Signals Systems and Communication Lab

Laboratory report submitted for the partial fulfillment
of the requirements for the degree of

Bachelor of Technology
in
Electronics and Communication Engineering

by

Mohit Akhouri - 19ucc023

Course Coordinator
Dr. Navneet Upadhyay

Department of Electronics and Communication Engineering
The LNM Institute of Information Technology, Jaipur

February 2021

Contents

;

Chapter *3*

Experiment - 3

3.1 Aim of the experiment

1. Implementation of discrete Fourier transform (DFT) and inverse DFT (IDFT) algorithm
2. Implementation of autocorrelation and cross correlation algorithm

3.2 Software Used

MATLAB

3.3 Theory

3.3.1 About Discrete Fourier transform (DFT) :

In mathematics, the discrete Fourier transform (DFT) converts a finite sequence of equally-spaced samples of a function into a same-length sequence of equally-spaced samples of the discrete-time Fourier transform (DTFT), which is a complex-valued function of frequency. The interval at which the DTFT is sampled is the reciprocal of the duration of the input sequence. It can be said to convert the sampled function from its original domain to the frequency domain. The sequence of N complex numbers x_0, x_1, x_2,...... , x_{N-1} is transformed into an N-periodic sequence of complex numbers:

$$X[k] = \sum_{n=0}^{N-1} x[n]e^{-j2\pi kn/N}, k \in \mathbb{Z} \tag{3.1}$$

3.3.2 About Inverse Discrete Fourier transform (IDFT) :

An inverse DFT is a Fourier series, using the DTFT samples as coefficients of complex sinusoids at the corresponding DTFT frequencies. It has the same sample-values as the original input sequence. The inverse Fourier transform maps the signal back from the frequency domain into the time domain.

The perfect invertibility of the Fourier transform is an important property for building filters which remove noise or particular components of a signals spectrum. The IDFT is given by :

$$X[n] = \frac{1}{N}\sum_{k=0}^{N-1} x[k]e^{j2\pi kn/N}, n \in \mathbb{Z} \tag{3.2}$$

3.3.3 About Correlation :

Correlation is a measure of similarity between two signals. The relationship between two signals indicates whether one depends on the other, both depend on some common phenomenon, or they are independent. Correlation function indicates how correlated two signals are as a function of how much one of them is shifted in time. There are two types of correlation :

1. Auto correlation
2. Cross correlation

3.3.3.1 Auto Correlation function (ACF) :

It is defined as correlation of a signal with itself. Auto correlation function is a measure of similarity between a signal its time delayed version. Consider a random process x(t) (continuous time) ,Its autocorrelation is computed as :

$$R_{xx}(\tau) = \lim_{T\to\infty} \frac{1}{2T}\int_{-T}^{T} x(t)x(t+\tau)dt \tag{3.3}$$

For sampled signal with N samples, ACF is defined as :

$$R_{xx}(m) = \frac{1}{N}\sum_{n=1}^{N-m+1} x[n]x[n+m-1], m = 1, 2, ...N+1 \tag{3.4}$$

3.3.3.2 Cross Correlation function (CCF) :

Cross correlation is the measure of similarity between two different signals. For two wide sense stationary (WSS) processes x(t) and y(t), the CCF is defined as :

$$R_{xy}(\tau) = \lim_{T\to\infty} \frac{1}{2T}\int_{-T}^{T} x(t)y(t+\tau)dt \tag{3.5}$$

For sampled signal with N samples, CCF is defined as :

$$R_{xy}(m) = \frac{1}{N}\sum_{n=1}^{N-m+1} x[n]y[n+m-1], m = 1, 2, ...N+1 \tag{3.6}$$

