Faculty of Engineering

Cairo University

# OFDM Project

Submitted to. Eng. Alaa Khairallah & prof.\Mohamed Khairy

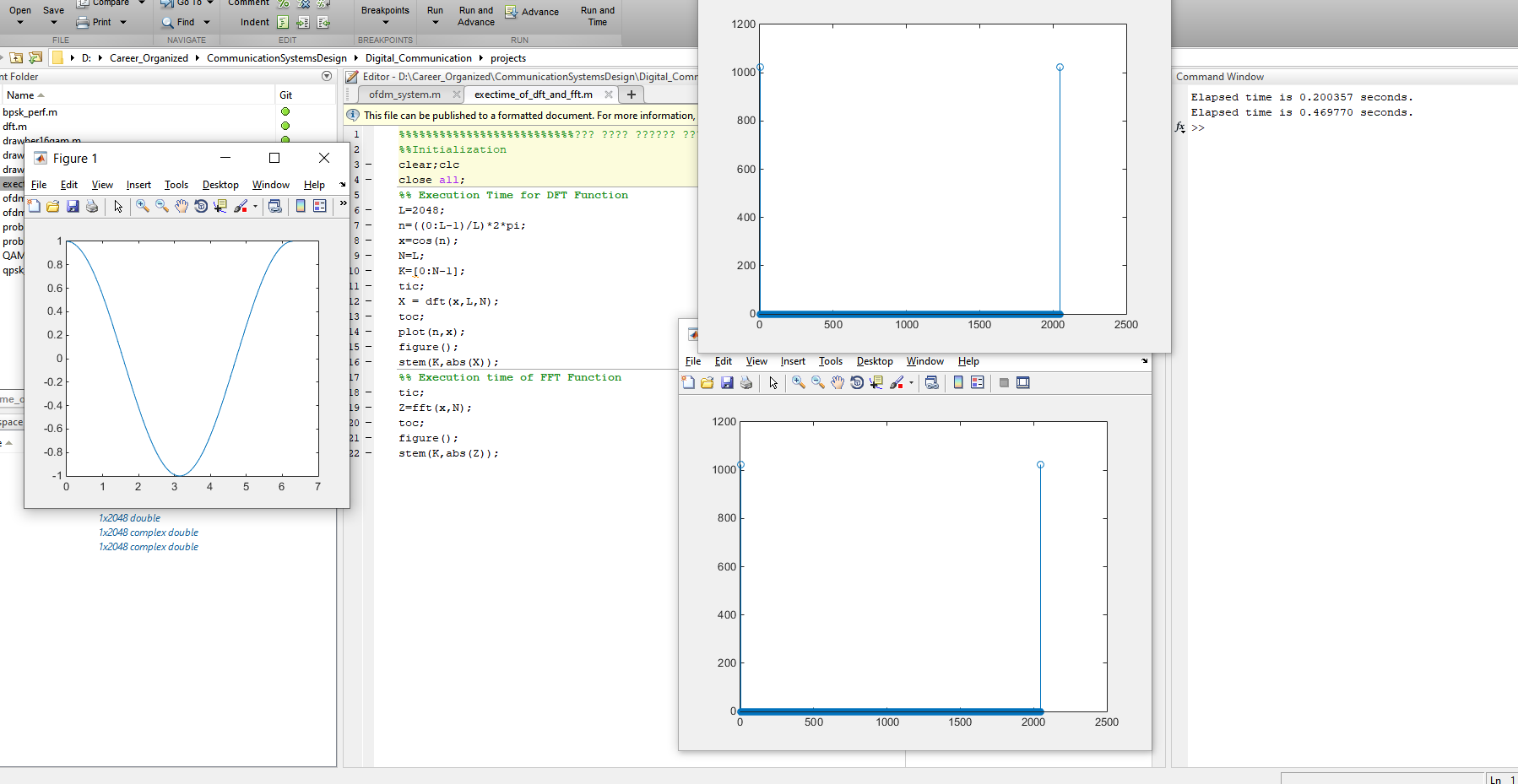
|  |  |  |
| --- | --- | --- |
| Name | Sec | BN |
| Hisham Mohamed Abdallah Naguib | 4 | 51 |

