EEE5502: FOUNDATION OF DIGITAL SIGNAL PROCESSING

PROJECT 2: OFDM USING MATLAB

NAME: APARNA HARIYANI

UFID: 69185846

EMAIL: ahariyani@ufl.edu

Report 2.a:

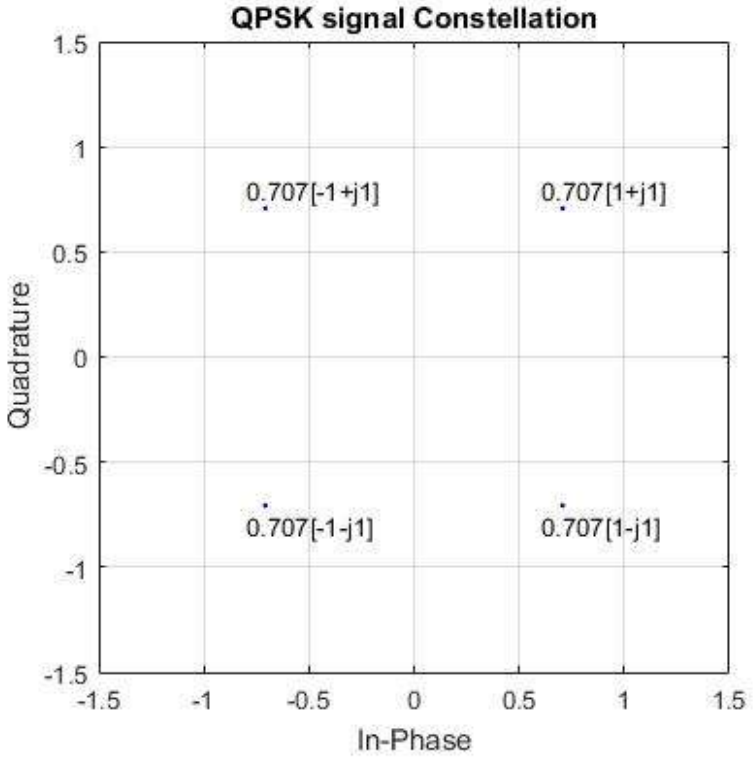

Fig 1. Input Signal generated using randi() that is given to QPSK modulator

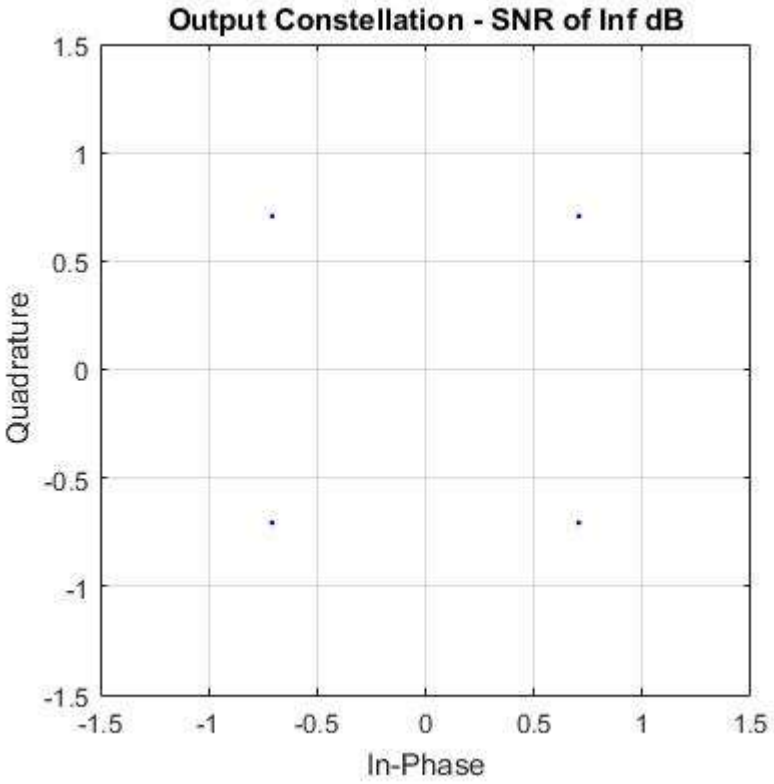

Fig 2. Output of OFDM modulator without any noise(AWGN infinit)

Report 2.b:

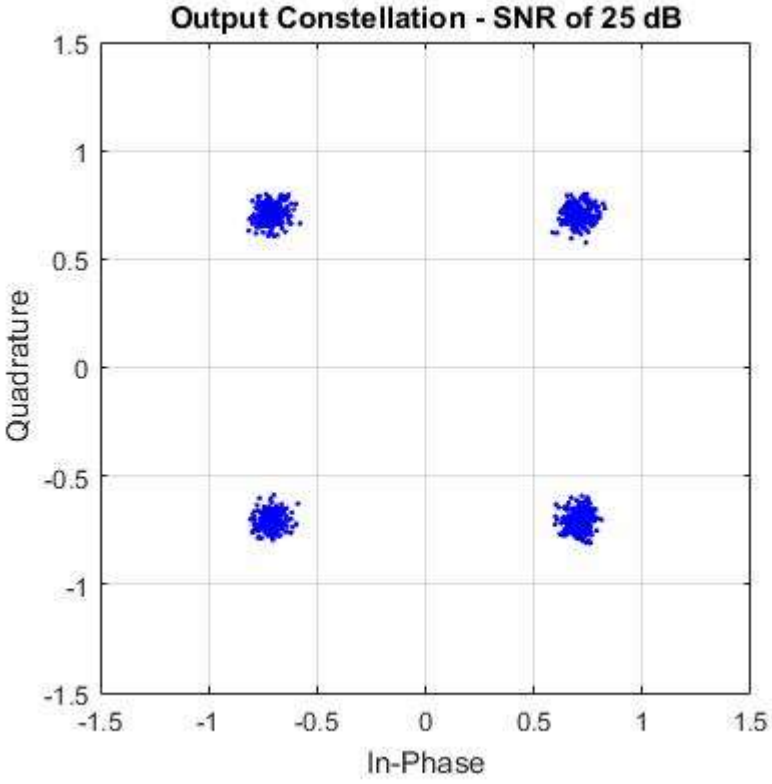

Fig 3. Output of OFDM modulator with AWGN 25dB

Report 2.c:

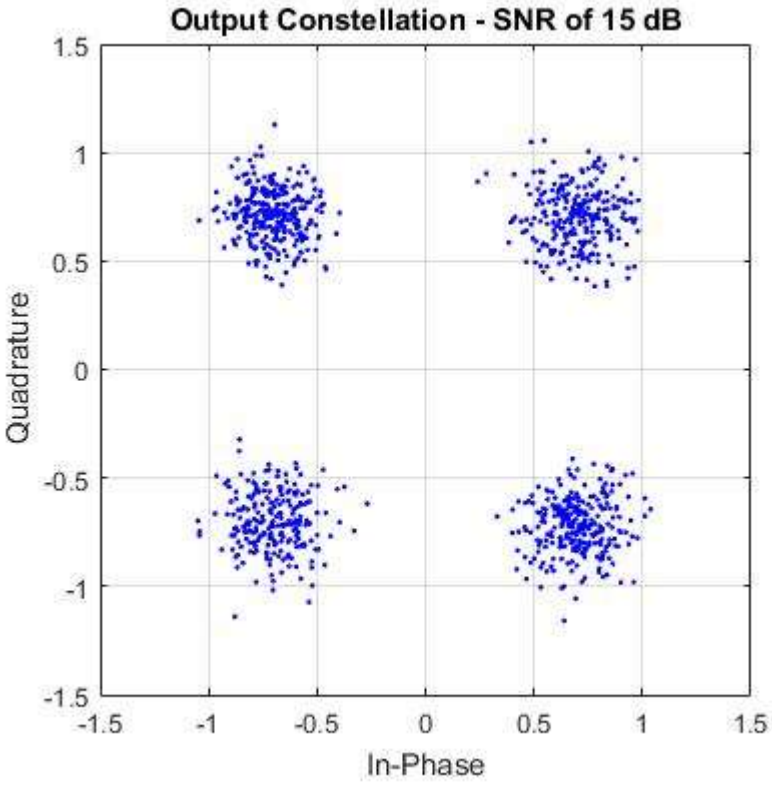

Fig 4. Output of OFDM modulator with AWGN 15dB

SER for AWGN added signal:

AWGN	AVERAGE SER	Error Variance
Infinite	0	2.8571e-32
25dB	0	0.0032
15dB	0	0.0330

We observe that as the noise is increased, the scattering of signal increases. Yet the Symbol Error rate is 0 because the signal is still reproducible. The reproducibility range is between 5dB and 45dB.