3.4 Code and Results

3.4.1 Exercise 1(a) : Creating myDFT (x,N) function to compute DFT of signal x(t) :

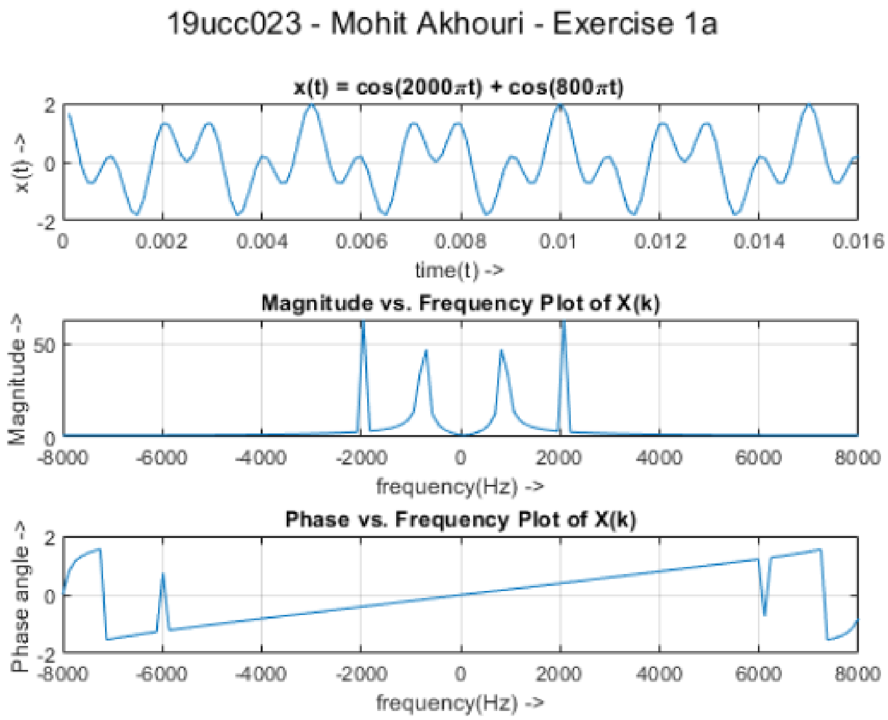
```
function [X] = myDFT(x,N)
% This function will calculate the DFT of function 'x' with total
 samples N

% Name = Mohit Akhouri
% Roll no = 19UCC023
% SSC LAB Batch D1 - Monday ( 2-5 pm )

X=zeros(1,N); %initializing output X
% main loop algorithm for calculating DFT of 'x' starts here
for k=1:N
    sum=0;
    for n=1:N
        sum=sum+(x(n).*(exp(-1j*2*pi*(k-1)*(n-1)/N)));
    end
    X(k)=sum;
end
end
```

Published with MATLAB® R2020b

Figure 3.1 Code for the function myDFT designed for finding DFT of signal x(t)

```
% Name = Mohit Akhouri
% Roll no = 19UCC023
% SSC LAB Batch D1 - Monday ( 2-5 pm )

% This is the code to perform DFT for
 x(t)=cos(2000*pi*t)+cos(800*pi*t)
% by calling function myDFT(x,N)

% initializing constants fs,N,ts and to
fs = 8000;
N = 128;
ts = 1/fs;
to = N*ts;
t = ts:ts:to; % defining range from 'ts' to 'to'
f = linspace(-fs,fs,N); % defining range from '-fs' to 'fs'
x = cos(2000*pi*t) + cos(800*pi*t); % defining x(t)
subplot(3,1,1);
plot(t,x);
xlabel('time(t) ->');
ylabel('x(t) ->');
title('x(t) = cos(2000\pit) + cos(800\pit)');
grid on;

X = myDFT(x,N); % calculating DFT of 'x' using myDFT
mag = fftshift(abs(X));
ang = angle(X);

subplot(3,1,2);
plot(f,mag);
xlabel('frequency(Hz) ->');
ylabel('Magnitude ->');
title('Magnitude vs. Frequency Plot of X(k)');
grid on;

subplot(3,1,3);
plot(f,ang);
xlabel('frequency(Hz) ->');
ylabel('Phase angle ->');
title('Phase vs. Frequency Plot of X(k)');
grid on;
sgtitle('19ucc023 - Mohit Akhouri - Exercise 1a');
```

