Lab No.:1 Exp No.: 7a Sum of Sinusoidal Signals

Aim: Write a program in MATLAB to generate four sinusoidal signals of four different amplitudes and frequencies and its sum.

Toolboxes: Basic Mathematics and Graphics Toolboxes of MATLAB

List of commands / functions: sin( ), subplot( ), plot( ), title( ), xlabel( ), ylabel( ), figure, padarray( ), 'colon'(;) operator

Theory:

Procedure:

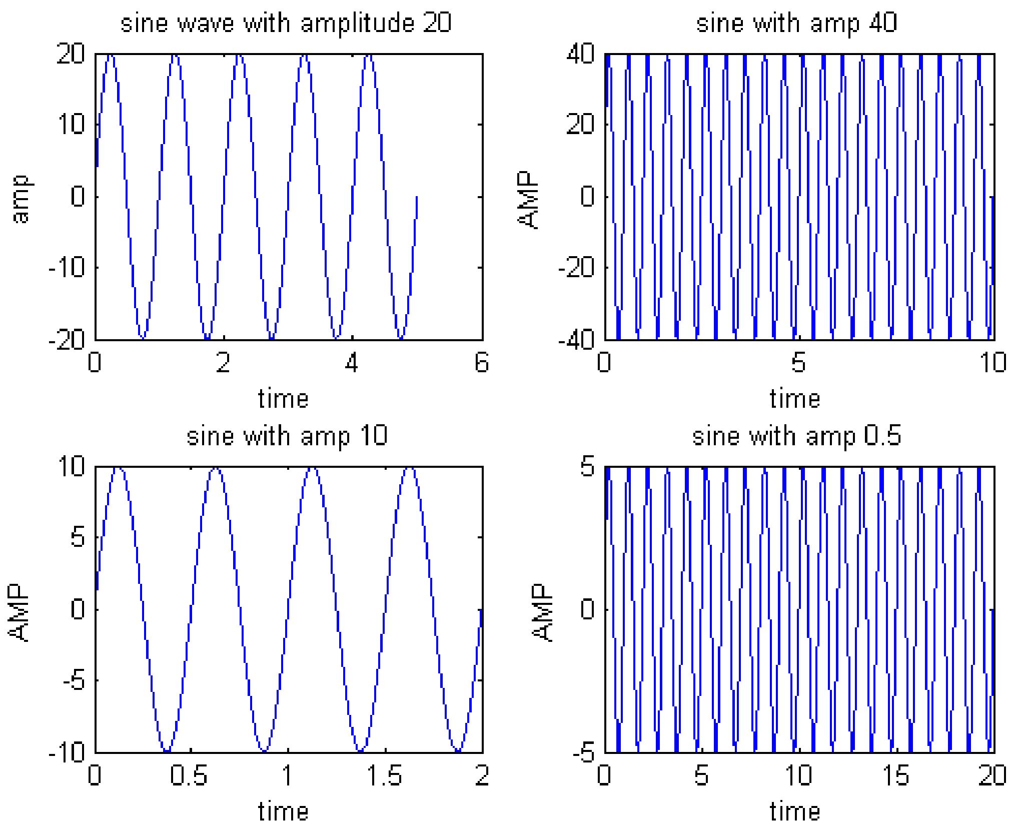
1. Signal addition is performed by adding corresponding samples in both the signals i.e., sample by sample addition.
2. When both the signals are of equal duration signal addition is straight forward.
3. Signals of different duration are added first by padding appropriate number of zero valued samples(zero padding) to the shorter signal, and then by sample by sample addition.

Program:

```
%% Exp 7a:
%TO Find SUM of 4 SINUSOIDAL waves of 4 different amplitudes

clc
close all
clear all

%SINE with amplitude 20
t1=0:0.001:5;
A=20;
xf=1;
x=A*sin(2*pi*xf*t1);
subplot(2,2,1), plot(t1,x), title('sine wave with amplitude 20'),
xlabel('time'), ylabel('amp')

%SINE with amplitude 40
t2=0:0.001:10;
B=40;
yf=2;
y=B*sin(2*pi*yf*t2);
subplot(2,2,2), plot(t2,y), title('sine with amp 40'),
xlabel('time'), ylabel('AMP')

%sine with amplitude 10
t3=0:0.001:2;
C=10;
zf=2;
z=C*sin(2*pi*zf*t3);
```

```
subplot(2,2,3), plot(t3,z), title('sine with amp 10'),
xlabel('time'), ylabel('AMP')

%sine with amplitude 0.5
t4=0:.001:20;
D=0.5;
wf=1;
W=D*sin(2*pi*wf*t4);
subplot(2,2,4), plot(t4,W), title('sine with amp 0.5'),
xlabel('time'), ylabel('AMP')

%addition of all the above sinusoids
x1=padarray(x,[0,15000],0,'post');
y1=padarray(y,[0,10000],0,'post');
z1=padarray(z,[0,18000],0,'post');
v=x1+y1+z1+W;
figure,plot(t4,v), title('sum of sines'), xlabel('time'),
ylabel('amp')
```

Result: Sum of sinusoidal signals generated using MATLAB.



Lab No.:1 Exp No.: 7b Illustration of Gibb's Phenomena

Aim: Write a program in MATLAB to reconstruct a square signal using Fourier coefficients and observe the Gibb's phenomenon.
Toolboxes: Basic Mathematics and Graphics Toolboxes of MATLAB
List of commands / functions: square( ), sin( ), input( ), plot( ), title( ), xlabel( ), ylabel( ), axis( ), legend( ), grid on, hold on, hold off, 'colon'(;) operator
Theory:

Procedure:
It can be seen from Fourier series that a perfect square signal can be generated by adding infinite number of sinusoidal signals with odd harmonics and amplitudes equal to reciprocal of odd harmonics respectively. But construction using only finite number of such sinusoidal signals always leads to the presence of oscillations and overshoots of 9%of amplitude at the edges or step discontinuities of square signal a phenomena called Gibbs phenomena.

1. Specify the time period of the required square wave.
2. Specify the number of harmonics to be added in the Fourier series.
3. Initialize the square wave with zero valued samples on the defined time axis.
4. Generate and add sinusoidal signals according to Fourier series representation of square wave.
5. Display the result.

