VIDYA JYOTHI
INSTITUTE OF TECHNOLOGY

(An Autonomous Institution)

(Accredited by NAAC & NBA, Approved by AICTE New Delhi & Permanenty Affiliated to JNTUH)

Aziz Nagar Gate, C.B. Post, Hyderabad - 500 075. Phone: Off. 76 76 637 637
Fax: 08413 - 235509 e-mail : info@vjit.ac.in www.vjit.ac.in

Name of the College: VIDYA JYOTHI INSTITUTE OF TECHONOLOGY

Certificate

This is to certify that the bonafide record of the practical work carried out by

Mr/~~Miss~~ ACHARY BHASWANTH *Roll No.* 18911A04C1

of Class III YEAR ECE *in the* DSP LAB

laboratory during the academic year 2020-2021

Faculty In-charge *Head of the Dept.* 17/8/21 *External Examiner*

INDEX

1. Generation of Sinusoidal waveform | Signal based on recursive difference equations
2. To find DFT | IDF of given DT signal
3. Implementation of FFT of given sequence
4. Determination of power spectrum of given signal(s)
5. Implementation of LP & HP FIR filter for a given sequence
6. Implementation of LP & HP IIR filter for a given sequence
7. Generation of DTMF signals.
8. Implementation of I/D sampling rate converters
9. Noise removal: Add noise above 3KHz and then remove, Interference supression using 400Hz tone
10. Impulse response of first order and second order systems.

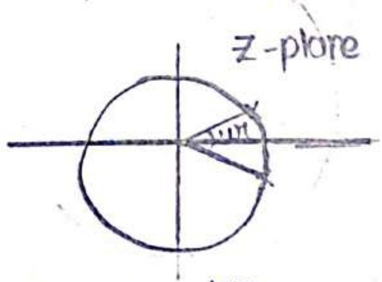
Experiment No 1: Generation of Sinusoidal wave form based on recursive difference equations.

Aim: To generate Sinusoidal wave form based on recursive difference equations

Requirements: personal computer with MATLAB Software

Procedure:

Theory: Sine and cosine functions are very common in communications and DSP applications. In Recursive function Evaluation, we consider a place pole pair on unit circle giving rise to the transfer function.



$$H(z) = \frac{1}{(z-e^{j\omega T})(z-e^{-j\omega T})}$$

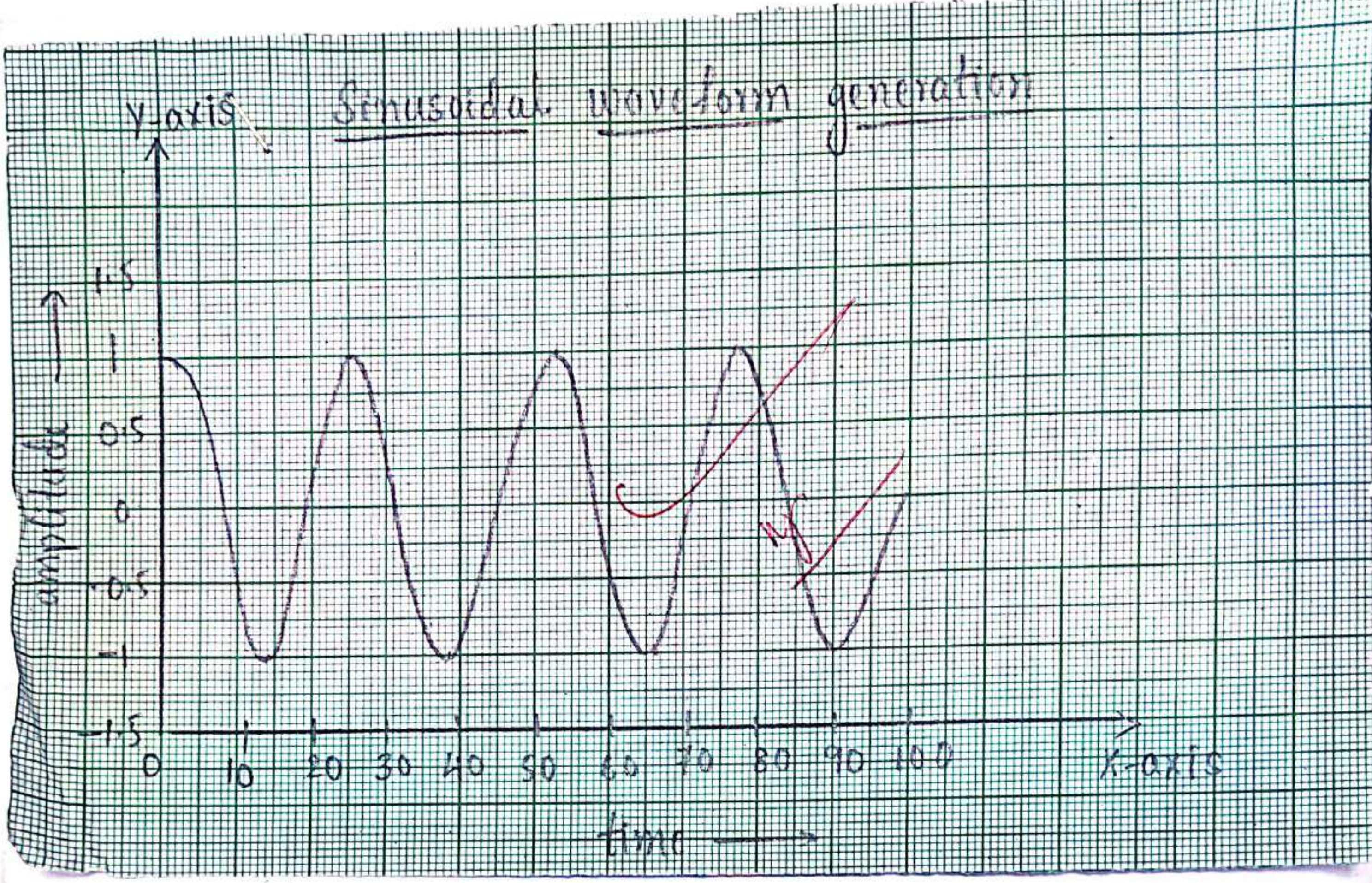
$$= \frac{z^2}{(1-2\cos(\omega T)z^{-1}+z^{-2})}$$

Rewriting the difference equation we get $y(n) = 2\cos(\omega T)y(n-1) - y(n-2) + x(n-2)$
The oscillating at fixed frequency ω with $x(n-2) = 0$ & $x(n)$ being an impulse signal. $\cos(\omega T) = \cos\left(\frac{2\pi f_0}{f_s}\right)$, where f_s is the sampling frequency, f_0 is the desired oscillating frequency. Let $f_0 = 300Hz$ and $f_s = 1/T = 8000Hz$

Procedure: 1. open MATLAB Software and go to file ⇒ new ⇒ M-file
2. Type the program in the editor window and save it as '.m' extension.
3. Select the file saved in the current directory and right click ⇒ run
4. Enter the input values in the command window
5. The result will be shown in the command window and the graph in the figure window.

Program:

```
% clearing and closing commands
clc          % To clear command window
clear all    % To clear the workspace
close all    % To clear the previous waveforms/Graphs if any
```



Sinusoidal waveform generation
y-axis
amplitude
1.5
1
0.5
0
-0.5
-1
-1.5
0
10
20
30
40
50
60
70
80
90
100
x-axis
time

Title : Generation of sinusoidal wave form based on recursive difference equations.

Date :

Page No. : 02

```
% difference equation approach.
% Assumptions
y(1)=1;
y(2)=1;
x=[0 0 1 zeros(1,97)]
for n=3:100
y(n) = 1.9947 * y(n-1) - y(n-2);
end
plot(y)
xlabel('Time -->');
ylabel('Amplitude -->');
title('sinusoidal wave form generation');
```

Result: Sinusoidal signal has been generated using recursive evaluation approach and the corresponding waveforms/graphs are plotted.

Enter the input [1 1 1 1]

x_n = 1 1 1 1

Enter the no. of. points of DFT: 8

N = 8

Xk = 4.0000
1.0000 - 2.4142i
-0.0000 - 0.0000i
1.0000 - 0.4142i
0.0000 - 0.0000i
1.0000 + 0.4142i
0.0000 - 0.0000i
1.0000 + 2.4142i

mag =
4.0000
2.6131
0.0000
1.0824
0.0000
1.0824
0.0000
2.6131

phase = 0
-1.1781
-2.3559
-0.3927
-1.5708
-0.3927
-0.7854
-0.3927
-1.1781

Experiment NO 2: To find DFT / IDFT of a given DT signal

Aim: To write a MATLAB program to generate the DFT / IDFT of a given sequence.

Requirements: personal computer with MATLAB software.

Theory: Discrete Fourier Transform (DFT) is used to perform frequency analysis of discrete time signals. DFT gives a discrete frequency domain representation. The N point DFT of discrete time signals $x[n]$ is given by the equation. DFT

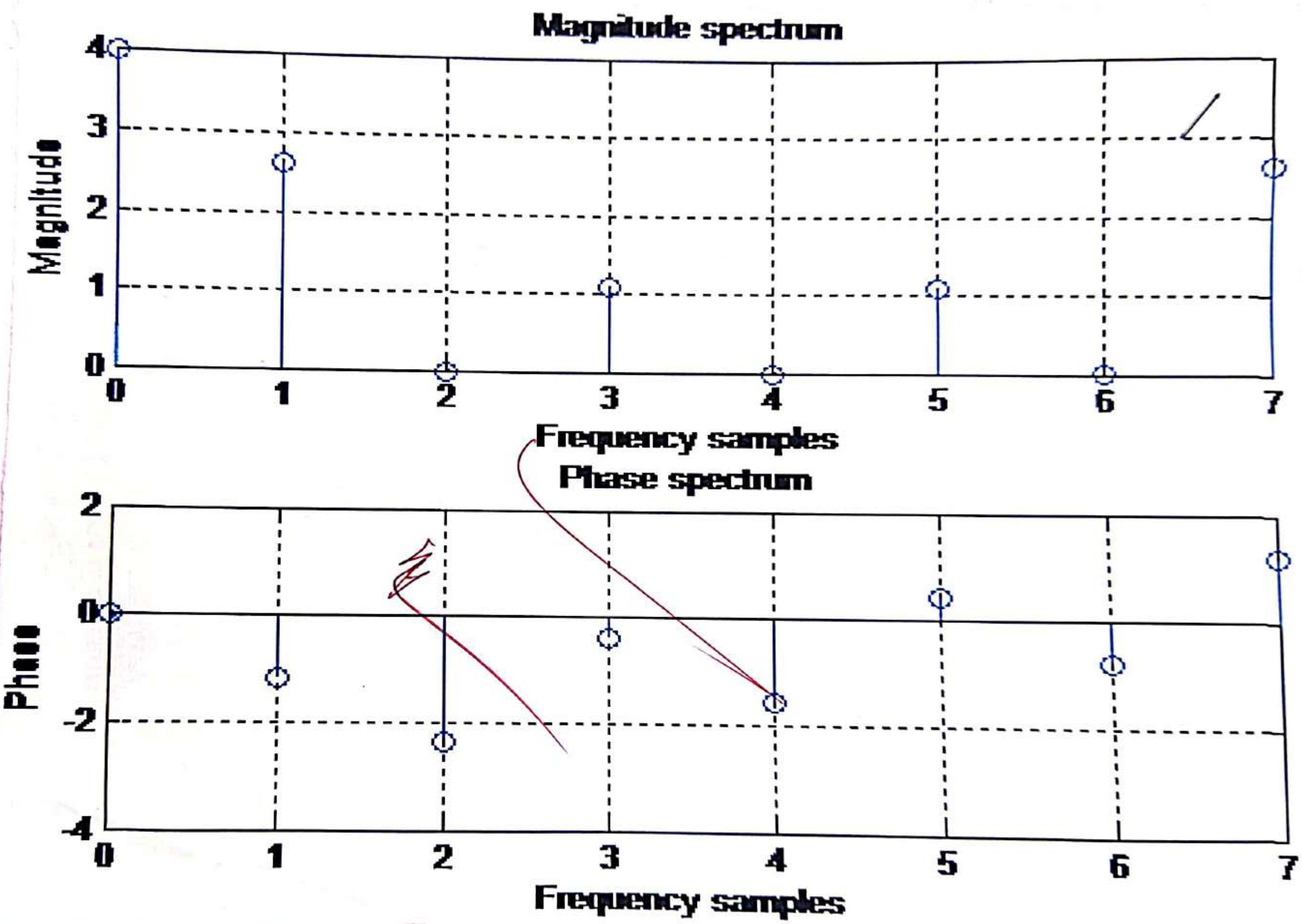
Analysis equation: $X[K] = \sum_{n=0}^{N-1} x(n)\, \omega_N^{Kn} \quad 0 \le K \le N-1$

Synthesis equation: $x(n) = \frac{1}{N}\sum_{K=0}^{N-1} X(K)\, \omega_N^{-Kn} \quad 0 \le n \le N-1$. ω_N = twiddle factor.