Figure 3.2 Code for calculating DFT of signal x(t) with N=128 and F_s = 8000 Hz



Published with MATLAB® R2020b

Figure 3.3 Graph of calculated DFT of signal x(t) with N=128 and F_s = 8000 Hz

3.4.2 Exercise 1(b) : Creating myIDFT (X,N) to calculate IDFT of signal x_1[n] = [1 1 1 1] :

```
function [x] = myIDFT(X,N)
% This function will calculate the inverse discrete fourier transform
% (IDFT) of X with total samples N

% Name = Mohit Akhouri
% Roll no = 19UCC023
% SSC LAB Batch D1 - Monday ( 2-5 pm )

x=zeros(1,N); % initializing output
% main loop algorithm to calculate IDFT
for n=1:N
    sum=0;
    for k=1:N
        sum=sum+(X(k).*(exp(1j*2*pi*(k-1)*(n-1)/N)));
    end
    sum=sum/N;
    x(n)=sum;
end
x=abs(x);
end
```

Published with MATLAB® R2020b

Figure 3.4 Code for the function myIDFT designed for finding IDFT of signal $x_1[n]$

```
% Name = Mohit Akhouri
% Roll no = 19UCC023
% SSC LAB Batch D1 - Monday ( 2-5 pm )

% This is the code to perform IDFT for x1[n]={1,1,1,1} with N=4
% by calling function myIDFT(X,N)

%defining constants N
N=4;
x1=[1 1 1 1]; %defining x1[n]
n=0:N-1; %defining range for 'n'

subplot(2,2,1);
stem(n,x1,'Linewidth',1);
xlabel('Total samples (N) ->');
ylabel('x_{1}(n) ->');
title('Plot of x_{1}(n) = [1 1 1 1]');
grid on;

X1=myDFT(x1,N); % calculating DFT of x1[n] using myDFT
mag=fftshift(abs(X1));
ang=angle(X1);

subplot(2,2,2);
stem(n,mag,'Linewidth',1);
xlabel('Total samples (N) ->');
ylabel('Magnitude ->');
title('Magnitude vs. Samples for X_{1}(k)');
grid on;

subplot(2,2,3);
stem(n,ang,'Linewidth',1);
xlabel('Total samples (N) ->');
ylabel('Phase angle ->');
title('Phase vs. Samples for X_{1}(k)');
grid on;

y1=myIDFT(X1,N); % calculating IDFT of X1[k]

subplot(2,2,4);
stem(n,abs(y1),'Linewidth',1);
xlabel('Total samples (N) ->');
ylabel('y_{1}(n) ->');
title('y_{1}(n) obtained after IDFT of X_{1}(k)');
grid on;

sgtitle('19ucc023 - Mohit Akhouri - Exercise 1b');
```