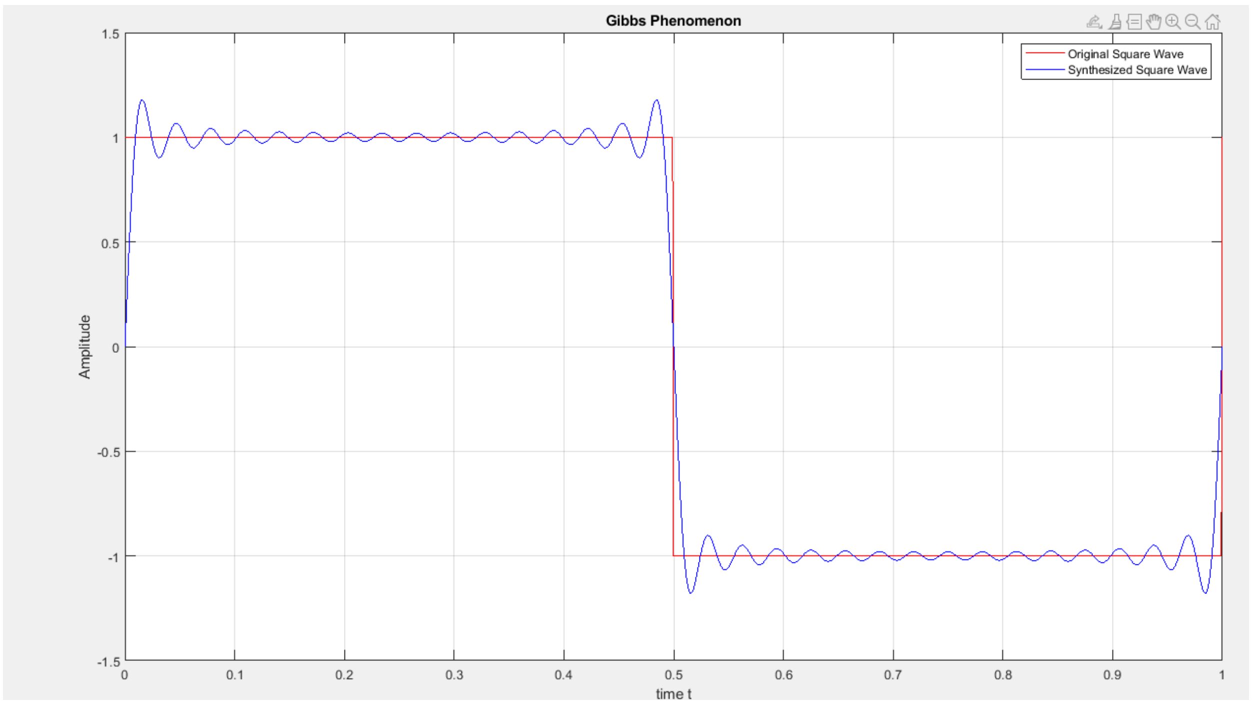
Program:
```
%% Exp 7b:
% Illustration of Gibb's phenomenon

clc
close all
clear all

%N=32;
N=input('type the number of harmonics: ');

%to generate original square wave
t=0:0.001:1;
y=square(2*pi*t);

%to plot the original square wave signal with red line
plot(t,y,'r');
grid on
axis([0 1 -1.5 1.5]);
hold on;
```

```
%generate a vector of zeros to initialize the square wave
sq=zeros(size(t));

%generate odd values beginning with 1 and incremented by 2
for n=1:2:N
      %equation for synthesized square wave
      sq=sq+(4/(pi*n)*sin(2*pi*n*t));
end

%to plot synthesized square wave
plot(t,sq,'b');
hold off;
xlabel('time t');
ylabel('Amplitude');
legend('Original Square Wave','Synthesized Square Wave');
title('Gibbs Phenomenon');
```

Output:

```
type the number of harmonics: 32
```

Result: A square signal using Fourier coefficients reconstructed and observed the Gibb's phenomenon using MATLAB.



Lab No.: 2 Exp No.: 2a Verification of Linear Convolution – Causal Sequences

AIM: Write a program in MATLAB to perform linear convolution of two causal sequences. Verify the same on MATLAB.
Toolboxes: Basic Mathematics and Graphics Toolboxes of MATLAB
List of commands / functions: input( ), length( ), conv( ), disp( ), pause, subplot( ), stem( ), title( ), xlabel( ), ylabel( ), 'colon'(;) operator
Theory:

PROCEDURE:

1. MATLABs in-built 'conv' function can calculate the response of a causal system to the causal input.
2. The response of the system i.e., the result of the convolution is also causal and finding its time axis is straight forward as 0 to convolution length-1.
3. Where convolution length = length of first sequence+ length of second sequence-1.
4. Enter the sample values within square brackets, each sample separated by a ',' or 'space', when it prompts.

PROGRAM:

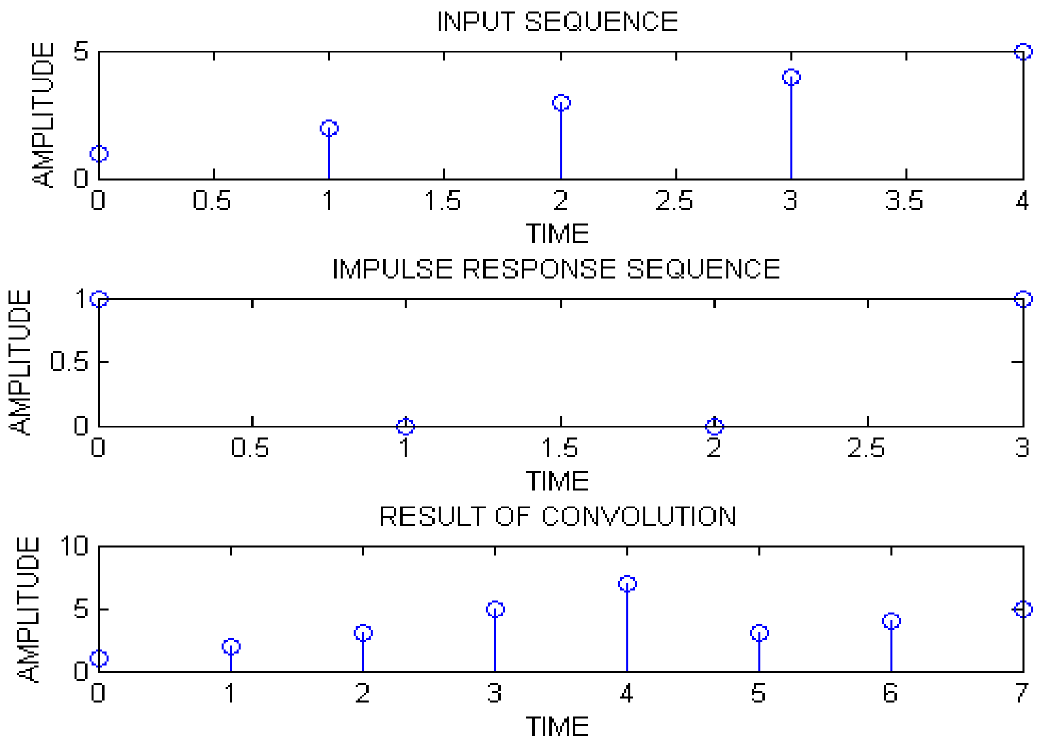
```
%% Exp 2a:
% Program to perform Linear Convolution of input two causal
Sequences by using MATLAB
clear all
close all
clc

x=input('ENTER THE INPUT SEQUENCE: ');
%calculate the length of the sequence x
n1=length(x);

h=input('ENTER THE IMPULSE RESPONSE SEQUENCE: ');
%calculate the length of the sequence h
n2=length(h);

%calculate the length of the convolution of two sequences
n=n1+n2-1;

% compute the convolution of the two input sequences
y=conv(x,h);
disp(['CONVOLUTION OF x AND h IS: [' num2str(y) ']'])

disp('PRESS ENTER FOR INPUT SEQUENCE');
pause
```

```
% k indicates duration of the corresponding sequences
k=0:1:n1-1;
subplot(3,1,1),stem(k,x);
title('INPUT SEQUENCE');
xlabel('TIME');ylabel('AMPLITUDE');
disp('PRESS ENTER FOR IMPULSE RESPONSE SEQUENCE');
pause
k=0:1:n2-1;
subplot(3,1,2),stem(k,h);
title('IMPULSE RESPONSE SEQUENCE');
xlabel('TIME');ylabel('AMPLITUDE');

disp('PRESS ENTER FOR OUTPUT SEQUENCE');
pause
k=0:1:n-1;
subplot(3,1,3),stem(k,y);
title('RESULT OF CONVOLUTION');
xlabel('TIME');ylabel('AMPLITUDE');
```

Output:

```
ENTER THE INPUT SEQUENCE: [1 2 3 4 5]
ENTER THE IMPULSE RESPONSE SEQUENCE: [1 0 0 1]
CONVOLUTION OF x AND h IS: [1 2 3 5 7 3 4 5]
PRESS ENTER FOR INPUT SEQUENCE
PRESS ENTER FOR IMPULSE RESPONSE SEQUENCE
PRESS ENTER FOR OUTPUT SEQUENCE
```

Result: Linear convolution of two causal sequences performed using MATLAB.