Procedure:

Main program:

```
clc
clear all
close all
xn = input('enter the input sequence')
N = input('enter the no.of points of DFT')
xk = fft(xn,N)
mag = abs(xk)
phase = angle(xk)
k=0:1:N-1;
subplot(2 1 1)
stem(k,mag)
xlabel('Frequency samples')
ylabel('Magnitude')
title('Magnitud Spectrum')
grid
subplot(2 1 2)
stem(k,phase)
```


Magnitude spectrum
Magnitude
0
1
2
3
4
0 1 2 3 4 5 6 7
Frequency samples
Phase spectrum
Phase
2
0
-2
-4
0 1 2 3 4 5 6 7
Frequency samples

```
xlabel('frequency samples')
ylabel ('phase')
title ('phase spectrum')
grid
```

Function defining program:

```
function [Xk] = dft(xn,N)
L=length (xn);
if N<L
error ('N must be greater than L');
end
xz = [xn zeros (1,N-L)];
for k=0:1:N-1
for n=0:1:N-1
p=exp(-i*2* pi* n* k/N);
W(k+1,n+1) = p;
end
end
Xk= W* xz;
```

Enter the DFT Sequence [4 1-2.4142i 0 1-0.4142i ...
1+2.4142i]

XK=
columns 1 through 4
4.0000 1.0000-2.4142i 0.0000 1.0000-0.4142i
columns 5 through 8
0.0000 1.0000+0.4142i 0.0000 1.0000+2.4142i

OUTPUT:

N=8
IDFT of a DFT sequence is
columns 1 through 4.
1.0000 1.0000-0.0000i 1.0000-0.0000i 1.0000-0.0000i
columns 5 through 8
-0.0000+0.0000i 0.0000+0.0000i -0.0000-0.0000i 0.0000+0.000...

Enter the DFT Sequence [4 1-2.4142i 0 1-0.4142i 0 1+0.4142i 0 1+2.4142i]

XK=

columns 1 through 4

4.0000 1.0000 - 2.4142i 0.0000 1.0000 - 0.4142i

columns 5 through 8

0.0000 1.0000 + 0.4142i 0.0000 1.0000 + 2.4142i

OUTPUT:

N=8

IDFT of a DFT sequence is

columns 1 through 4.

1.0000 1.0000 - 0.0000i 1.0000 - 0.0000i 1.0000 - 0.0000i

columns 5 through 8

-0.0000 + 0.0000i 0.0000 + 0.0000i -0.0000 - 0.0000i 0.0000 + 0.0000i

Title : To find DFT / IDFT of a DT signal

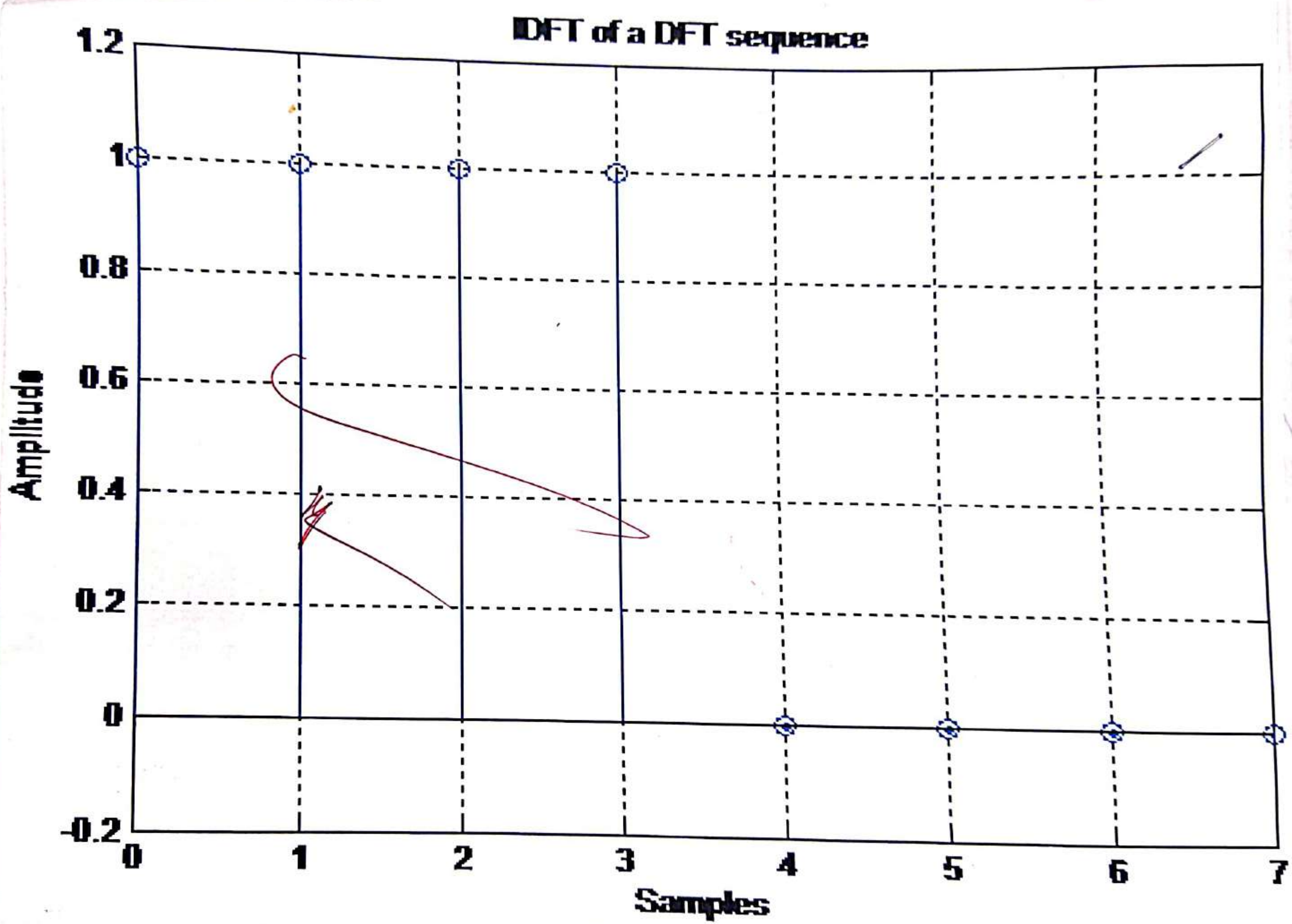
Date :

2. IDFT of a DFT Sequence:

Program:

Main program:

```
clc
clear all
close all
xk = input ('enter the DFT sequence')
N = length (xK)
xn = idft (XK,N);
disp('IDFT of a DFT Sequence is')
disp(xn)
n= 0:1:N-1;
stem (n,xn)
xlabel ('samples')
ylabel ('Amplitude')
title('IDFT of a DFT sequence')
grid
```


IDFT of a DFT sequence
Amplitude
1.2
1
0.8
0.6
0.4
0.2
0
-0.2
0
1
2
3
4
5
6
7
Samples

Title : To find a DFT/ IDFT of a given sequence

Date :
Page No. : 06

Function Defining program:

```
function [xn] = idft (XK,N)
for n=0:1: N-1
for K = 0: 1: N-1
p= exp( i*2* pi* N* K/N);
w (K+1, n+1) = p;
end
end
xn= (xk*w) /N;
```

Result: Thus a MATLAB program is written to generate DFT/IDFT sequence and all sequences are plotted.

Enter the Sequence [1 1 1 1]

x= 1 1 1 1

Enter the length of DFT Sequence 8

N= 8

DFT of the Sequence

columns 1 through 4

4.0000 1.0000 - 2.4142i 0 1.0000 - 0.4142i

columns 5 through 8

0.0000 1.0000 + 0.4142i 0 1.0000 + 2.4142i

Magnitude of XK

4.0000 2.6131 0 1.0824 0 1.0824 0 2.6131

Phase of XK

0.0000 -1.1781 0 -0.3927 0 0.3927 0 1.1781

Experiment NO 3: Implementation of FFT of given sequence

Aim: To write a MATLAB program to generate IDFT & DFT of a given sequence using FFT Algorithm