# Table of Contents:

# Problem 1:

Elapsed time for DFT: 0.469770 seconds

Elapsed time for FFT: 0.200357 seconds



# 

# Problem 2:

## **Transmitter Model:**

Generating BitStream and grey Encode them:

%% 16-QAM Transmitter Model

N=100; % Number of Samples

M=16; % Number bits per symbol

%-------------------Data Generation---------------------------%

data=randi([0 M-1],N,1);

%data=[0:M-1];

%-----------Grey Encoding------------%

[datagrey,mapgrey] = bin2gray(data,'qam',M);

Mapping Symbols to Constellations:

%-------------------16-QAM Modulation----------------------------%

consttable=[-3-3i, -3-1i, -3+3i, -3+1i,...

-1-3i, -1-1i, -1+3i, -1+1i,...

3-3i, 3-1i, 3+3i, 3+1i,...

1-3i, 1-1i, 1+3i, 1+1i];

for k=1:length(datagrey)

tx(k) = consttable(datagrey(k)+1);

end

tx=tx(:);

scatterplot(tx,1,0,'b\*');

for k = 1:16

text(real(tx(k))-0.3,imag(tx(k))+0.3,...

dec2base(data(k),2,4));

text(real(tx(k))-0.3,imag(tx(k))-0.3,...

dec2base(datagrey(k),2,4),'Color',[1 0 0]);

end

title('Transmitted Symbols');

## Fading Channel Model:

%% Rayleigh-Fading Channel Model

hr=normrnd(0,sqrt(0.5),1);

hi=normrnd(0,sqrt(0.5),1);

h=(hr+1i\*hi)\*ones(1,N);

h=h(:);

rx=tx.\*h;

scatterplot(rx)

title('Rayleigh-Fading Channel Effect');

## **AWGN Channel Model:**

%% AWGN Channel

%AWGN Noise

mu=0;

No=0.1;

variance=No/2;

sigma=sqrt(variance);

nc=normrnd(mu,sigma,[1,N]);

ns=normrnd(mu,sigma,[1,N]);

n=nc+1i\*ns;

n=n(:);

yk=rx+n;

scatterplot(yk);

title('AWGN Channel Effect');

## Equalizer:

%Equalizer assuming channel is known

yk=yk/h;

## Correlation and Decision Model:

%% Correlator & Decision Model

consttable=[-3-3i, -3-1i, -3+3i, -3+1i,...

-1-3i, -1-1i, -1+3i, -1+1i,...

3-3i, 3-1i, 3+3i, 3+1i,...

1-3i, 1-1i, 1+3i, 1+1i];

for N = 1:length(yk)

%compute the minimum distance for each symbol

[~, idx] = min(abs(yk(N) - consttable));

datademod(N) = idx-1;

end

[datademodbin,mapbin] = gray2bin(datademod,'qam',M);

datademodbin=datademodbin(:);

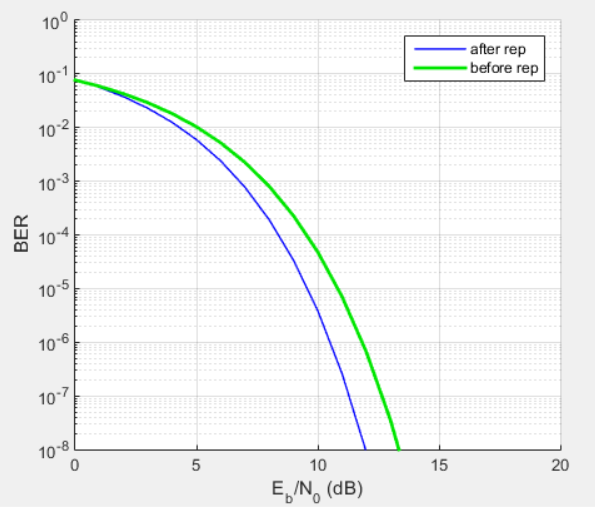
## BitError Rate Estimation:

%% BER

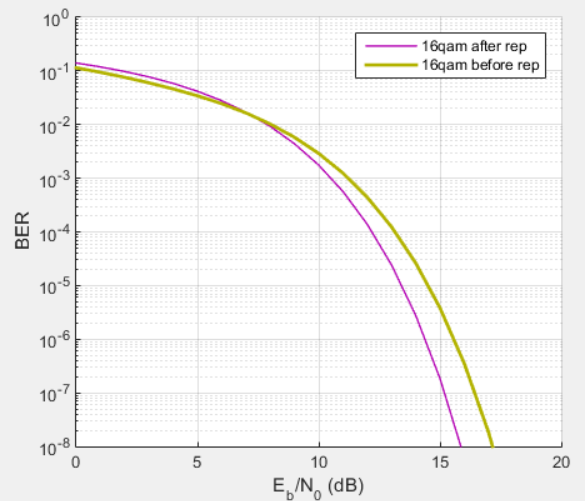
BER=biterr(datademodbin,data)/(length(data)\*log2(M));

ALL above procedures are same for different types of modulation we just change params like M for number of levels, constellation tables

**BPSK&QPSK BER Before & After Repetition:**



**16-QAM BER Before & After Repetition:**



# Problem 3:

## Coding

%% Generating and coding data

N=21;

data=randi([0 3],N,1)';

codeddata=repmat(data,1,3);

codeddata=[0, codeddata];

## Interleaver

interleavematrix=reshape(codeddata,8,8);

## Mapper: Same like question 2

## 32-point IFFT

ifft\_sig=ifft(y,32);

## Add Cyclic Extension

%% Add Cyclic extension

% Cyclic Prefixing

CP\_part=serial(:,end-Ncp+1:end); % this is the Cyclic Prefix part to be appended.

cp=[CP\_part serial];

## Channel

## Flat Channel

%% Rayleigh-Flat Fading Channel

hr=normrnd(0,sqrt(0.5),1);

hi=normrnd(0,sqrt(0.5),1);

h=(hr+1i\*hi)\*ones(1,N);

h=h(:);

## Selective Channel

% Selective Channel

selhr=normrnd(0,sqrt(0.5),N);

selhi=normrnd(0,sqrt(0.5),N);

selh=selhr+1i\*selhi;

selh=selh(:);

## AWGN Channel

%% AWGN Channel

ofdm\_sig=awgn(cext\_data,snr,'measured'); % Adding white Gaussian Noise

## Receiver

% Removing Cyclic Extension

for i=1:32

rxed\_sig(i)=ofdm\_sig(i+16);

end

% FFT

ff\_sig=fft(rxed\_sig,32);

% Demodulation

dem\_data= qpskdemod(ff\_sig);

% Decimal to binary conversion

bin=de2bi(dem\_data','left-msb');

bin=bin';

% De-Interleaving

deintlvddata = matdeintrlv(bin,2,2); % De-Interleave

deintlvddata=deintlvddata';

deintlvddata=deintlvddata(:)';

% Decoding data

n=6;

k=3;

decodedata =vitdec(deintlvddata,trellis,5,'trunc','hard'); % decoding datausing veterbi decoder

rxed\_data=decodedata;

% Calculating BER

rxed\_data=rxed\_data(:)';

errors=0;

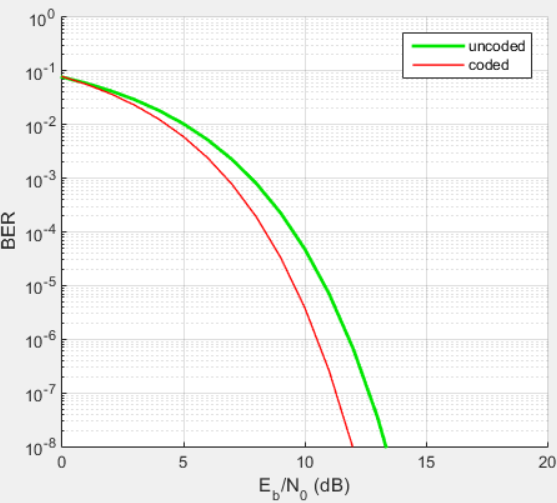
c=xor(data,rxed\_data);

errors=nnz(c);