Lab No.: 2 Exp No.: 2b Verification of Linear Convolution – Non-Causal Sequences

AIM: Write a program in MATLAB to perform linear convolution of two input sequences in which one or both of them are non-causal. Verify the same on MATLAB.
Toolboxes: Basic Mathematics and Graphics Toolboxes of MATLAB
List of commands / functions: input( ), length( ), conv( ), disp( ), pause, subplot( ), stem( ), title( ), xlabel( ), ylabel( ), 'colon'(;) operator
Theory:

PROCEDURE:

1. MATLABs in-built 'conv' function can calculate the response of a causal system to the causal input.
2. The result of the convolution is non-causal if one of the signals is non-causal.
3. The time axis of the resultant sequence is
   Starting point=starting point of first signal+ starting point of second signal
   Ending point=ending point of first signal+ ending point of second signal
   Finally, the time axis of the resulting signal is from 'starting point' to 'ending point'.
   Where convolution length = length of first sequence + length of second sequence-1.
4. Create two files one is script file and another is function file. Save the script file as "exp2b.m" and save the function file with the function name itself "conv_m.m".
5. Script file takes the input, calls the 'conv_m' function, and displays the input & output sequences.
6. 'conv_m' function perform actual convolution.
7. Run the script file.
8. Enter the sample values within square brackets, each sample separated by a ',' or 'space', when it prompts.

PROGRAM:

```
%Script file: exp2b.m
clear all
close all
clc

x=input('ENTER THE FIRST SEQUENCE: ');
%calculate the duration of the sequence x
nx1=input('ENTER THE STARTING POINT OF FIRST SEQUENCE: ');
nx2=input('ENTER THE ENDING POINT OF FIRST SEQUENCE: ');
nx=nx1:nx2;

h=input('ENTER THE SECOND SEQUENCE: ');
%calculate the duration of the sequence h
nh1=input('ENTER THE STARTING POINT OF SECOND SEQUENCE: ');
nh2=input('ENTER THE ENDING POINT OF SECOND SEQUENCE: ');
nh=nh1:nh2;
```

```
[y,ny]=conv_m(x,nx,h,nh);

disp('PRESS ENTER FOR FIRST SEQUENCE');
pause
subplot(3,1,1),stem(nx,x);
title('FIRST SEQUENCE');
xlabel('TIME');ylabel('AMPLITUDE');

disp('PRESS ENTER FOR SECOND SEQUENCE');
pause
subplot(3,1,2),stem(nh,h);
title('SECOND SEQUENCE');
xlabel('TIME');ylabel('AMPLITUDE');

disp('PRESS ENTER FOR OUTPUT SEQUENCE');
pause
subplot(3,1,3),stem(ny,y);
title('RESULT OF CONVOLUTION');
xlabel('TIME');ylabel('AMPLITUDE');
```

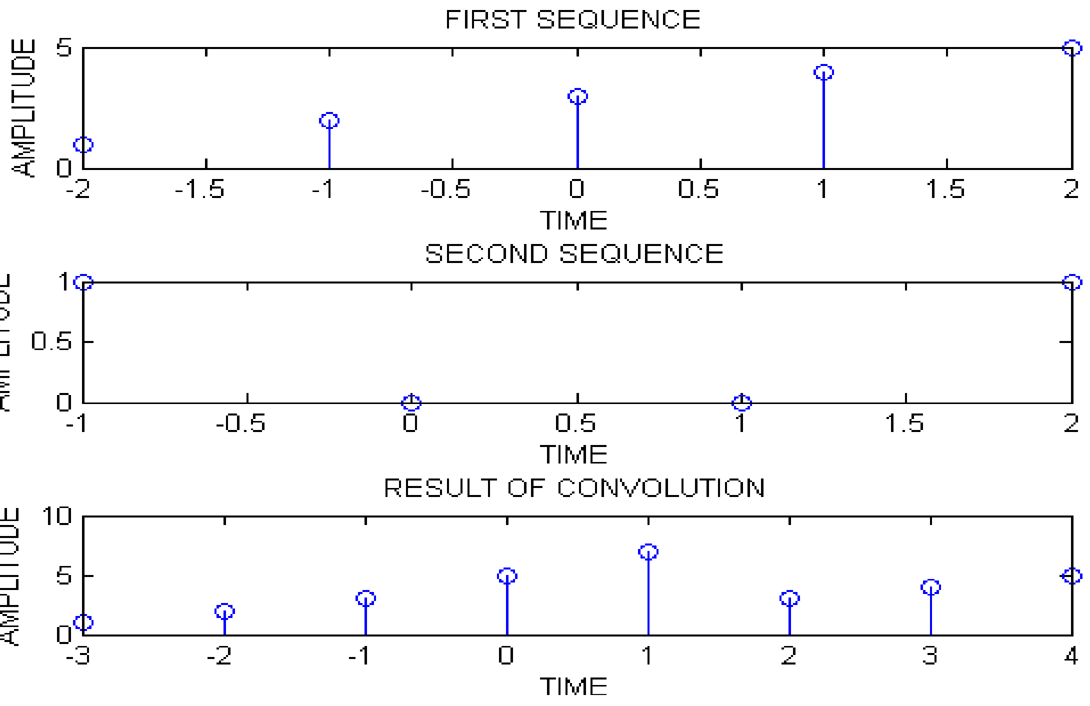
Function file: conv_m.m

```
function [y,ny]=conv_m(x,nx,h,nh)
nyb=nx(1)+nh(1);
nye=nx(length(x))+nh(length(h));
ny=nyb:nye;
y=conv(x,h);
```

Output:

```
ENTER THE FIRST SEQUENCE: [1 2 3 4 5]
ENTER THE STARTING POINT OF FIRST SEQUENCE: -2
ENTER THE ENDING POINT OF FIRST SEQUENCE: 2
ENTER THE SECOND SEQUENCE: [1 0 0 1]
ENTER THE STARTING POINT OF SECOND SEQUENCE: -1
ENTER THE ENDING POINT OF SECOND SEQUENCE: 2
PRESS ENTER FOR FIRST SEQUENCE
PRESS ENTER FOR SECOND SEQUENCE
PRESS ENTER FOR OUTPUT SEQUENCE
```

Result: Linear convolution of two input sequences in which one or both of them are non-causal performed using MATLAB.



VIVA -VOCE QUESTIONS:

1. What is convolution?
2. Write the expression for discrete convolution?
3. What is the need of convolution?
4. Why DFT does not support Linear Convolution?
5. How to obtain the response of discrete LTI system from Linear Convolution?
6. What is the length of the resultant sequence in linear convolution?
7. Why zero padding is used in linear convolution?
8. What are the four steps to find linear convolution?
9. Give the properties of linear convolution?
10. In MATLAB which command is used to plot the discrete sequence?
11. Which window is preferable to write MATLAB program?
12. What is num2str?
13. How does num2str work in Matlab?
14. What is the use of pause command?
15. What is conv function in Matlab?
16. What is the difference between size ( ) and length ( ) commands?