Requirements: personal computer with MATLAB Software

Theory: The FFT has a fairy easy algorithm to implement

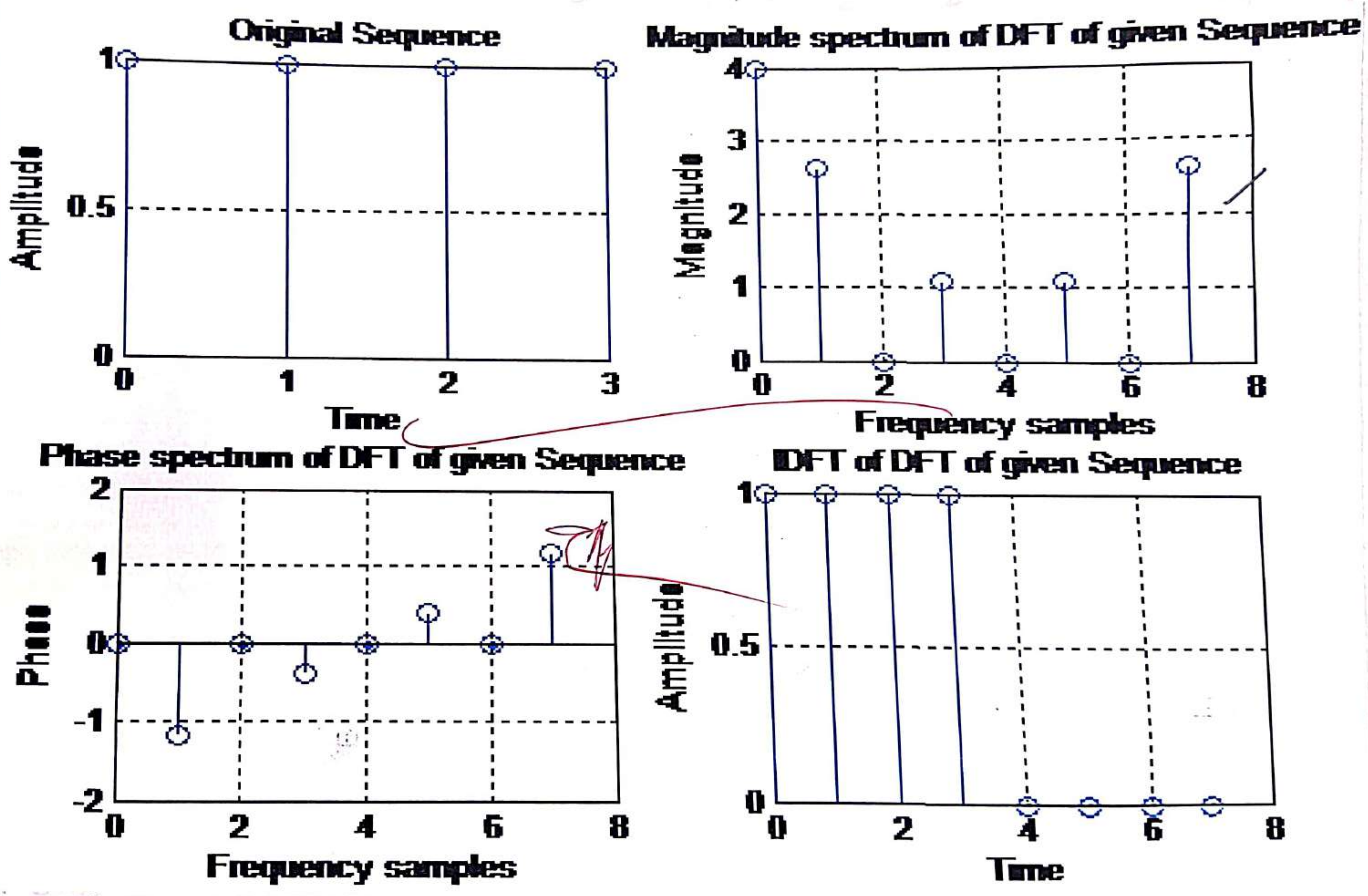
- pad input sequence, of N samples, with zero's until the number of samples is the nearest power of two.
- Bit reverse the input Sequence
- compute (N/2) two sample DFT'S from the shuffled inputs.
- compute (N/4) four sample DFT'S from the two sample DFT'S
- compute (N/2) eight sample DFT'S from the four sample DFTS
  - until the all the samples combine into one N-sample DFT

Procedure:

1. open MATLAB software and go to file ⇒ new ⇒ M-file and save it as '.m' extension
2. Type the program in the editor window and save it in current directory & right click ⇒ run.
3. Select the file saved in current directory & right click ⇒ run.
4. Enter the input values in the command window
5. The result will be shown in the command window and the graph in the figure window.

Program:

```
clc
clear all
close all
x= input ('Enter the sequence')
L= length(x)
N=input('Enter the Length of DFT sequence')
if (N<L)
error('The value of N must be greater than L');
end
```


Original Sequence
Amplitude
Time
Magnitude spectrum of DFT of given Sequence
Magnitude
Frequency samples
Phase spectrum of DFT of given Sequence
Phase
Frequency samples
IDFT of DFT of given Sequence
Amplitude
Time

```
Xk= fft(x,N);
mag =abs(Xk);
phase = angle(xk);
idft =ifft (xk,N);
n=0:1:L-1;
subplot (2 2 1)
stem (n,x)
xlabel ('samples')
ylabel('Amplitude')
title ('original sequence')
grid
n1 = 0:1:N-1;
subplot (2 2 2)
stem (n1,mag)
xlabel('Frequency samples')
ylabel ('Magnitude')
title('Magnitude Spectrum of DFT of given Sequence')
subplot (2 2 3)
stem (n1,phase)
xlable('Frequency Samples')
ylabel ('phase')
title(phase spectrum of DFT of given sequence')
grid
subplot (2 2 4)
stem(n1,idf)
xlabel('Samples')
ylabel('amplitude')
title ('IDFT of DFT of given sequence')
grid
```

```
disp('original sequence')
disp(xk)
disp('DFT of a given sequence')
disp('xk')
disp('magnitude of xk')
disp(mag)
disp('phase of xk')
disp('phase')
```

Result: Thus a MATLAB program is written to generate DFT and IDFT of a given sequence using FFT Algorithm and DFT and IDFT sequence are plotted.

Enter the length of sequence 100

N=100

Enter the Interpolation factor 2

I=2

Enter the decimation factor 2

D=2

Experiment 4: Determination of power Spectrum of given signal(S).

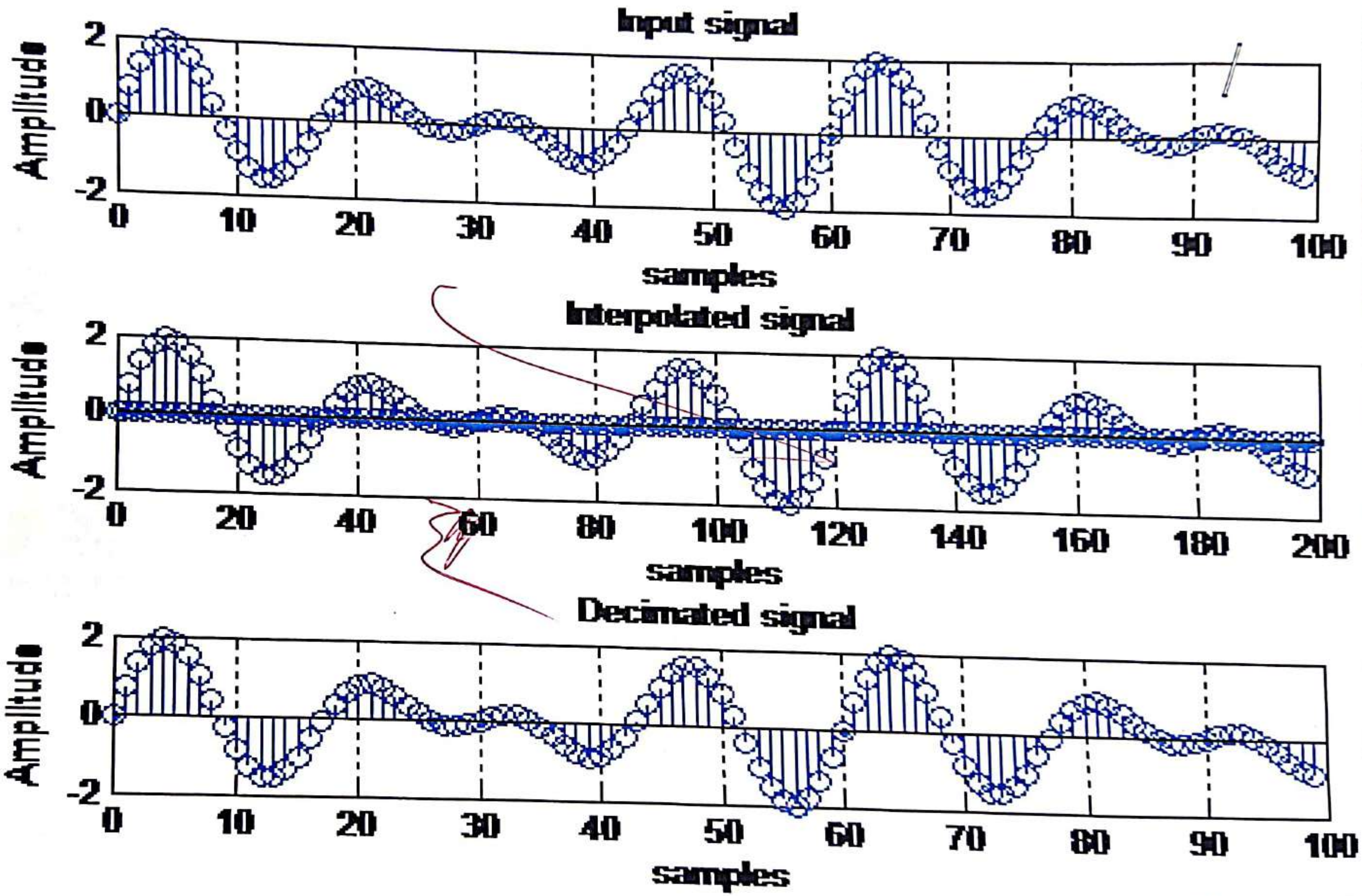
Aim: To write a MATLAB program to generate power spectrum of a signal.

Requirements: personal computer with MATLAB Software

Theory: The sample rate conversion specially has two cases (i) Interpolation (ii) decimation. Sampling rates may be changed upward or downward. Increasing the sample rate is called interpolation, and decreasing the sampling rate is called decimation. Reducing the sample rate by a factor M is acheived by discarding every M-1 samples or equivalently keeping every Mth sample. Increasing the sampleing rate by a factor of L is acheived by Inserting L-1 zeros into the output stream after every sample from Input stream of samples.

Program:

```
clc
clear all
close all
N= input('Enter the length of a sequence')
n=0:1:N-1;
x=sin(2*pi*(n/20))+sin(2*pi*(n/15));
I = input('Enter the interpolation factor')
D= input('Enter the decipolation factor')
xi = zeros(1, I*N);
j=1:I:I*N;
xi(j) =x;
q1 = length(xi);
m = 0:1:q1-1;
y1 = abs(fft(x,N));
y2 = abs(fft(xi,N));
f= (0:N-1)/N;
xd= xi(1: D:I*N);
q2 = length(xd);
```

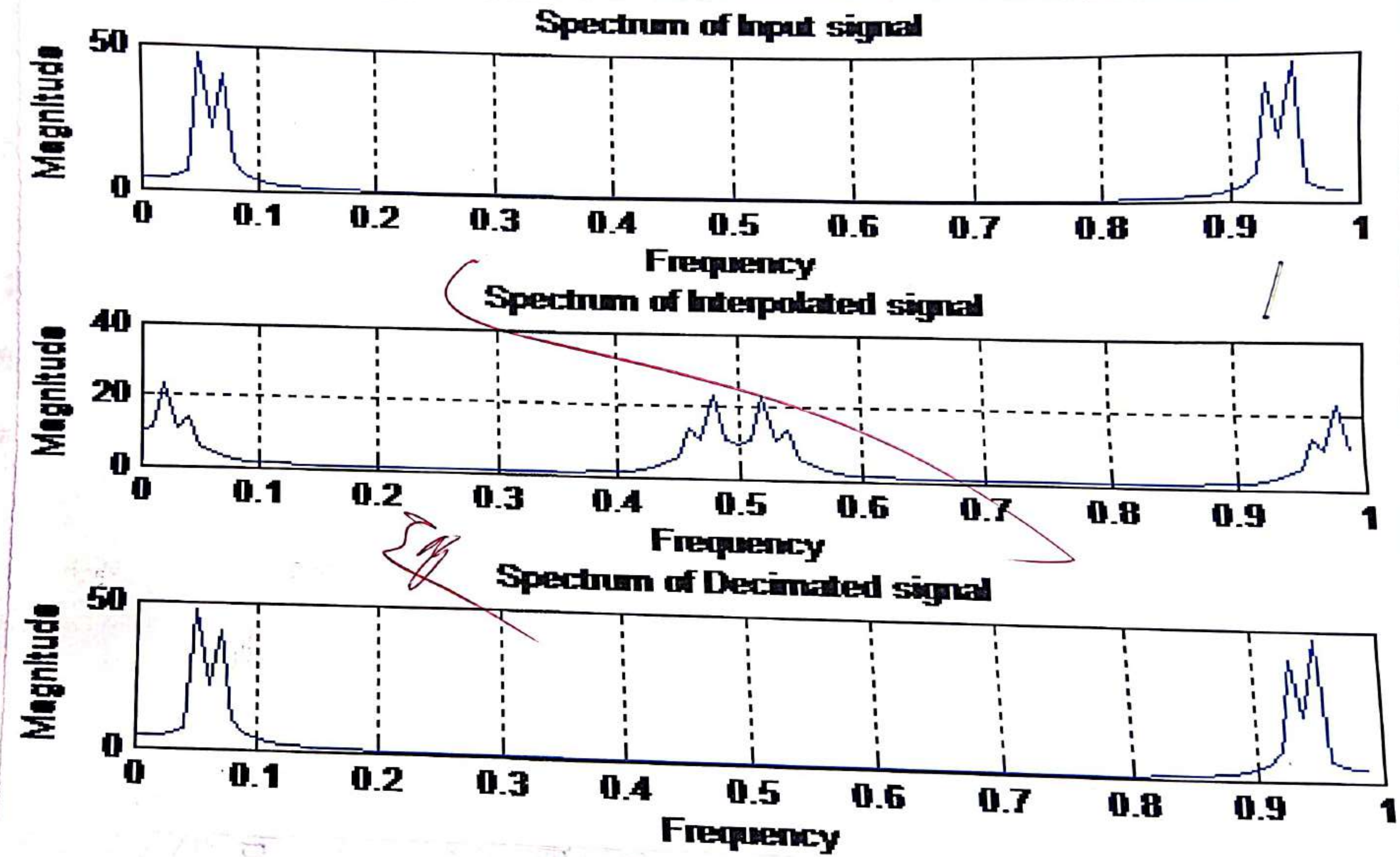


Input signal
Amplitude
2
0
-2
0
10
20
30
40
50
60
70
80
90
100
samples
Interpolated signal
Amplitude
2
0
-2
0
20
40
60
80
100
120
140
160
180
200
samples
Decimated signal
Amplitude
2
0
-2
0
10
20
30
40
50
60
70
80
90
100
samples