Figure 3.5 Code for calculating IDFT of signal $x_1[n]$ with N=4

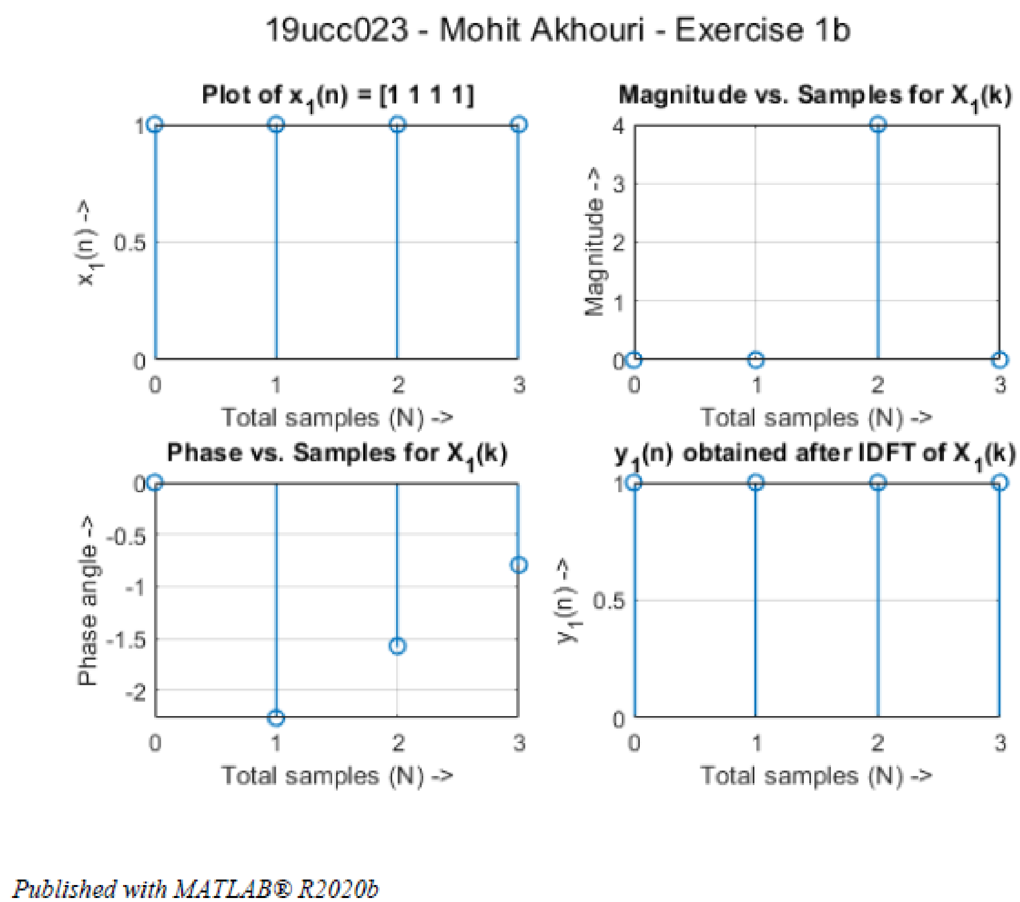
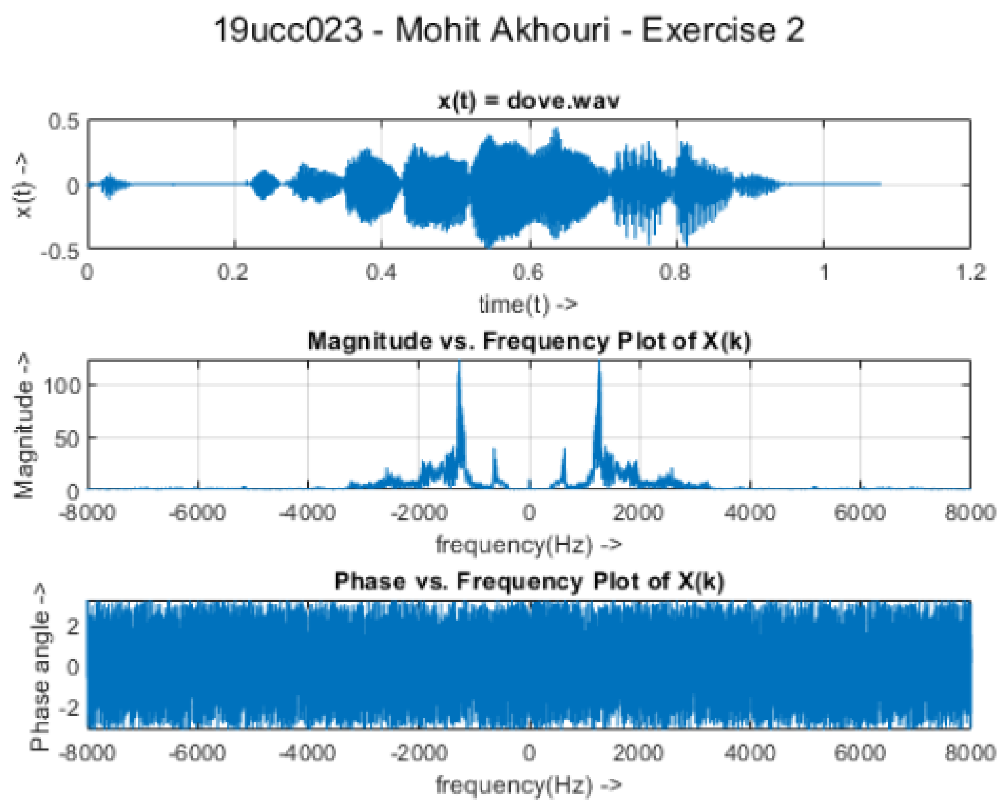