Lab No.:2 Exp No.: 2c Verification of Linear Convolution using Circular Convolution

Aim: Write a program in MATLAB to perform linear convolution of two sequences by using circular convolution. Verify the same on MATLAB.
Toolboxes: Basic Mathematics and Graphics Toolboxes of MATLAB
List of commands / functions: input( ), length( ), fft( ), ifft( ), pause, disp( ), subplot( ), stem( ), title( ), xlabel( ), ylabel( ), zeros( )
Theory:

Procedure:
MATLABs in built 'conv' function can calculate only linear convolution of two given sequences. It cannot be used for the calculation of circular convolution. From the Fourier transform properties we know that the convolution in one domain is equal to multiplication in another domain. If both the signals are continuous signals then this property is talking about linear convolution, if both the signals are discrete sequences then this property is talking about circular convolution.

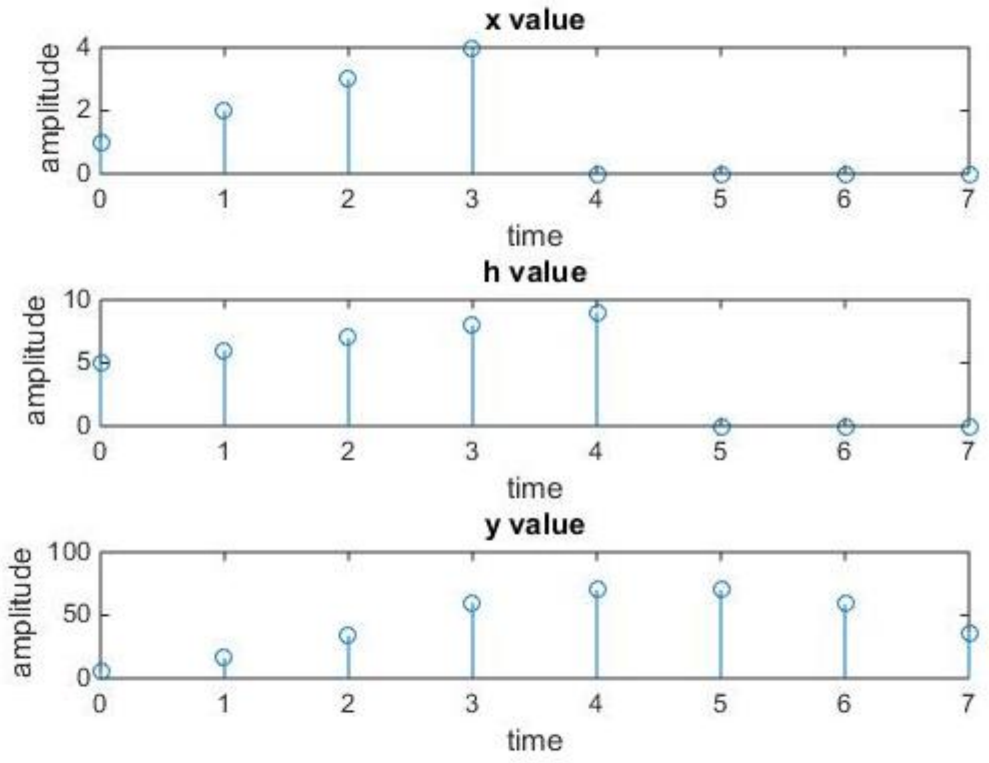
1. Pad appropriate number of zeros to the both sequences, so that both sequences become of equal length.
2. Take the FFT of both of them using the function 'fft'.
3. Multiply corresponding spectral components.
4. Take the inverse FFT using the function 'ifft'.
5. The resulting time domain sequence is the linear convolution of the given two sequences.

Program:
```
% Exp 2c:
clear all
close all
clc

x=input('enter the input sequence: ');
h=input('enter the impulse sequence: ');
n1=length(x);
n2=length(h);
l=n1+n2-1;
zpx=[x zeros(1,l-n1)];
disp('zpx values: ');
disp(zpx);
zph=[h zeros(1,l-n2)];
disp('zph values: ');
disp(zph);
x=fft(zpx);
h=fft(zph);
y=ifft(x.*h);
disp('y values: ');
disp(y);
disp('press enter for zero padding of x(n)');
pause
k=0:1:l-1;
subplot(3,1,1);
stem(k,zpx);
title('x value');
xlabel('time');
ylabel('amplitude');
disp('press enter for zero padding of h(n)');
pause
k=0:1:l-1;
subplot(3,1,2);
stem(k,zph);
title('h value');
xlabel('time');
ylabel('amplitude');
disp('press enter for response');
pause
k=0:1:l-1;
subplot(3,1,3);
stem(k,y);
title('y value');
xlabel('time');
ylabel('amplitude');
```

Output:

```
enter the input sequence: [1 2 3 4]
enter the impulse sequence: [5 6 7 8 9]
zpx values:
        1  2  3  4  0  0  0  0
zph values:
        5  6  7  8  9  0  0  0
y values :
        Columns 1 through 7
                5.0000   16.0000   34.0000   60.0000   70.0000   70.0000   59.0000
        Column 8
                36.0000
press enter for zero padding of x(n)
press enter for zero padding of h(n)
press enter for response
```

Result: The linear convolution of two sequences by using circular convolution computed and verified using MATLAB.



VIVA -VOCE QUESTIONS:

1. In what way Linear Convolution is different from Circular Convolution?
2. How will you perform linear convolution via circular convolution?
3. Why zero padding is required in circular convolution?
4. How to obtain the response of discrete LTI system from circular convolution?
5. How to obtain the circular convolution through DFT & IDFT?
6. Why DFT supports only circular convolution?
7. In MATLAB without using "conv" command, how to compute the convolution?
8. In MATLAB how to pad the given sequence, to obtain the required length?
9. What is the use of "clf" command in MATLAB?
10. What is the use of "pause" command in MATLAB?
11. Explain the command "subplot(a,b,c)"?
12. What do you mean by zero padding?

Lab No.:3 Exp No.: 3 Verification of Circular Convolution

Aim: Write a program in MATLAB to compute the response of a discrete LTI system with input sequence x(n) and impulse response h(n) by using circular convolution. Verify the same on MATLAB.

Toolboxes: Basic Mathematics and Graphics Toolboxes of MATLAB

List of commands / functions: input( ), length( ), max( ) fft( ), ifft( ), pause, disp( ), subplot( ), stem( ), title( ), xlabel( ), ylabel( ), zeros( )

Theory:

Procedure:

MATLABs in built 'conv' function can calculate only linear convolution of two given sequences. It cannot be used for the calculation of circular convolution. From the Fourier transform properties we know that the convolution in one domain is equal to multiplication in another domain. If both the signals are continuous signals then this property is talking about linear convolution, if both the signals are discrete sequences then this property is talking about circular convolution.

1. Pad appropriate number of zeros to the smaller length sequence, so that both sequences become of equal length.
2. Take the FFT of both of them using the function 'fft'.
3. Multiply corresponding spectral components.
4. Take the inverse FFT using the function 'ifft'.
5. The resulting time domain sequence is the circular convolution of the given two sequences.