Title : Determination of power spectrum of signal(s)

Date :
Page No. : 11.

```
n2 = 0:1:92-1;
y3 = absfft(xd,N);
figure(1)
subplot(3 1 1)
stem(n,x)
xlabel('samples')
ylabel('Amplitude')
title('Input signals')
grid
subplot(3 1 2)
stem (n1,x1)
xlabel('samples')
ylabel('Amplitude')
title('Interpolated signal')
grid
subplot (3 1 3)
stem (n2,xd)
xlabel('samples')
ylabel('Amplitude')
title('Decimated signal')
grid
figure(2)
subplot (3 1 1)
plot (f,y1)
xlabel('Frequency')
ylabel ('Magnitude')
title('spectrum of Input signal')
grid
```


Spectrum of Input signal
Magnitude
Frequency
Spectrum of Interpolated signal
Magnitude
Frequency
Spectrum of Decimated signal
Magnitude
Frequency

```
Subplot (3 1 2)
plot(f,y2)
xlabel('frequency')
ylabel('magnitude')
title('spectrum of Interpolated signal')
grid
subplot (3 1 3)
plot(f,y3)
xlabel('frequency')
ylabel('Magnitude')
title('spectrum of decimated signal')
grid.
```

Result: Thus a MATLAB program is written to generate powerspectrum of a signal and power spectrum is plotted.

Enter the window length 7

N=7

Ente

wc=

outpu

wir

b1=

wi

0.1

0.3

0.6

0.8

0.6

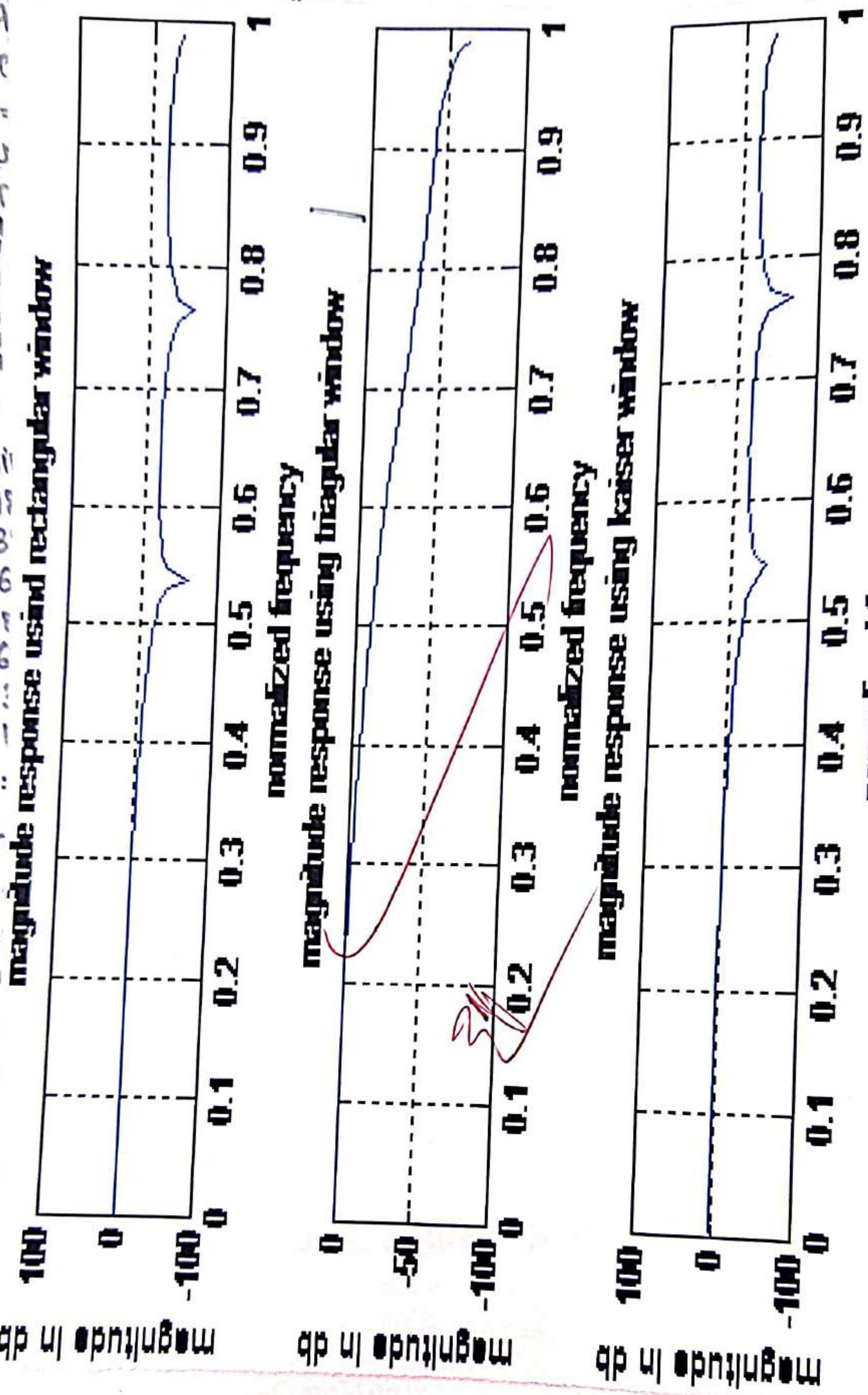
0.3

0.1

b2:

wir

b3

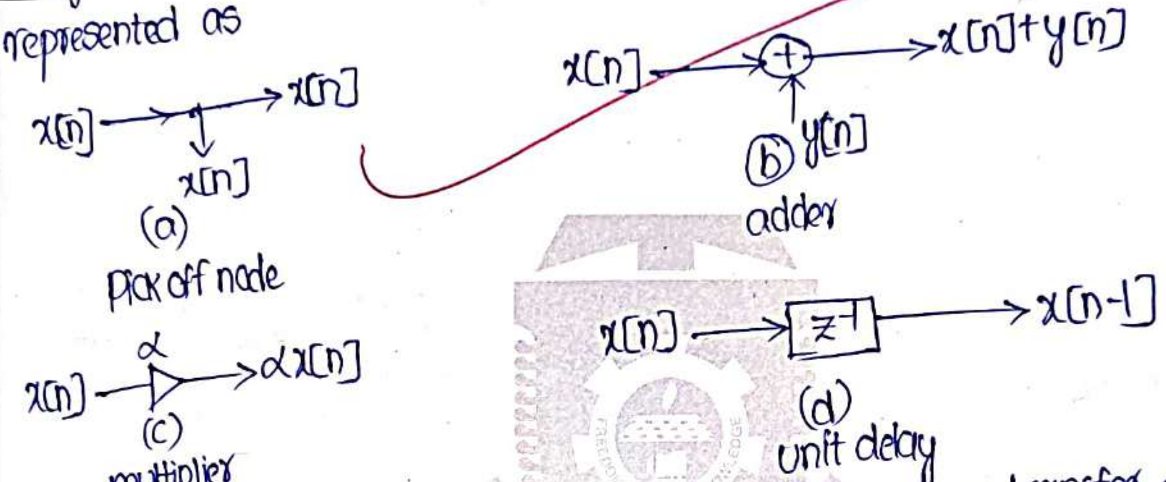


Experiment NO5: Implementation of Lp & Hp FIR filter of a given sequence.

Aim: To write a MATLAB program to generate the magnitude response of FIR digital Lp & Hp filter using windowing technique.

Requirements: Personal computer with MATLAB Software

Theory: The computational algorithm of LTI digital filter can be represented represented as



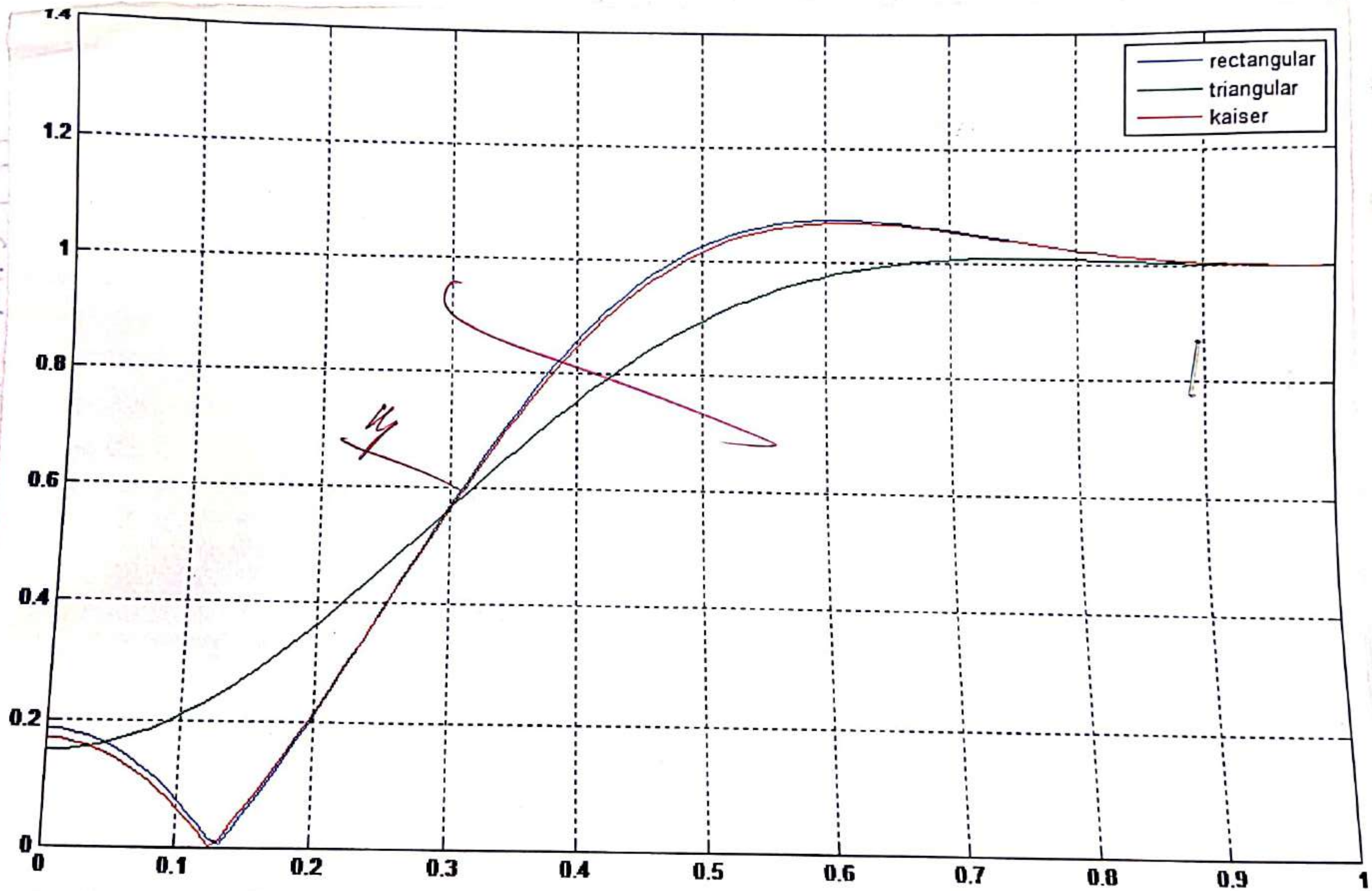
A causal FIR filter of length M is characterized by a transfer function H(z):

$$H(z) = \sum_{K=0}^{M-1} h[K] z^{-K}$$

which is a polynomial in of degree M-1. In the time domain the input-output relation of FIR filter is $y[n] = \sum_{K=0}^{M-1} h[K]\, x[n-K]$