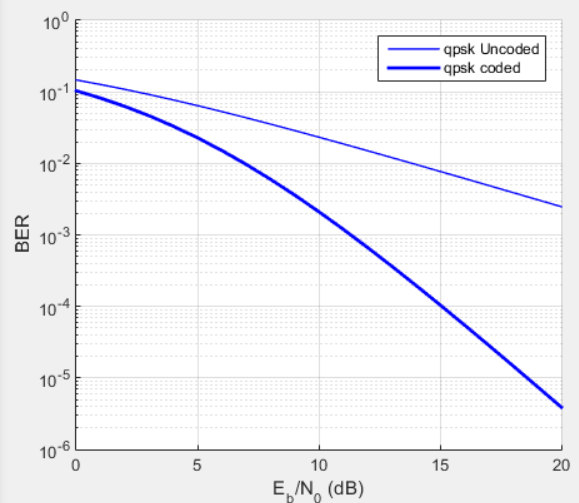
BER=errors/length(data);

## Results:

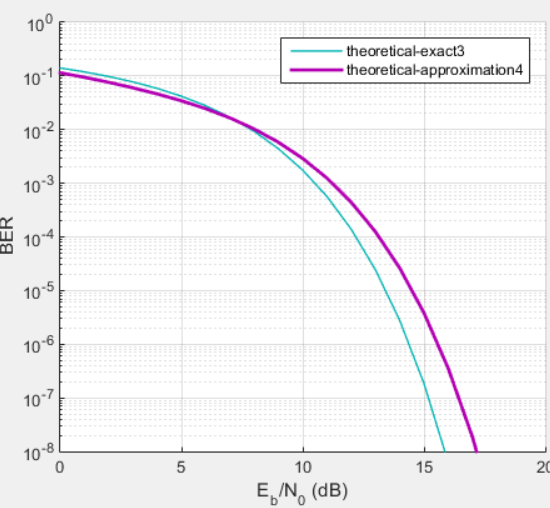
### QPSK Flat channel & (Coding & NoCoding)



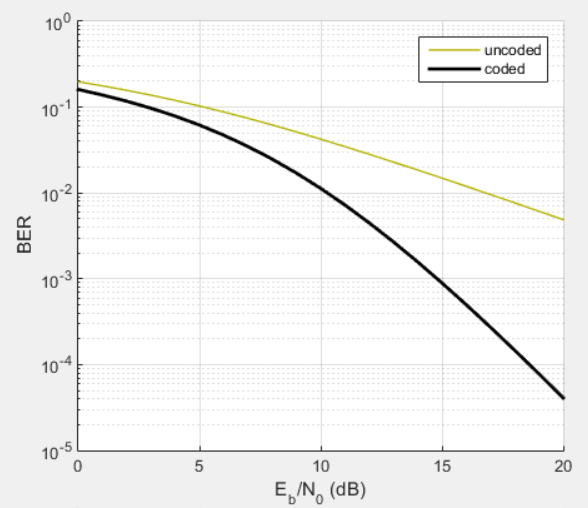
### QPSK Selective channel & (Coding&No Coding)



### 16-QAM Flat channel & (Coding & No Coding)



### 16-QAM Selective channel & (Coding & No Coding)



# Full Code:

## Problem 1:

%%%%%%%%%%%%%%%%%%%%%%%%%%??? ???? ?????? ??????%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%Initialization

clear;clc

close all;

%% Execution Time for DFT Function

L=2048;

n=((0:L-1)/L)\*2\*pi;

x=cos(n);

N=L;

K=[0:N-1];

tic;