Program:

```
% Exp 3:
clear all
close all
clc

x=input('enter the input sequence: ');
h=input('enter the impulse sequence: ');
N1=length(x);
N2=length(h);
N=max(N1,N2);
N3=N1-N2;
if(N3>0)
   h=[h,zeros(1,N3)];
else
   x=[x,zeros(1,-N3)];
end
disp('x values: ');
disp(x);
disp('h values: ');
disp(h);
```

```
X=fft(x);
H=fft(h);
y=ifft(X.*H);
disp('y values: ');
disp(y);
disp('press enter for x(n)');
pause
k=0:1:N-1;
subplot(3,1,1);
stem(k,x);
title('x value');
xlabel('time');
ylabel('amplitude');
disp('press enter for h(n)');
pause
k=0:1:N-1;
subplot(3,1,2);
stem(k,h);
title('h value');
xlabel('time');
ylabel('amplitude');
disp('press enter for response');
pause
k=0:1:N-1;
subplot(3,1,3);
stem(k,y);
title('y value');
xlabel('time');
ylabel('amplitude');
```

Output:

```
enter the input sequence: [1 2 3 4 5 6]
enter the impulse sequence: [6 5 4 3 2]
x values:
        1  2  3  4  5  6
h values:
        6  5  4  3  2  0
y values :
        74       64       60       62       70       90
press enter for x(n)
press enter for h(n)
press enter for response
```

Result: The response of a discrete LTI system with input sequence x(n) and impulse response h(n) by using circular convolution computed and verified using MATLAB.

Lab No.: 4 Exp No.: 10 Verification of Fast Fourier Transform (FFT)

Aim: Write a program in MATLAB to compute the Fast Fourier Transform of a given discrete sequence x(n). Verify the same on MATLAB.
Toolboxes: Basic Mathematics and Graphics Toolboxes of MATLAB
List of commands / functions: input( ), length( ), fft( ), abs( ), angle( ), stem( ), pause, subplot( ), title( ), xlabel( ), ylabel( ), disp( )
Theory:

Procedure:
The DFT is used to convert N-point discrete time sequence x(n) to an N-point frequency domain sequence X(k).

$$X(k) = \sum_{n=0}^{N-1} x(n)\, W_N^{nk}$$

where k is ranging from 0 to N – 1.
To compute DFT from above formulae is very difficult, if number of points (N) increases, to overcome this problem we can for Fast Fourier Transform (FFT). The FFT is a method or algorithm is used to compute DFT with reduced number of calculations (Complex additions & Complex multiplications).

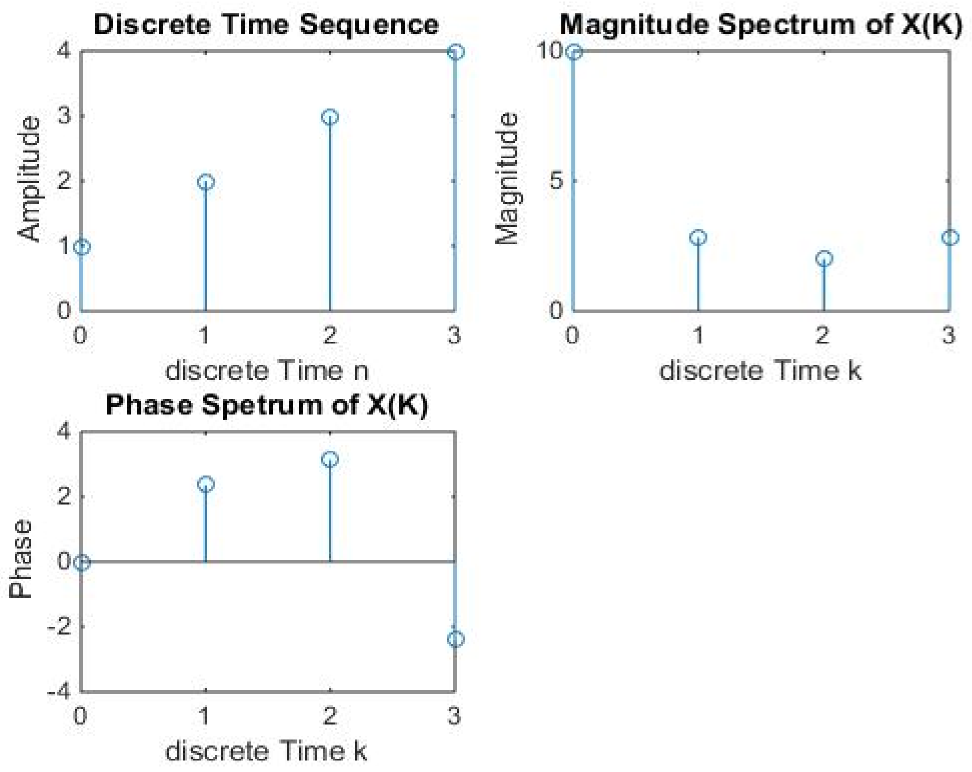
Program:
```
%% Exp 10:
clear all
close all
clc

x=input('Enter the discrete sequence x(n)::');
N=length(x);
X=fft(x);
MSX=abs(X);
PSX=angle(X);
disp('Discrete time sequence x(n)::');
disp(x);
disp('DFT of x(n)= X(K)is');
disp(X);
disp('Magnitude Spectrum of X(K)is ');
disp(MSX);
disp('Phase Spectrum of X(K)is ');
disp(PSX);
disp('press "ENTER" for given sequence');
pause
n=0:1:N-1;
```

```
subplot(2,2,1);
stem(n,x);
title('Discrete Time Sequence');
xlabel('discrete Time n');
ylabel('Amplitude');
disp('For Magnitude Spectrum press "ENTER"');
pause
k=0:1:N-1;
subplot(2,2,2);
stem(k,MSX);
title('Magnitude Spectrum of X(K)');
xlabel('discrete Time k');
ylabel('Magnitude');
disp('For Phase Spectrum press "ENTER"');
pause
k=0:1:N-1;
subplot(2,2,3);
stem(k,PSX);
title('Phase Spetrum of X(K)');
xlabel('discrete Time k');
ylabel('Phase');
```

Output:

```
Enter the discrete sequence x(n):: [1, 2, 3, 4]
Discrete time sequence x(n)::
        1   2   3   4
DFT of x(n)= X(K)is
        10.0000+0.0000i   -2.0000+2.0000i   -2.0000+0.0000i   -2.0000-2.0000i
Magnitude Spectrum of X(K)is
        10.0000   2.8284   2.0000   2.8284
 Phase Spectrum of X(K)is
        0   2.3562   3.1416   -2.3562
press "ENTER" for given sequence
For Magnitude Spectrum press "ENTER"
For Phase Spectrum press "ENTER"
```

Result: The Fast Fourier Transform of a given discrete sequence x(n) computed and verified using MATLAB.



VIVA -VOCE QUESTIONS:

1. Define transform. What is the need for transform?
2. Differentiate DFT and DTFT?
3. How to obtain Discrete Fourier Transform of a sequence x(n)?
4. What is Fast Fourier Transform?
5. What are different types of FFT algorithms?
6. Explain clearly how computational efficiency increases through FFT?
7. How to compute IDFT through FFT?
8. How to obtain response of the system from FFT?
9. What is the command used to compute DFT in MATLAB?
10. What is meant by magnitude plot, phase plot, power spectrum?
11. Which command is used to compute Magnitude Spectrum in MATLAB?
12. Which command is used to compute Phase Spectrum in MATLAB?
13. What is the use of ABS ( )?
14. What is the angle function?
15. What does angle ( ) do in MATLAB?

Lab No.: 5 Exp No.: 6 N-point FFT algorithm

Aim: Write a program in MATLAB to compute the N1 & N2-point FFT and display the magnitude spectrum of the given sequence.
Toolboxes: Basic Mathematics and Graphics Toolboxes of MATLAB
List of commands / functions: input( ), length( ), fft( ), abs( ), ones( ), zeros( ), nextpow2( ), linspace( ), figure, stem( ), subplot( ), title( ), xlabel( ), ylabel( ), num2str( )
Theory:

Procedure:
The DFT is used to convert N-point discrete time sequence x(n) to an N-point frequency domain sequence X(k).