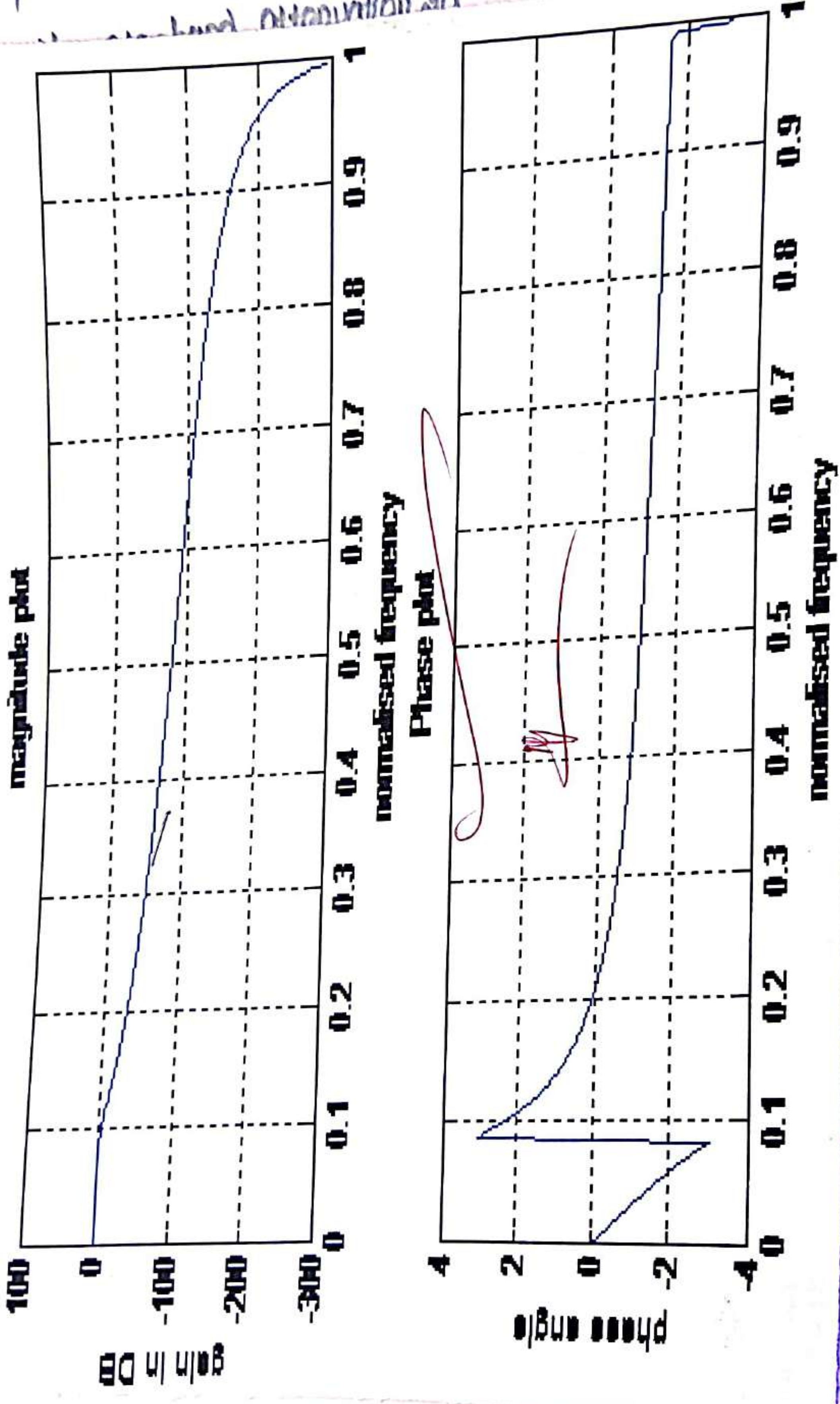
Program for FIR Low pass filter

```
clc
clear all
close all
N= input('Enter window length')
wc= input ('Enter cut off frequency')
window1 = rectwin (N+1)
b1 = fir1 (N,wc, window1)
[h1,w1] = freqz(b1,1,128);
Subplot(3,1,1)
plot( w1/pi, 20*log10(abs(h1)))
```


rectangular
triangular
kaiser
0
0.2
0.4
0.6
0.8
1
1.2
1.4
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

Result: The magnitude response characteristics of a Lowpass FIR filter and High Pass FIR filter using windowing technique are generated using MATLAB.

Enter the passband frequency 500
wp= 500
Enter the stopband frequency 1000
ws=1000
Enter the passband attenuation 1
Ep= 1
...stopband attenuation 20



0.0570

Experiment No 6: Implementation of LP &HP IIR filter for a given sequence.

Aim: To write a MATLAB program to generate the frequency response of IIR filter

Requirements: personal computer with MATLAB Software

Theory: In the design of frequency selective Network filters, the desired filter characteristics are specified in the frequency domain in terms of the desired magnitude and phase response of the filter. In the filter design process, we determine the coefficients of a causal IIR filter that closely approximates the desired frequency response specifications.

In many applications of signal processing we want to change the relative amplitudes and frequency contents of a signal. This process is generally referred as filtering. since frequency response of system and input fourier transform product gives the fourier transform of output, we have to use appropiate frequency response

Program for IIR Lowpass filter:

```
clc
clear all
close all
wp=input('enter the bandpass frequency')
ws=input('enter the stopband frequency')
rp=input ('enter the passband attenuation')
rs=input('enter the stop band attenuation')
W1 = 2* wp/fs;
W2 = 2* ws/fs;
if(W1>W2)
   error('for LPF W1 should be less than W2');
end
[N,wc]=buttord (w1,w2, rp,rs,'s')
[b,a] = butter(N,wc,'s')
```

Enter the pass band frequency 1000

wp = 1000

Enter the stopband frequency 500

ws = 500

Enter the passband attenuation 1

rp = 1

Enter the stop band attenuation 20

rs = 20

<u>output</u>

N = 5

wc = 0.7917

b = 1 0 0 0 0 0

a = 1.0000 2.5618 3.2815 2.5978 1.2910 0.3109

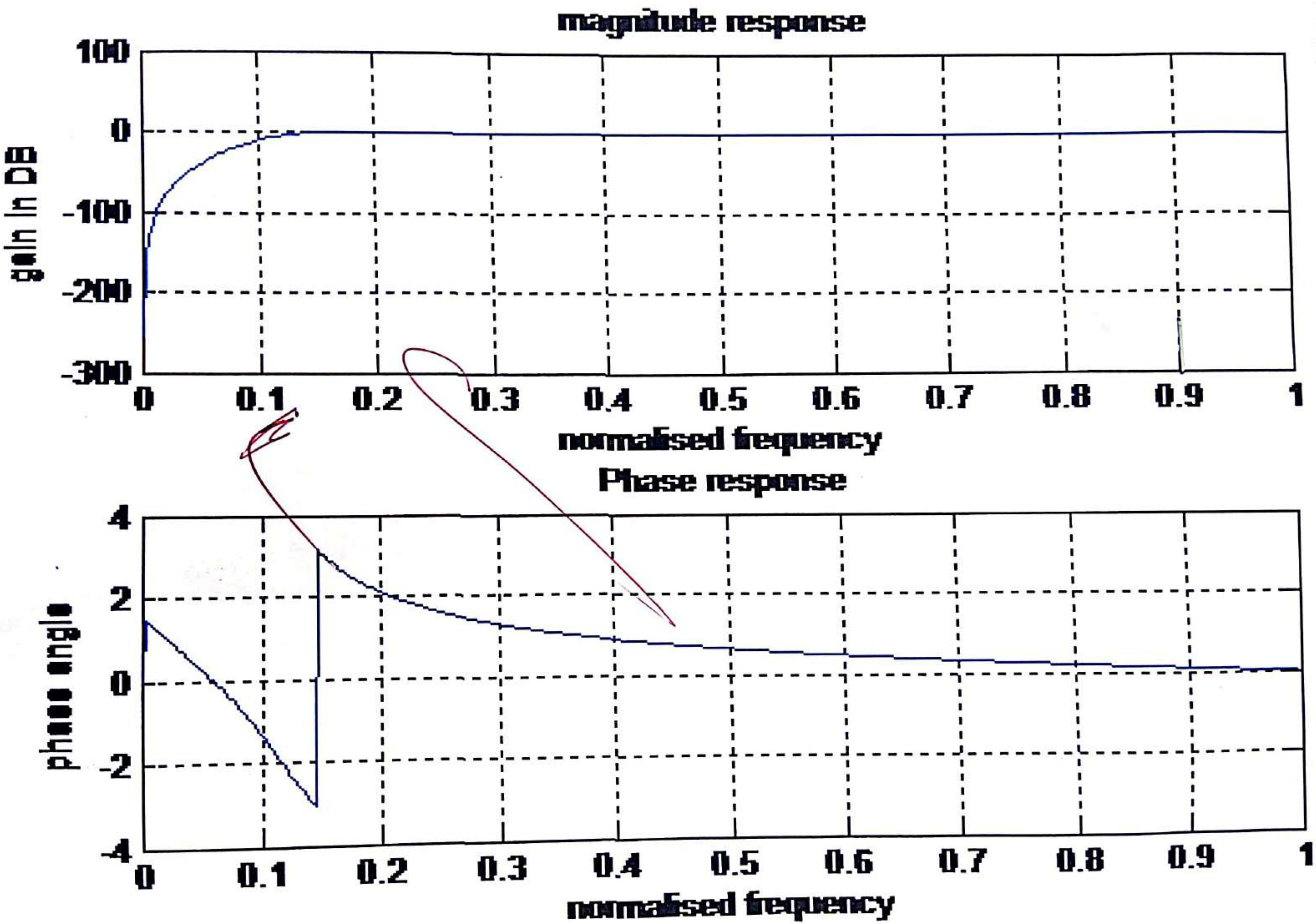
b1 = 0.5287 -2.6455 5.2871 -5.2871 2.6485 -0.5287

a1 = 1.0000 -3.7377 5.2140 -4.7395 1.7505 -0.2395

```
[b1,a1] = bilinear (b,a,2)
w=0:0.01:pi;
[h,omega] = freqz (b1,a1, w );
gain = 20*log10(abs(h));
an = angle(h);
subplot (2,1,1)
plot (omega/pi,gain)
xlabel('normalised frequency')
ylabel('gain in DB')
title ('magnitude plot')
grid
subplot (2,1,2)
plot(omega/pi,an)
xlabel ('normalised frequency')
ylabel ('phase angle')
title('phase plot')
grid
```