X = dft(x,L,N);

toc;

plot(n,x);

figure();

stem(K,abs(X));

%% Execution time of FFT Function

tic;

Z=fft(x,N);

toc;

figure();

stem(K,abs(Z));

function [X] = dft( x, L, N)

%DFT Summary of this function goes here

% Detailed explanation goes here%% Implementing DFT Function

n=[0:L-1];

for K=0:N-1

X(K+1)=sum(x.\*exp((-1j\*(2\*pi\*n\*K))/N));

end

end

## Problem 2:

function BER=drawber16qam(No)

N=1000; % Number of Samples

M=16; % Number bits per symbol

%-------------------Data Generation---------------------------%

data=randi([0 M-1],N,1);

%-----------Grey Encoding------------%

[datagrey,mapgrey] = bin2gray(data,'qam',M);

%-------------------16-QAM Modulation----------------------------%

consttable=[-3-3i, -3-1i, -3+3i, -3+1i,...

-1-3i, -1-1i, -1+3i, -1+1i,...

3-3i, 3-1i, 3+3i, 3+1i,...

1-3i, 1-1i, 1+3i, 1+1i];

for k=1:length(datagrey)

tx(k) = consttable(datagrey(k)+1);

end

tx=tx(:);

% Rayleigh-Fading Channel Model

hr=normrnd(0,sqrt(0.5),1);

hi=normrnd(0,sqrt(0.5),1);

h=(hr+1i\*hi)\*ones(1,N);

h=h(:);

rx=tx.\*h;

% AWGN Channel

%AWGN Noise

mu=0;

variance=No/2;

sigma=sqrt(variance);

nc=normrnd(mu,sigma,[1,N]);

ns=normrnd(mu,sigma,[1,N]);

n=nc+1i\*ns;

n=n(:);

yk=rx+n;

yk=yk/h;

% Correlator & Decision Model

consttable=[-3-3i, -3-1i, -3+3i, -3+1i, -1-3i, -1-1i, -1+3i, -1+1i, 3-3i, 3-1i, 3+3i, 3+1i, 1-3i, 1-1i, 1+3i, 1+1i];

for N = 1:length(yk)

%compute the minimum distance for each symbol

[~, idx] = min(abs(yk(N) - consttable));

datademod(N) = idx-1;

end

[datademodbin,mapbin] = gray2bin(datademod,'qam',M);

datademodbin=datademodbin(:);

% BER

BER=biterr(datademodbin,data)/(length(data)\*log2(M));

end

%% Initialization

clear;clc

close all;

%% 16-QAM Transmitter Model

N=100; % Number of Samples

M=16; % Number bits per symbol

%-------------------Data Generation---------------------------%

data=randi([0 M-1],N,1);

%data=[0:M-1];

%-----------Grey Encoding------------%

[datagrey,mapgrey] = bin2gray(data,'qam',M);

%-------------------16-QAM Modulation----------------------------%

consttable=[-3-3i, -3-1i, -3+3i, -3+1i,...

-1-3i, -1-1i, -1+3i, -1+1i,...

3-3i, 3-1i, 3+3i, 3+1i,...

1-3i, 1-1i, 1+3i, 1+1i];

for k=1:length(datagrey)

tx(k) = consttable(datagrey(k)+1);

end

tx=tx(:);

scatterplot(tx,1,0,'b\*');

for k = 1:16

text(real(tx(k))-0.3,imag(tx(k))+0.3,...

dec2base(data(k),2,4));

text(real(tx(k))-0.3,imag(tx(k))-0.3,...

dec2base(datagrey(k),2,4),'Color',[1 0 0]);

end

title('Transmitted Symbols');

%% Rayleigh-Fading Channel Model

hr=normrnd(0,sqrt(0.5),1);

hi=normrnd(0,sqrt(0.5),1);

h=(hr+1i\*hi)\*ones(1,N);

h=h(:);

rx=tx.\*h;

scatterplot(rx)

title('Rayleigh-Fading Channel Effect');

%% AWGN Channel

%AWGN Noise

mu=0;

No=0.1;

variance=No/2;

sigma=sqrt(variance);

nc=normrnd(mu,sigma,[1,N]);

ns=normrnd(mu,sigma,[1,N]);

n=nc+1i\*ns;

n=n(:);

yk=rx+n;

scatterplot(yk);

title('AWGN Channel Effect');

%% Correlator & Decision Model

consttable=[-3-3i, -3-1i, -3+3i, -3+1i,...