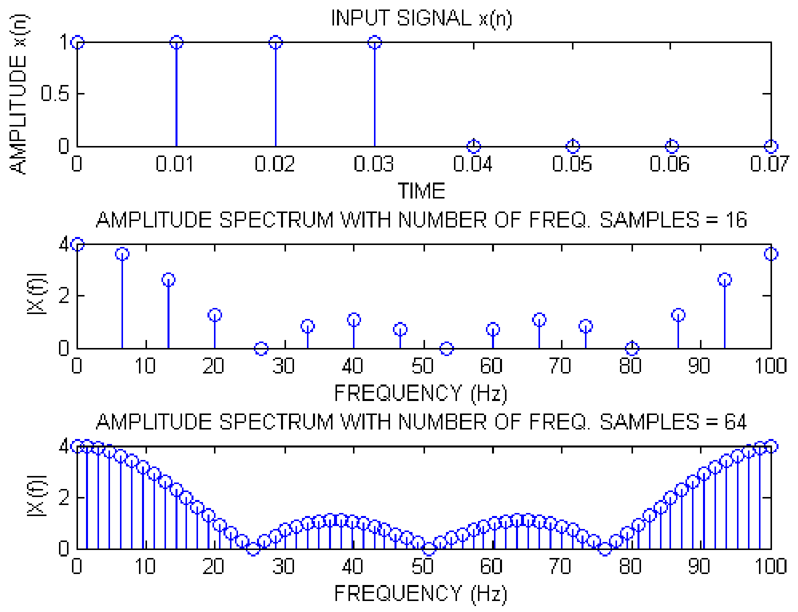
$$X(k) = \sum_{n=0}^{N-1} x(n)\, W_N^{nk}$$

where k is ranging from 0 to N – 1.
To compute DFT from above formulae is very difficult, if number of points (N) increases, to overcome this problem we can for Fast Fourier Transform (FFT). The FFT is a method or algorithm is used to compute DFT with reduced number of calculations (Complex additions & Complex multiplications).

Program:
```
%% Exp 6:
clear all
close all
clc

%Set the sampling frequency
fs=100;

%Generate a time domain sequence with N=8 number of samples
%x=[1,0,1,0,1,0,1,0];
x=[ones(1,4),zeros(1,4)];
%x=input('Enter the discrete sequence x(n)::');
L=length(x);

% Compute the time axis for the sequence
n=(0:L-1)./fs;
N=2^nextpow2(L);
%Set the number of frequency domain samples (N1)that are required
N1=2*N;

% Compute the N1-point FFT
```

```
X1=fft(x,N1);
% Compute the frequency axis for the N1-point FFT
f1=linspace(0,1,N1)*fs;

%Compute magnitude response
magX1=abs(X1(1:N1));
figure,subplot(311),stem(n,x)
title('INPUT SIGNAL x(n)')
xlabel('TIME'),ylabel('AMPLITUDE x(n)')
subplot(312),stem(f1,magX1)
title(['AMPLITUDE SPECTRUM WITH NUMBER OF FREQ.SAMPLES= ' num2str(N1)])
xlabel('FREQUENCY(Hz)')
ylabel('|X(f)|')

%Set the number of frequency domain samples (N2)that are required
N2=4*N1;

%ComputetheN2-pointFFT
X2=fft(x,N2);

% Compute the frequency axis for the N1-point FFT
f2=linspace(0,1,N2)*fs;

%Compute magnitude response
magX2=abs(X2(1:N2));
subplot(313),stem(f2,magX2)
title(['AMPLITUDE SPECTRUM WITH NUMBER OF FREQ.SAMPLES= ' num2str(N2)])
xlabel('FREQUENCY(Hz)')
ylabel('|X(f)|')
```

Result: The N1 & N2-point FFT computed and the magnitude spectrum displayed of the given sequence using MATLAB.



Lab No.: 6 Exp No.: 9 Power Density Spectrum

Aim: Write a program in MATLAB to compute power density spectrum of a given sequence.
Toolboxes: Basic Mathematics and Graphics Toolboxes of MATLAB
List of commands / functions: sin( ), length( ), fft( ), abs( ), figure, plot( ), subplot( ), title( ), xlabel( ), ylabel( ), grid on
Theory:

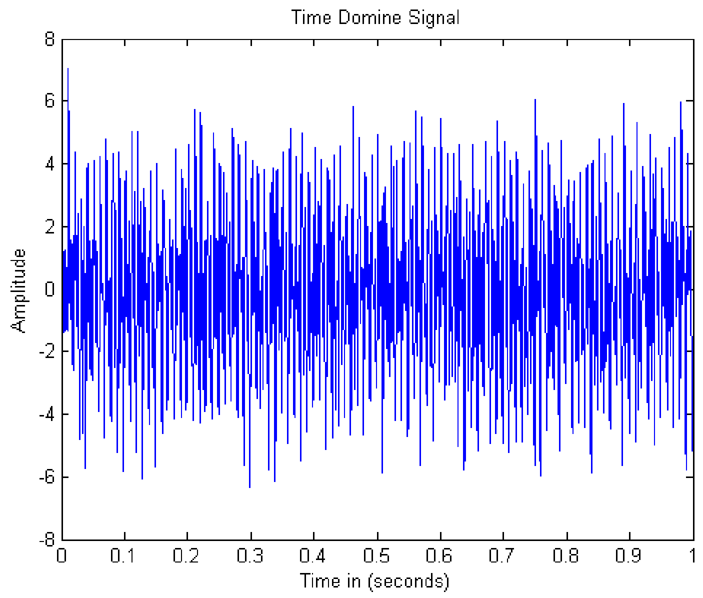
Procedure:

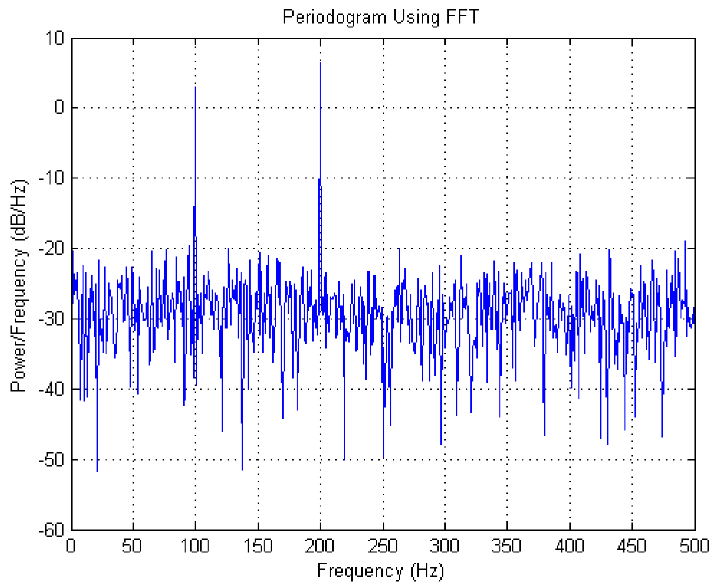
1. Create a signal consisting of sum of a 100 Hz, 200Hz sine waves in N (0, 1) additive noise. The sampling frequency is 1 kHz. The signal length is 1000 samples. Use the default settings of the random number generator for reproducible results.
2. Obtain the periodogram. Compute DFT using 'fft'. Normalize its magnitude square with the product of sampling frequency and the length of the sequence.
3. The signal is real-valued and has even length. Because the signal is real-valued, you only need power estimates for the positive or negative frequencies.
4. In order to conserve the total power, multiply all frequencies that occur in both sets -- the positive and negative frequencies -- by a factor of 2. Zero frequency (DC) and the Nyquist frequency do not occur twice.
5. Plot the result.