Figure 3.6 Graph of calculated IDFT of signal $x_1[n]$ with N=4

3.4.3 Exercise 2: Plotting the magnitude and phase spectra of speech signal (dove.wav) :

```
% Name = Mohit Akhouri
% Roll no = 19UCC023
% SSC LAB Batch D1 - Monday ( 2-5 pm )

% This is the code to calculate and plot the magnitude and phase
% spectra of speech signal 'dove.wav'

[x,fs]=audioread('dove.wav'); % reading audio signal
% defining constants ts,N,to and f
ts=1/fs;
N=length(x);
to=N*ts;
f=linspace(-fs,fs,N); % defining rangde of 'f'
t=ts:ts:to; % defining range of 't'

subplot(3,1,1);
plot(t,x);
xlabel('time(t) ->');
ylabel('x(t) ->');
title('x(t) = dove.wav');
grid on;

X=myDFT(x,N); % calculating DFT of audio signal
mag=fftshift(abs(X));
ang=angle(X);

subplot(3,1,2);
plot(f,mag);
xlabel('frequency(Hz) ->');
ylabel('Magnitude ->');
title('Magnitude vs. Frequency Plot of X(k)');
grid on;

subplot(3,1,3);
plot(f,ang);
xlabel('frequency(Hz) ->');
ylabel('Phase angle ->');
title('Phase vs. Frequency Plot of X(k)');
grid on;
sgtitle('19ucc023 - Mohit Akhouri - Exercise 2');
```

Figure 3.7 Code for calculating DFT of speech signal (dove.wav) using myDFT function



Published with MATLAB® R2020b

Figure 3.8 Graph of Magnitude and phase spectra of speech signal (dove.wav)

3.4.4 Exercise 3: MATLAB program to calculate the CCF of signals x[n] and y[n] :

```
function [Rxy] = myXCORR(x,y)
% This function will calculate the cross correlation of two signals
% 'x' and 'y'

% Name = Mohit Akhouri
% Roll no = 19UCC023
% SSC LAB Batch D1 - Monday ( 2-5 pm )

% defining constants nx,ny and N
nx=length(x);
ny=length(y);
N=nx+ny-1;
Rxy=zeros(1,N); % initializing Rxy

flip(y); % flipping y

x=[x zeros(1,N-nx)]; % padding with right zeros
y=[zeros(1,N-ny) y]; % padding with left zeros

% main loop algorithm to calculate cross-correlation of 'x' and 'y'
for m=1:N+1
    sum=0;
    for n=1:N-m+1
        sum=sum+(x(n)*y(n+m-1));
    end
    sum=sum/N;
    Rxy(m)=sum;
end

end
```

Published with MATLAB® R2020b

Figure 3.9 Code for function myXCORR(x,y) to perform the CCF of signals x[n] and y[n]

```
% Name = Mohit Akhouri
% Roll no = 19UCC023
% SSC LAB Batch D1 - Monday ( 2-5 pm )

% This code will calculate the cross correlation of two discrete
 signals
% 'x' and 'y' where y=x+w and w=randn(1,N) , a zero-mean , unit
 variance
% of the Gaussian random process

% defining constants f,fs,N and n
f=1;
fs=200;
N=1024;
n=1:N;
x=sin((2*pi*f*n)/fs); % defining x[n]
subplot(2,2,1);
plot(n,x);
xlabel('samples (n) ->');
ylabel('x[n] ->');
title('x[n] = sin((2\pifn)/F_{s})');
grid on;

w=randn(1,N); % defining Gaussian random process
y=x+w; % defining y[n]

subplot(2,2,2);
plot(n,y);
xlabel('samples (n) ->');
ylabel('y[n] ->');
title('y[n] = x[n]+w[n]');
grid on;

Rxy_inbuilt=xcorr(x,y); % calculating cross correlation of 'x' and 'y'
 using inbuilt function
subplot(2,2,3);
plot(Rxy_inbuilt);
xlabel('samples (m) ->');
ylabel('R_{xy}[m] ->');
title('R_{xy}[m] = xcorr(x,y)');
grid on;

Rxy_myfunc=myXCORR(x,y); % calculating cross correlation of 'x' and
 'y' using myXCORR function
subplot(2,2,4);
plot(Rxy_myfunc);
xlabel('samples (m) ->');
ylabel('R_{xy}[m] ->');
title('R_{xy}[m] = myXCORR(x,y)');
grid on;

sgtitle('19ucc023 - Mohit Akhouri - Exercise 3');
```