Program for IIR High pass filter :

```
clc
clear all
close all
wp= input('enter the passband frequency')
ws =input ('enter the stopband frequency')
rp = input('enter the passband attenuation')
rs = input ('enter the stopband attenuation')
fs =input ('enter the normailising frequency')
w1= 2* wp/fs;
w2 = 2* ws/fs;
if(w1<w2)
    error('for HPF w1 should be more than w2');
else
```


magnitude response
gain in DB
100
0
-100
-200
-300
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
normalised frequency
Phase response
phase angle
4
2
0
-2
-4
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
normalised frequency

```
[N,wc] = buttord(W1,W2,rp,rs,'s'))
[b,a] = butter(N,wc, high,'s'))
[b1,a1] = bilinear(b,a,2)
w=0:0.01:pi;
[h,omega] = freqz(b1,a1,w);
gain = 20*log(abs(h));
an=angle(h);
subplot(2,1,1)
 plot(omega/pi,gain)
 xlabel('Normalised frequency')
 ylabel('gain in DB')
 title('magnitude plot')
 grid
 subplot(2,1,2)
 plot(omega/pi,an)
 xlabel('normalised frequency')
 ylabel('phase angle')
 title('phase plot')
 end
```

Result: The magnitude and phase response characteristics of IIR high pass butterworth filter are generated using MATLAB

Enter symbol 0

Key = 0

<u>Output:</u>

symbol = 1 2 3 4 5 6 7 8 9 0 * #

lfg = 697 770 852 941

hfg = 1209 1336 1477

f' =	697	1209
	697	1336
	697	1477
	770	1209
	770	1336
	770	1477
	852	1209
	852	1336
	852	1477
	941	1209
	941	1336
	941	1477

Experiment NO 7: Generation of DTMF SIGNALS:

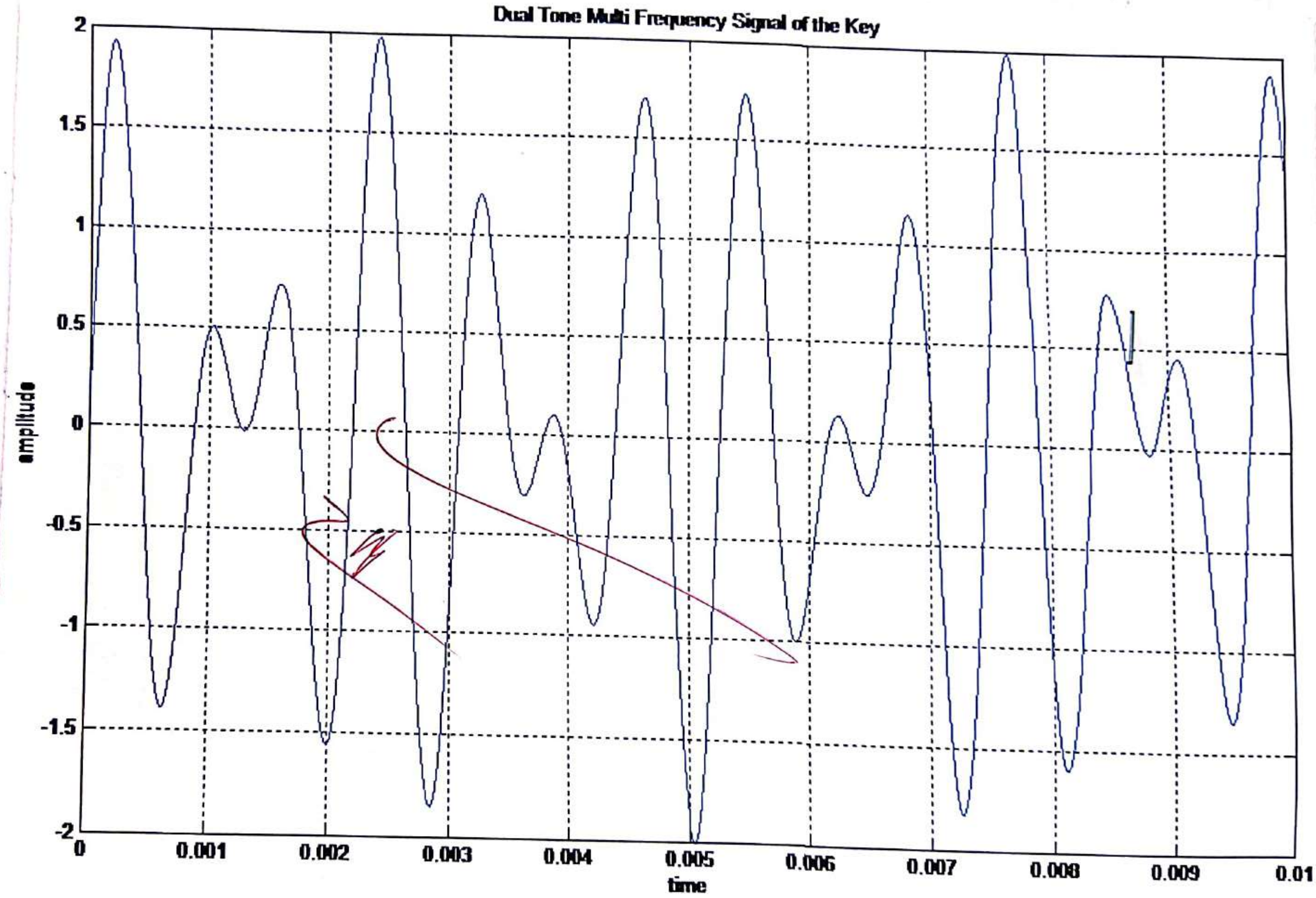
Aim: To write a MATLAB Program to generate DTMF signals of the keypad

Requirements: Personal computer with MATLAB software

Theory: Analog DTMF telephone signalling is based on encoding standard telephone Keypad digits and symbols in two audible sinusoidal signals of frequencies LFG and HFG. Thus the scheme gets it name as DTMF (Dual tone multi frequency). DTMF signalling is basis for voice communications control and is widely used worldwide in modern telephony to dial numbers and configure switchboards. It is also used in systems such as in voice mail, electronic mail and telephone banking. The minimum duration of a DTMF signal defined by ITU standard is 40ms.

Program:

```
clc
clear all
close all
symbol = ['1','2','3','4','5','6','7','8','9','0','*','#']
lfg = [697 770 852 941]
hfg = [1209 1336 1477]
f=[];
fs=8000;
n=800;
t=(0:(n-1))/fs;
for c=1:4
for r=1:3
f=[f[lfg(c);hfg(r)]];
end
f)
end
key=input('enter symbol')
switch key
```


Dual Tone Multi Frequency Signal of the Key
amplitude
time
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
0
0.001
0.002
0.003
0.004
0.005
0.006
0.007
0.008
0.009
0.01

Title : Generation of DTMF Signals

Date :

Page No. : 21

```
case 1
m=sin(2*pi* 697 *t) +sin(2* pi* 1209*t);
case 2
m= sin (2*pi*697*t)+sin(2* pi* 1336*t);
case 3
m= sin (2*pi*697*t)+sin (2*pi* 1477*t);
case 4
m= sin(2*pi*770*t) +sin( 2*pi* 1209 *t);
case 5
m=sin(2*pi* 770*t) +sin(2*pi* 1336*t);
case 6
m=sin (2*pi*770*t)+sin (2* pi* 1477*t);
case7
m=sin(2* pi* 852*t)+sin(2*pi* 1209*t);
case8
m= sin (2*pi* 852*t) +sin(2* pi* 1336 *t);
case9
m=sin(2* pi* 852*t) +sin (2*pi* 1477*t);
case '*'
m= sin (2*pi* 941*t)+sin (2*pi* 1209*t);
case 0
m=sin (2* pi* 941*t) +sin(2*pi* 1336*t);
case '#'
m=sin (2*pi* 941*t) +sin (2* pi* 1477*t);
otherwise
m=0;
end
plot (t,m)
xlabel ('time')
ylabel('amplitude')
title(' Dual tone Multi frequency signal of the key')
grid
```

Result: Thus, a MATLAB program is written to generate DTMF signals and signals are plotted.