Program:
```
%% Exp 9:
clear all
close all
clc

rng default
Fs = 1000;
t = 0:1/Fs:1-1/Fs;

x=2*sin(2*pi*100*t)+3*sin(2*pi*200*t)+ randn(size(t));

N = length(x);
xdft = fft(x);
xdft = xdft(1:N/2+1);
psdx = (1/(Fs*N)) * abs(xdft).^2;
psdx(2:end-1) = 2*psdx(2:end-1);
freq = 0:Fs/length(x):Fs/2;

plot(t,x)
title('Time Domain Signal')
xlabel('Time in (seconds)')
ylabel('Amplitude')
```

```
figure;
plot(freq,10*log10(psdx))
grid on
title('Periodogram Using FFT')
xlabel('Frequency (Hz)')
ylabel('Power/Frequency (dB/Hz)')
```

Result: The power density spectrum of the given sequence computed using MATLAB.





Lab No.:07 Exp No.: 8 Frequency Response of Analog LP/HP Filter

Aim: Write a program in MATLAB to obtain the frequency response of Analog LP/HP filter for the given specifications cutoff frequency and capacitance.
Toolboxes: Basic Mathematics and Graphics Toolboxes of MATLAB
List of commands / functions: sqrt( ), log10( ), semilogx( ), floor( ), find( ), max( ), subplot( ), title( ), xlabel( ), ylabel( ), grid on
Theory:

Procedure:

1. Determine the values of R&C required, obtaining given cut-off frequency of the filter. Assume C value first, based on that determine R value according to the formula

$$\Omega_c=1/(RC)$$

2. The transfer function of analog single pole low pass filter is

$$H(s)=1/(1+sRC)=1/(1+j\Omega RC)$$

3. The transfer function of analog single pole low pass filter is

$$H(s)=sRC/(1+sRC)= j\Omega RC /(1+j\Omega RC)$$

4. Now select various frequencies i.e., along the imaginary axis of s-plane, for which magnitude response is to be calculated.

5. Compute the magnitude response and plot in the decibel scale.

Program:

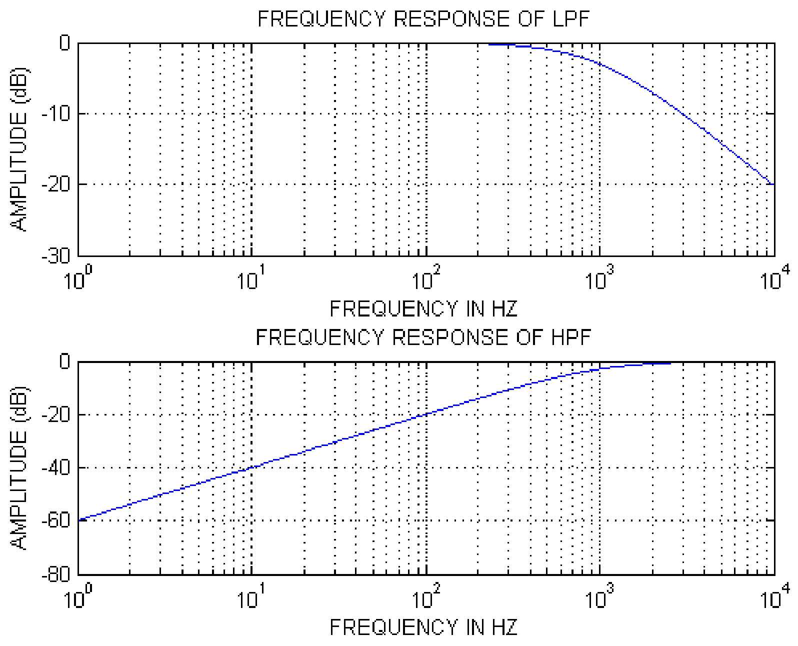
```
%% Lab 07:
%DESIGN OF ANALOG LP/HP FILTER
clear all
close all
clc

disp('-----INPUT DATA FOR SIMPLE RC ANALOG LPF    ');
disp('');
cutoff_freq = input('Enter the cutoff frequency (in KHz)of the LPF:');
cutoff_freq = 1000*cutoff_freq;
disp('');

%%Choose capacitance as 0.1 micro farads
C= input('Choose the value of capacitor(in micro faradays):');
C= C*10^-6;
disp('');
```

```
%Calculating the value of resistance required........
R = 1/(2*pi*cutoff_freq*C);


%Specify the frequency range for obtaining frequency response....
%SELECT FREQUENCIES IN S-PLANE WHERE MAGNITUDE RESPNOSE IS TO BE
%CALICULATED
freq_range = 0:1:10^4;
omega = 2.*pi.* freq_range;

% For LPF H(s) = 1/(1+sRC)
wRC = omega*R*C;
char_fun = sqrt(1+wRC.^2);
MSofLPF = char_fun.^-1;
MSofLPFinDB = 20.*log10(MSofLPF);

%For HPFH(s) = sRC/(1+sRC)
MSofHPF = wRC.*(char_fun.^-1);
MSofHPFinDB = 20.*log10(MSofHPF);

subplot(2,1,1);
semilogx(freq_range,MSofLPFinDB);
title('FREQUENCY RESPONSE OF LPF');
xlabel('FREQUENCY IN HZ');
ylabel('AMPLITUDE (dB)'),
grid on
subplot(2,1,2);
semilogx(freq_range,MSofHPFinDB);
title('FREQUENCY RESPONSE OF HPF');
xlabel('FREQUENCY IN HZ');
ylabel('AMPLITUDE(dB)'),
grid on


%Verification of the cutoff frequency of the designed filter

cutoff_freq_of_disg_in_rad=omega(max(find(floor(MSofLPFinDB)==-3))+1);
cutoff_freq_of_disg_in_KHz = cutoff_freq_of_disg_in_rad/(2*pi*1000);

disp(['The desired cutoff frequency of the filter is:'...
num2str(cutoff_freq/1000)'KHz']);
disp(['The cutoff frequency of the designed filter is:' ...
num2str(cutoff_freq_of_disg_in_KHz)'KHz']);
```

Output:

```
The desired cutoff frequency of the filter is : 1 KHz
The cutoff frequency of the designed filter is : 0.998 KHz
```

Result: The frequency response of Analog LP/HP filter for the given specifications cutoff frequency and capacitance using MATLAB.



VIVA -VOCE QUESTIONS:

1. Define filter?
2. Explain different types of filters?
3. Compare analog / digital filters?
4. Explain the operation of RC low pass filter?
5. Explain the operation of RC high pass filter?