Figure 3.10 Code for performing CCF of x[n] and y[n] using myXCORR(x,y)

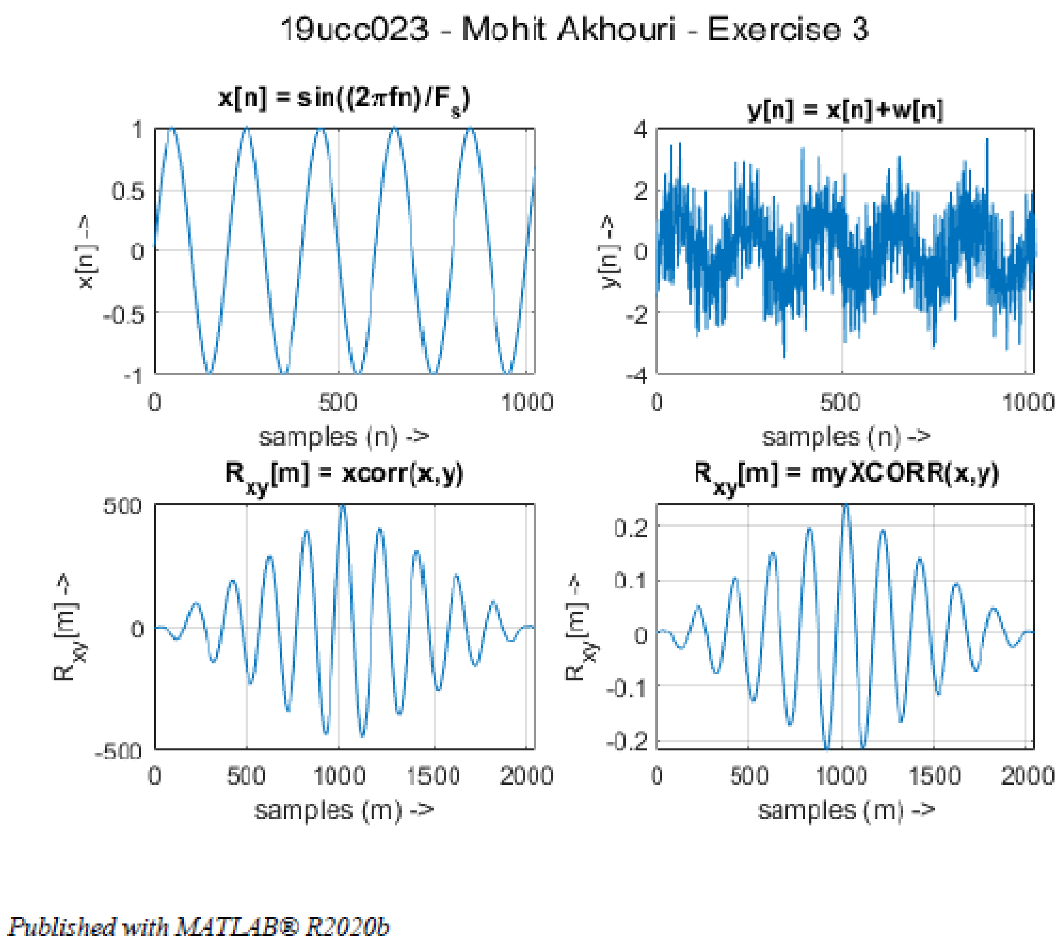


Figure 3.11 Graph for performed CCF of x[n] and y[n] using myXCORR(x,y)

3.5 Conclusion

In this experiment, we learnt about the transforms DFT for converting a signal from time domain to frequency domain and IDFT for the reverse process of converting signal from frequency domain to time domain . We also learnt about the Correlation algorithms (Cross Correlation and Auto Correlation functions) to calculate the similarity between two signals . We also created **myDFT(x,N)** and **myIDFT(X,N)** to perform DFT and IDFT respectively . We also learnt how to process an audio signal in MATLAB using **audioread()** function and how to plot its magnitude and phase spectra. Finally we created **myXCORR(x,y)** function to calculate cross-correlation of two signals x[n] and y[n].