-1-3i, -1-1i, -1+3i, -1+1i,...

3-3i, 3-1i, 3+3i, 3+1i,...

1-3i, 1-1i, 1+3i, 1+1i];

for N = 1:length(yk)

%compute the minimum distance for each symbol

[~, idx] = min(abs(yk(N) - consttable));

datademod(N) = idx-1;

end

[datademodbin,mapbin] = gray2bin(datademod,'qam',M);

datademodbin=datademodbin(:);

%% BER

BER=biterr(datademodbin,data)/(length(data)\*log2(M));

i=1;

for Es\_N0\_dB = [-4:0.1:16]

for meannum=1:100

meanv(meannum)=drawber16qam(10^(-Es\_N0\_dB/10));

end

BER(i)=sum(meanv(:))/100;

i=i+1;

end

Es\_N0\_dB = [-4:0.1:16];

plot(Es\_N0\_dB,1\*log10(BER));

%% Repitition Code

%Repeat by modifying N=N\*3 I removed it from here for better explanation

## Problem 3:

%% Initialization

clear;clc

close all;

%% Parameters

N=21;%Size of OFDM Symbol

m=16;%Number of OFDM Symbols

M=4;

L=1;%Up-Sampling Factor

PoQ=1;%Type of Mapping 1 PSK and 2 QAM

Phase\_Offset=0;%Constellation Phase Offset

Symbol\_Order=2;%Constellation Symbol Order

Ncp=0;%Size of cyclic prefix samples

%% Coding && InterLeaver && Mapper

% Creating Baseband modems Tx/Rx

% data generation

Data=randi([0 M-1], m, N/L);

% Mapping

Dmap=qpskmod(Data);

% Serial to Parallel

parallel=Dmap.';

% Oversampling

upsampled=upsample(parallel,L);

%% 32-point IFFT

% Amplitude modulation (IDFT using fast version IFFT)

am=ifft(upsampled,N);

% Parallel to serial

serial=am.';

%% Add Cyclic extension

% Cyclic Prefixing

CP\_part=serial(:,end-Ncp+1:end); % this is the Cyclic Prefix part to be appended.

cp=[CP\_part serial];

%% Channel

% Adding Noise using AWGN

SNRstart=-4;

SNRincrement=2;

SNRend=16;

c=0;

r=zeros(size(SNRstart:SNRincrement:SNRend));

%% Receiver

for snr=SNRstart:SNRincrement:SNRend

c=c+1;

noisy=awgn(cp,snr,'measured');

% Remove cyclic prefix part

cpr=ofdm.noisy(:,Ncp+1:N+Ncp); %remove the Cyclic prefix

% serial to parallel

parallel=cpr.';

% Amplitude demodulation (DFT using fast version FFT)

amdemod=fft(parallel,N);

% Down-Sampling

downsampled=downsample(amdemod,L);

% Parallel to serial

rserial=downsampled.';

% Baseband demodulation (Un-mapping)

hRx=qpskdemod(rserial);

Umap=hRx;

% Calculating the Symbol Error Rate

[n, r(c)]=symerr(DATA,Umap);

disp(['SNR = ',num2str(snr),' step: ',num2str(c),' of ',num2str(length(r))]);

end

snr=SNRstart:SNRincrement:SNRend;

% Plotting SER vs SNR

semilogy(snr,r,'-ok','linewidth',2,'markerfacecolor','r','markersize',8,'markeredgecolor','b');grid;

title('OFDM Symbol Error Rate vs SNR');

ylabel('Symbol Error Rate');

xlabel('SNR [dB]');