x(i)==0 && x(i+1)==1 & x(i+2)==1 & x(i+3)==1

y = -d/2+j\*(d/2);

elseif x(i)==0 && x(i+1)==1 & x(i+2)==1 & x(i+3)==0

y = -d/2+j\*(3\*d/2);

elseif x(i)==1 && x(i+1)==1 & x(i+2)==0 & x(i+3)==0

y = d/2+j\*(-3\*d/2);

elseif x(i)==1 && x(i+1)==1 & x(i+2)==0 & x(i+3)==1

y = d/2+j\*(-d/2);

elseif x(i)==1 && x(i+1)==1 & x(i+2)==1 & x(i+3)==1

y = d/2+j\*(d/2);

elseif x(i)==1 && x(i+1)==1 & x(i+2)==1 & x(i+3)==0

y = d/2+j\*(3\*d/2);

elseif x(i)==1 && x(i+1)==0 & x(i+2)==0 & x(i+3)==0

y = 3\*d/2+j\*(-3\*d/2);

elseif x(i)==1 && x(i+1)==0 & x(i+2)==0 & x(i+3)==1

y = 3\*d/2+j\*(-d/2);

elseif x(i)==1 && x(i+1)==0 & x(i+2)==1 & x(i+3)==1

y = 3\*d/2+j\*(d/2);

elseif x(i)==1 && x(i+1)==0 & x(i+2)==1 & x(i+3)==0

y = 3\*d/2+j\*(3\*d/2);

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