Matlab Code for Project 1

```
%Name: Aparna Hariyani
%UFID: 69185846
%Course: EEE 5502
%Project2 - OFDM

close all;
clear all;
clc;

%Signal generation
data = randi([0 3],16,64);
[row,col] = size(data);


%QPSK Modulation of Signal
qpskMod = comm.QPSKModulator;
dataMap = reshape(data,(row*col),1);
modData = qpskMod(dataMap);


scatterplot(modData)
axis([-1.5 1.5 -1.5 1.5])
title('QPSK signal Constellation')
text(0.600,0.800,'0.707[1+j1]');
text(-0.800,0.800,'0.707[-1+j1]');
text(-0.800,-0.800,'0.707[-1-j1]');
text(0.600,-0.800,'0.707[1-j1]');
grid on


modDataMap = reshape(modData,row,col);

%IFFT of QPSK modulated Data
for i=1:col
    in_ifft = modDataMap(:,i);
    ifftData = ifft(in_ifft, row);

    if i == 1
        ifftOut = ifftData;
    else
```

```
        ifftOut = [ifftOut,ifftData];
    end
end
%parallel to serial conversion
serialOut = reshape(ifftOut,1,(row*col));

tx_sig = ifftOut;

%rx_infdB = awgn(out_tx,inf,'measured');
rxSigInf = awgn(serialOut,inf,'measured');
rxSig25 = awgn(serialOut,25,'measured');
rxSig15 = awgn(serialOut,15,'measured');

rxSigInfMap = reshape(rxSigInf,row,col);
rxSig25Map = reshape(rxSig25,row,col);
rxSig15Map = reshape(rxSig15,row,col);

%FFT of Modulated data

for k = 1:col
    fftInInf = rxSigInfMap(:,k);
    fftIn25 = rxSig25Map(:,k);
    fftIn15 = rxSig15Map(:,k);

    fftDataInf = fft(fftInInf,row);
    fftData25 = fft(fftIn25,row);
    fftData15 = fft(fftIn15,row);

    if k == 1
        fftOutInf = fftDataInf;
        fftOut25 = fftData25;
        fftOut15 = fftData15;
    else
        fftOutInf = [fftOutInf,fftDataInf]; % Output of Receiver
        fftOut15 = [fftOut15,fftData15]; % Output of Receiver
        fftOut25 = [fftOut25,fftData25]; % Output of Receiver
    end
end

fftOutInfMap = reshape(fftOutInf,(row*col),1);
fftOut25Map = reshape(fftOut25,(row*col),1);
fftOut15Map = reshape(fftOut15,(row*col),1);

scatterplot(fftOutInfMap);
axis ([-1.5 1.5 -1.5 1.5]);
title('Output Constellation - SNR of Inf dB');
grid on;

scatterplot(fftOut15Map);
axis ([-1.5 1.5 -1.5 1.5]);
title('Output Constellation - SNR of 15 dB');
grid on;

scatterplot(fftOut25Map);
axis ([-1.5 1.5 -1.5 1.5]);
title('Output Constellation - SNR of 25 dB');
grid on;

%demodulation of QPSK data
qpskDemod= comm.QPSKDemodulator;
```

```
demodDataInf = qpskDemod(fftOutInfMap);
demodData25 = qpskDemod(fftOut25Map);
demodData15 = qpskDemod(fftOut15Map);

%SER calculation for infinite noise

errRate = comm.ErrorRate('ResetInputPort',true);
serInf = zeros(16,1);

for k = 1:16
    errors = errRate(dataMap,demodDataInf,1);
    serInf(k) = errors(1);
end

mean(serInf(1:16))

%SER calculation for 25dB noise


ser25 = zeros(16,1);

for k = 1:16
    errors = errRate(dataMap,demodData25,1);
    ser25(k) = errors(1);
end

mean(ser25(1:16))

%SER calculation for 15dB noise

ser15 = zeros(16,1);

for k = 1:16
    errors = errRate(dataMap,demodData15,1);
    ser15(k) = errors(1);
end

mean(ser15(1:16))

%Error Variance Calculation
vErrInf = var(modDataMap - fftOutInf);
mean(vErrInf)

vErr25 = var(modDataMap - fftOut25);
mean(vErr25)

vErr15 = var(modDataMap - fftOut15);
mean(vErr15)
```