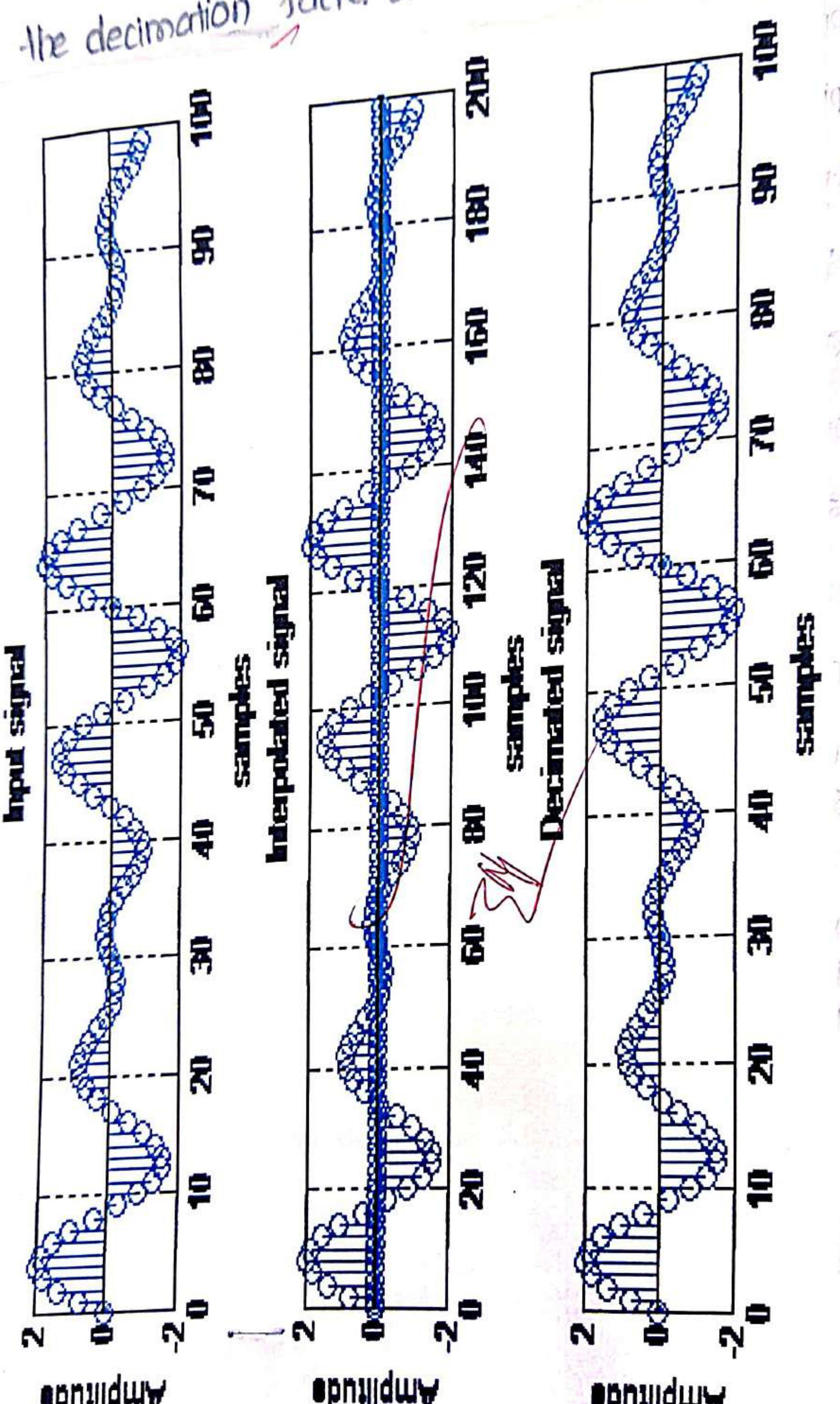
N=100

Enter the interpolation factor 2

I=2

Enter the decimation factor 2

D=



Experiment No 8: Interpolation / Decimation (I/D) of Given sequence

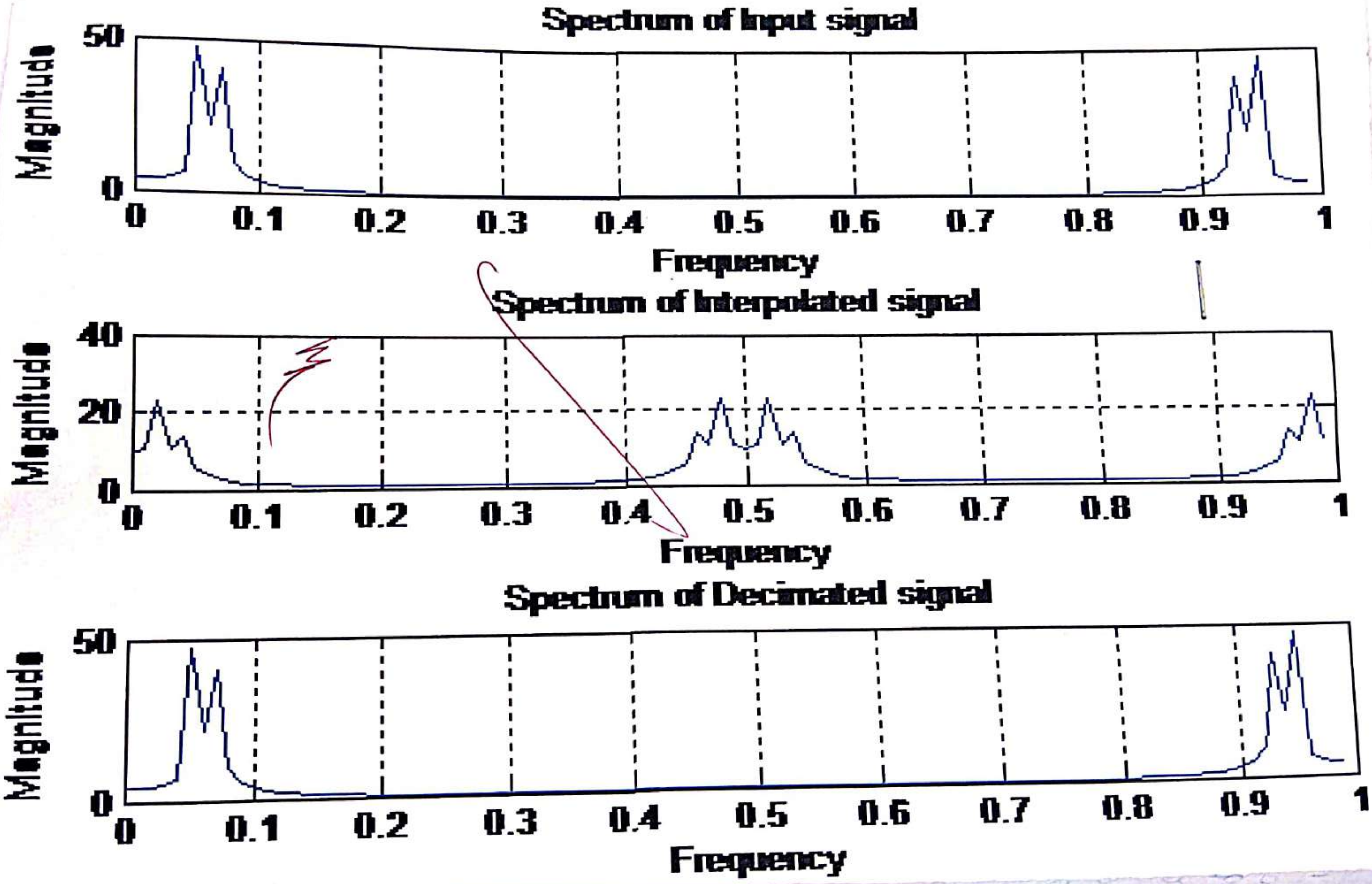
Aim: To write a MATLAB program to generate the interpolated/ decimated output of a given sequence.

Requirements: personal computer with MATLAB software.

Theory: The sample rate conversion specially has two cases (i) Interpolation (ii) decimation sampling rates can be changed upward or downward. Increasing the sampling rate is called Interpolation and decreasing the sampling rate is called decimation. Reducing the sample rate by "M" is achieved by discarding every M-1 samples or equivalently keeping every M^{th} sample. Increasing the sampling rate by L is achieved by a factor of L-1 zeros are inserted into the output stream.

Program:

```
clc
clear all
close all
N=input ('Enter the length of sequence')
n=0:1:N-1;
x=sin(2*pi*(n/80))+sin (8*pi*(n/15));
I= input ('Enter the Interpolation factor')
D= input ('Enter the decimation factor')
xi = zeros(1,I*N);
j = 1:I:I*N;
xi(j) = x;
y1 =abs(fft(x,N));
y2 = abs(fft(xi,N));
f= (0:N-1)/N;
xd = xi(1 : D: I*N);
q2 = length(xd);
n2 = 0:1:q2-1;
y3 =abs(fft(xd,N));
figure(1)
subplot (3 1 1)
stem(n,x)
xlabel('samples')
```


Spectrum of Input signal
Magnitude
50
0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Frequency
Spectrum of Interpolated signal
Magnitude
40
20
0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Frequency
Spectrum of Decimated signal
Magnitude
50
0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Frequency

```
ylabel('Amplitude')
title('Input signals')
grid
subplot(3 1 2)
stem(n1,xi)
xlabel('samples')
ylabel ('Amplitude')
title('Interpolated signal')
grid
subplot(3 1 3)
stem(n2,xd)
xlabel('samples')
ylabel('Amplitude')
title('Decimated signals')
grid
figure(2)
subplot(3 1 1)
plot(f,y1)
xlabel('frequency')
ylabel('Magnitude')
title('spectrum of Input signal')
grid
subplot(3 1 2)
plot(f,y2)
xlabel('frequency')
ylabel('Magnitude')
title('spectrum of Interpolated signal')
grid
subplot(3 1 3)
plot(f,y3)
xlabel('frequency')
ylabel('Magnitude')
title('spectrum of decimated signal')
grid
```

Result: A MATLAB program is written to generate Interpolation and decimation (I/D) of a sequence.

Output (first order):

Enter coefficients of numerator: 1

b=1

Enter coefficients of denominator: [1 -0.6]

a = 1.0000   -0.6000

ydirect =

columns

1.0000

columns

0.0280

columns

0.0008

yzt =

column

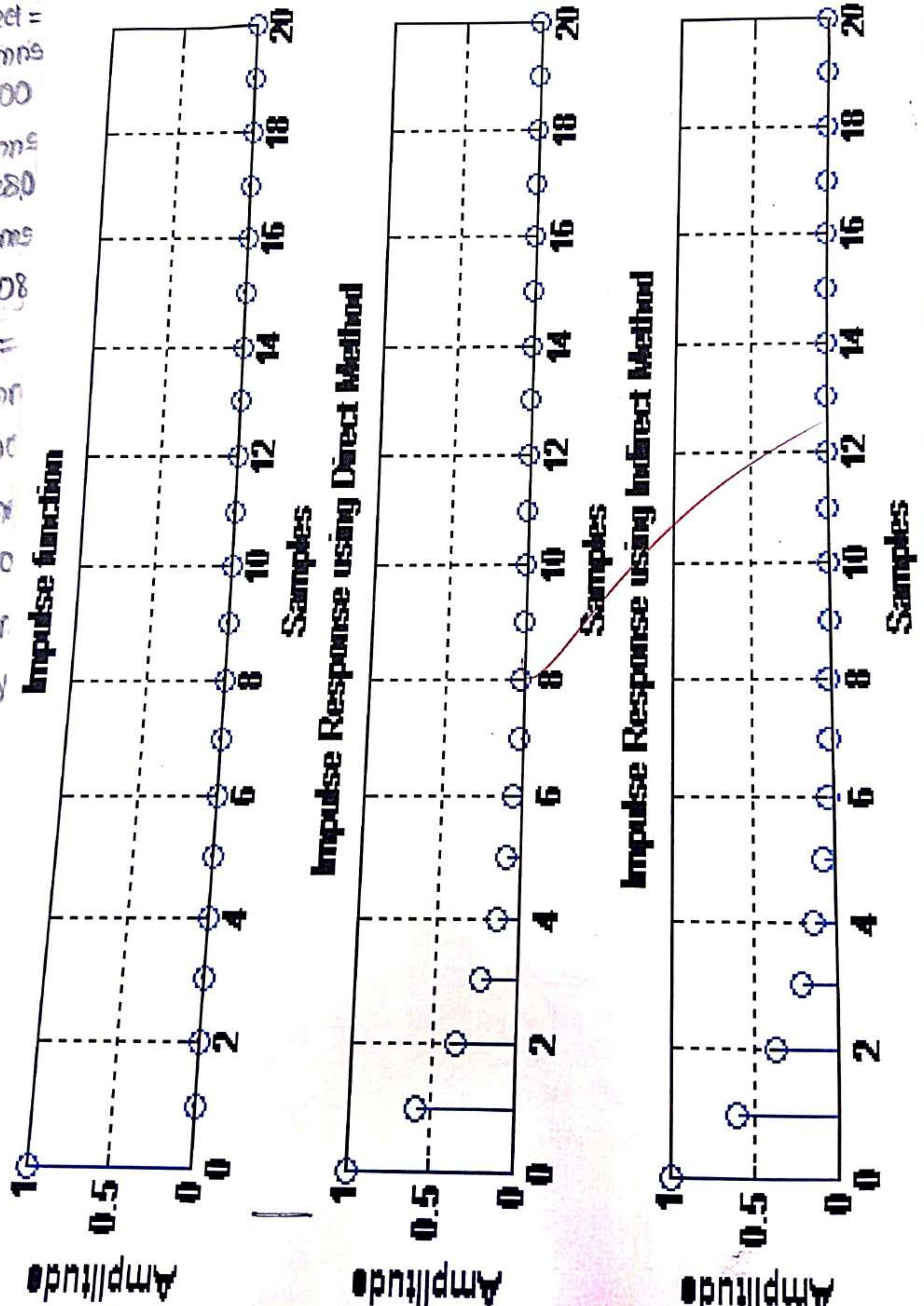
1.0000

column

0.0880

column

0.000



Title : Impulse response of first order and second order System Date :
Page No. : 84.

Experiment No 9: Impulse response of first order and second order system:

Aim: To write a program on MATLAB to generate Impulse response of first order and second order system.

Requirements: personal computer with MATLAB software.

Theory: Noise is added with the original signal is removed. The response of the system when the input is impulse function is called impulse response. If h(n) denotes impulse response of an LTI discrete-time fourier transform of h(n) that is

$$H(e^{jw}) = \sum_{n=-\infty}^{\infty} h(n) e^{-jw}.$$

By applying z-transform $H(z) = Y(z) X(z)$.

Program:

```
clc
clear all
close all
n=0:1:20;
x=[1 zeros(1,20)];
b=input('Enter the coefficients of Numerator')
a=input ('Enter the coefficients of Denominator')
ydirect =(0.6).^n
yzt =filter(b,a,x)
subplot (3 1 1)
stem(n,x)
xlabel('samples')
ylabel('Amplitude')
title('Impulse function')
grid
subplot(3 1 2)
stem (n,ydirect)
title('Impulse response using Direct Method')
xlabel('samples')
ylabel ('Amplitude')
grid
subplot (3 1 3)
stem (n,yzt)
```

output: Second order.

Enter coefficients of numerator : 1

b=1

Enter coefficients of denominator : [1 -0.6 0.08]

a=

1.0000 -0.6000 0.0800

ydirect =

columns 1:

1.0000 C

columns

0.0083

colums

0.0000

y2t =

columns

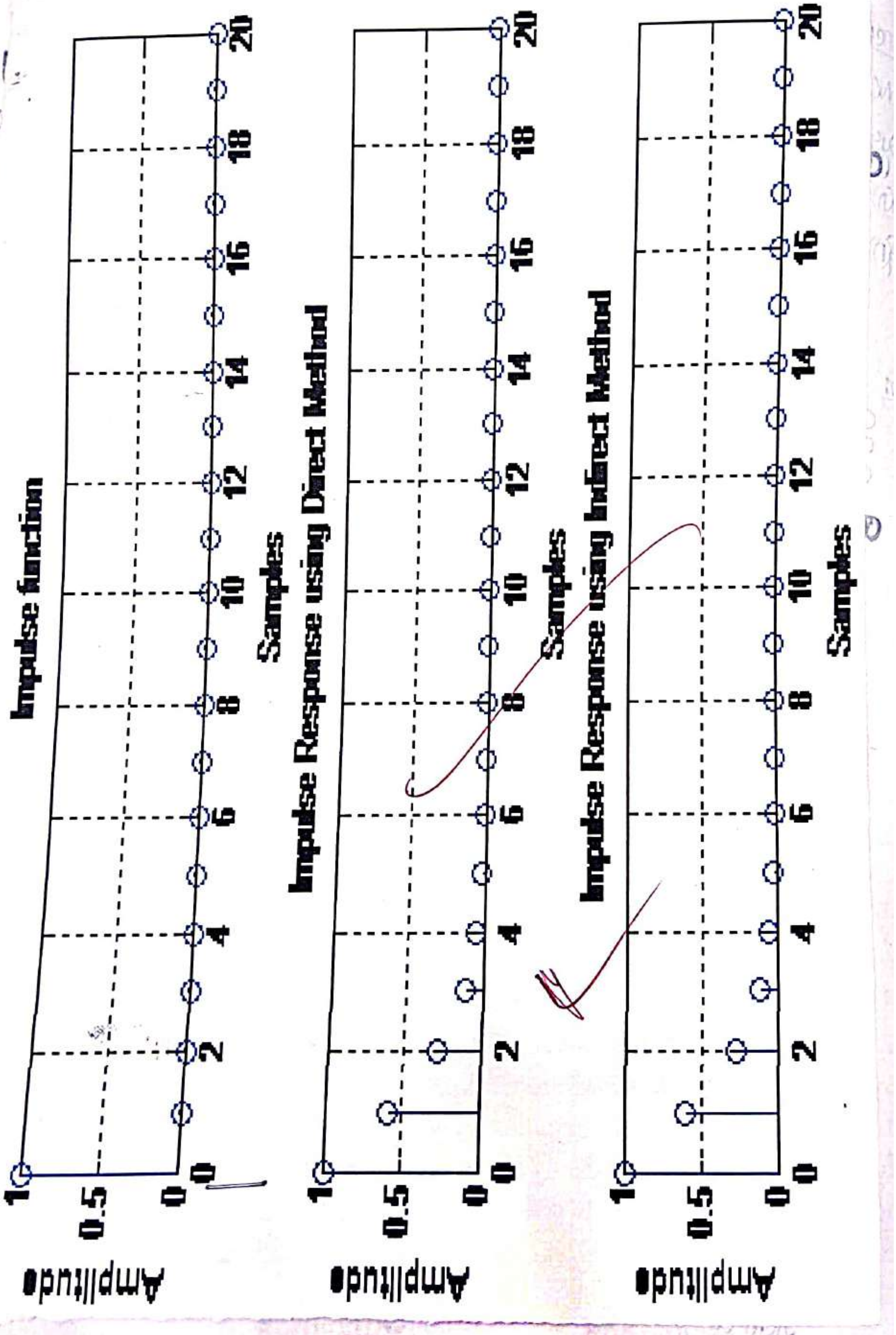
1.0000

columns

0.0083

columns

0.0000



```
title('Impulse Response using Indirect Method')
xlabel('samples')
ylabel('Amplitude')
grid
```

2. second order System

```
clc
clear all
close all
n=0:1:20;
x=[1 zeros(1,20)];
b=input('Enter the coefficients of numerator')
a=input('enter the coefficients of denominator')
ydirect= 2*(0.4).^n - (0.2).^n
yzt =filter(b,a,x)
subplot(3 1 1)
stem(n,x)
xlabel('samples')
ylabel ('Amplitude')
title('Impulse function')
grid
subplot(3 1 2)
stem(n,ydirect)
xlabel('samples')
ylabel('Amplitude')
title('Impulse Response using Direct Method')
grid
subplot(3 1 3)
stem(n,yzt)
xlabel('samples')
ylabel('Amplitude')
title('Impulse Response using Indirect Method')
grid
```

Result: Thus a MATLAB program is written to generate Impulse response of first and second order systems and impulse responses are plotted.

EN

Enter the passband frequency 450

$F_p = 450$

Enter the stopband frequency 2000

$f_s = 2000$

enter the passband attenuation 1

$r_p = 1$

enter the stop band attenuation 20

$r_s = 20$

enter the sampling frequency 4000

$f_s = 4000$

<u>output</u>:

N=2

$\omega c = 0.9960$

a = 1.0000   1.4085   0.9919

b1 = 0.0438   0.0877   0.0438

a1 = 1.0000   -1.3266   0.5020

Experiment NO 10: Noise.

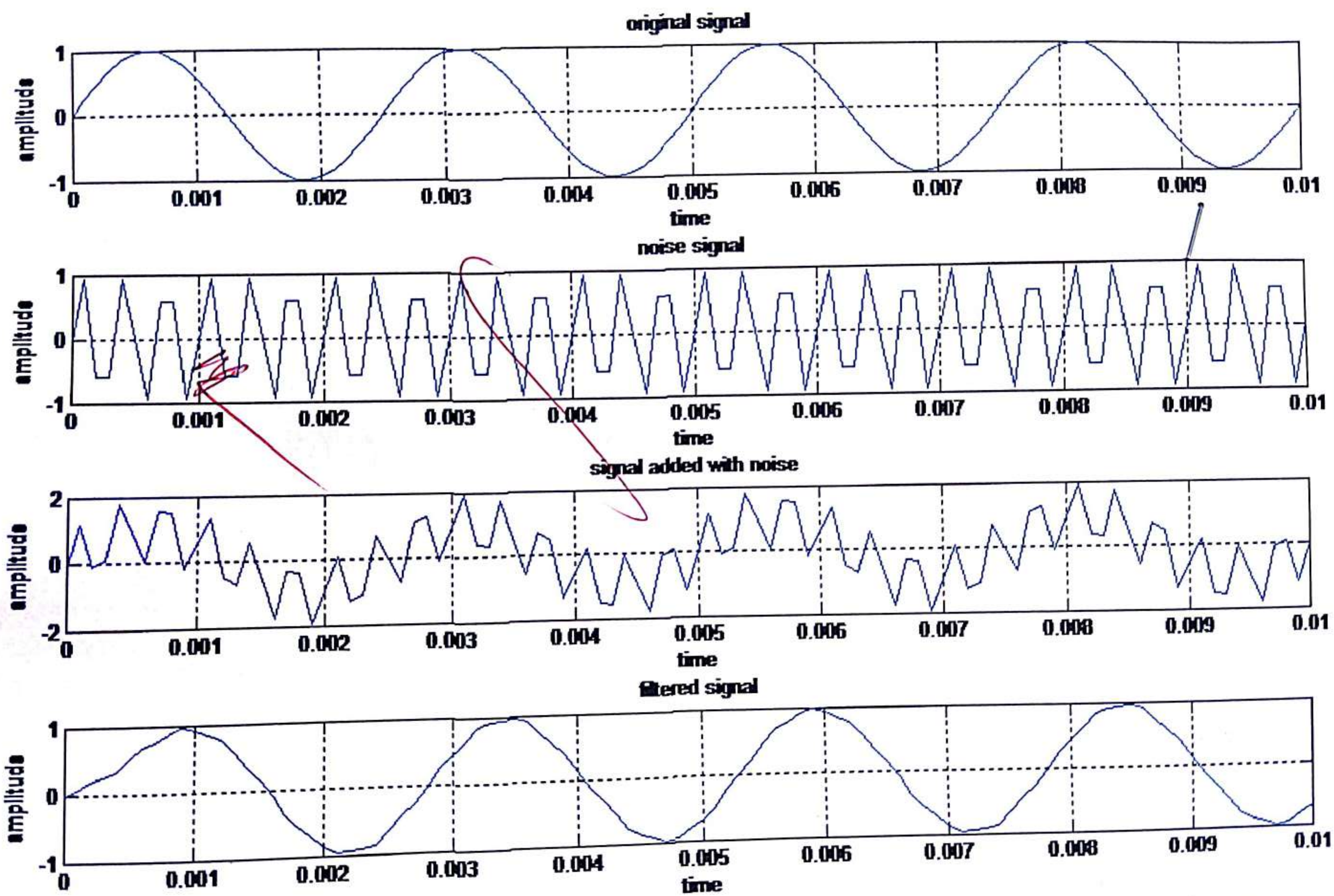
Aim: To write a MATLAB program to remove noise from noisy signal.

Requirements: personal computer with MATLAB software.

Theory: Noise is added to original signal so that the original signal is converted into Noisy signal. so that signal added with Noise can be converted into original signal through some filters.

Program:

```
clc
clear all
close all
t=0:0.0001:0.01;
x=sin(2*pi*400*t);
noise =sin(2*pi*3000*t);
x1= x+noise;
fp= input('enter the pass band frequency')
fs= input('enter the stop band frequency')
rp= input('enter the pass band attenuation')
rs= input('enter the stop band attenuation')
fs = input('enter the sampling frequency')
w1 =2* pi* fp/fs;
w2= 2* pi* fs/fs;
[N,wc] = buttord(w1,w2,rp,rs,'s')
[b,a] = butter(N,wc,'s')
[b1,a1] =bilinear (b,a,2)
y=filter (b1,a1, x1);
subplot(411)
plot(t,x)
xlabel('time')
ylabel('amplitude')
title ('original signal')
grid
subplot(412)
plot(t,noise)
xlabel('time')
ylabel('amplitude')
```


original signal
amplitude
time
noise signal
amplitude
time
signal added with noise
amplitude
time
filtered signal
amplitude
time

```
title('Noise signal')
grid
subplot(413)
plot(t,x)
xlabel('time')
ylabel('amplitude')
title('signal added with noise')
grid
subplot(414)
plot(t,y)
xlabel('time')
ylabel('amplitude')
title('filtered signal')
grid
```

Result: Thus a MATLAB program is written to remove the noisy signal from the original signal and the signals are plotted.