Lab No.:08 Exp No.: 5a Frequency Response of IIR low pass Butterworth Filter

Aim: Write a program MATLAB to design IIR BUTTERWORTH low pass filter for the given specifications such as order of the filter (N) and cutoff frequency (rad/sec).
Toolboxes: Basic Mathematics and Graphics Toolboxes of MATLAB
List of commands / functions: butter( ), bilinear( ), format long, log10( ), abs( ), tf( ), disp( ), freqz( ), semilogx( ), figure, zplane( ), subplot( ), title( ), xlabel( ), ylabel( ), grid on
Theory:

Procedure:

1. A digital IIR filter is designed by first designing analog prototype filter and then converting that analog filter into digital filter by using one of the, ***bilinear*** or ***impulse invariance***, transformation techniques.
2. **Design of Analog Filter:** Analog Butterworth filter can be designed using MATLAB in-built functions ***'butter'***. Give the specifications, such as order of the filter (N) and cutoff frequency (rad/sec), of Lowpass Butterworth filter. The function ***'butter'*** computes the numerator & denominator coefficients and store them into output vectors ***'b'*** and ***'a'***.
3. **Analog to Digital mapping:** It requires the sampling frequency ***(Fs)*** with which the mapping is to be done. The MATLAB functions that perform these mapping are ***'bilinear'*** (Bilinear Transformation) and ***'impinvar'***(Impulse Invariance Transformation). Call ***'bilinear'*** function with ***'b', 'a',*** and ***'Fs'*** as input arguments.
4. **'bilinear'** function computes the numerator and denominator coefficients of the corresponding digital filter and store them into output vectors ***'numd'*** and ***'denmd'*** .
5. Finally display magnitude and phase spectrum of the resultant digital filter and observe its cutoff frequency, transition bandwidth and etc.

Program:

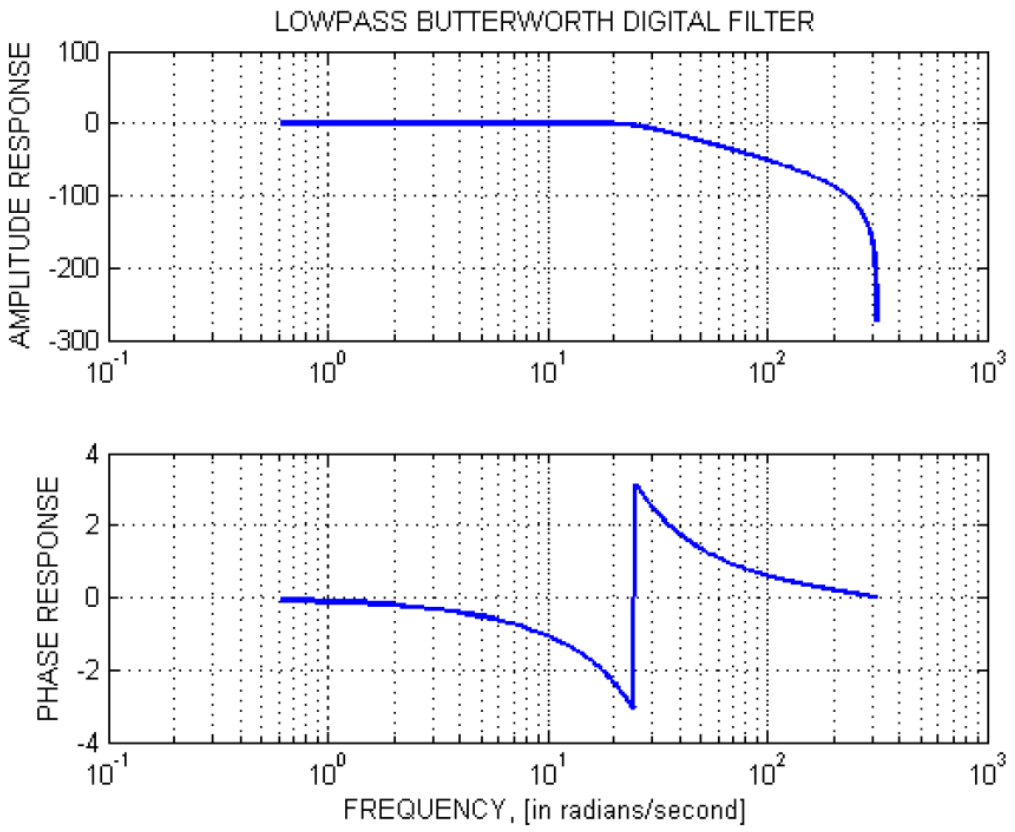
```
%% Lab 08:
%DESIGN OF AN DIGITAL BUTTERWORTH IIR LOWPASS FILTER TO MEET
%THE GIVEN SPECIFICATIONS
clear all
close all
clc
N=input('ENTER THE ORDER OF THE FILTER: ');
fc=input('ENTER THE CUTOFF FREQUENCY(in Hz): ');
wc=2*pi*fc;
disp('------DESIGNING OF DIGITAL FILTER STARTED------')
% DESIGN OF ANALOG FILTER WITH THE GIVEN SPECIFICATIONS
[b,a]=butter(N,wc,'s');
% SAME SAMPLING FREQUENCY THAT IS USED FOR THE CONSTRUCTION OF SQUARE
% WAVE IS USED FOR THE DESIGN OF DIGITAL FILTER ALSO
% ANALOG TO DIGITAL MAPPING
Fsamp=input('ENTER THE SAMPLING FREQUENCY(in Hz): ');
```

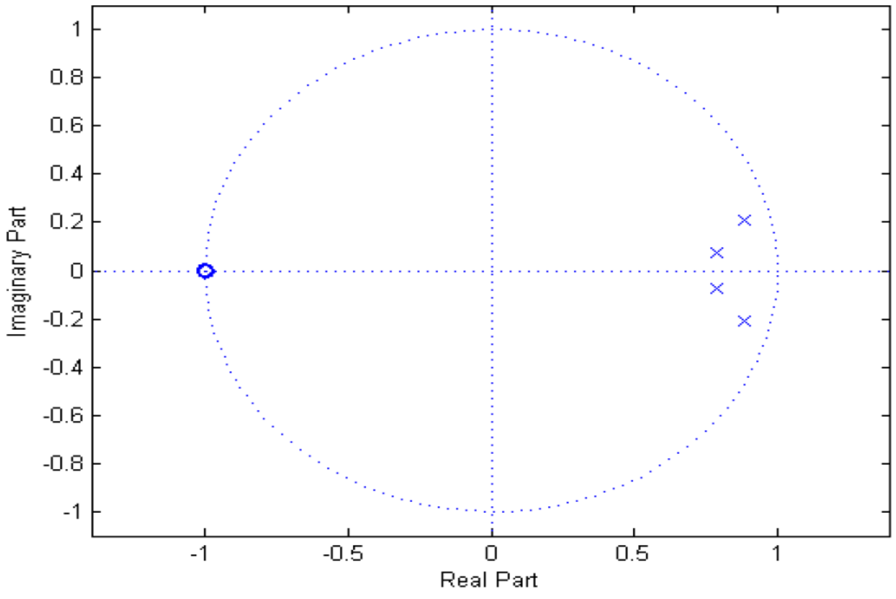
```
[numd,dend]=bilinear(b,a,Fsamp);
format long
% PRINT NUMERATOR COEFFICIENTS IN COLUMN VECTOR
disp('numd=');
disp(numd')
% PRINT DENOMINATOR COEFFICIENTS IN COLUMN VECTOR
disp('dend=');
disp(dend')
% DETERMINE THE FREQUENCY RESPONSE
[Hd,F]=freqz(numd,dend,512,Fsamp);
% COMPUTE MAGNITUDE RESPONSE OF DIGITAL FILTER
magd=20.*log10(abs(Hd));
% PLOT THE MAGNITUDE RESPONSE
figure, subplot(2,1,1);
semilogx(2*pi*F,magd,'LineWidth',2);
grid on;
title('LOWPASS BUTTERWORTH DIGITAL FILTER');
ylabel('AMPLITUDE RESPONSE');
subplot(2,1,2);
semilogx(2*pi*F,angle(Hd),'LineWidth',2);
grid on
ylabel('PHASE RESPONSE');
xlabel('FREQUENCY, [in radians/second]');
figure,zplane(numd,dend)
% Disply the Tranfer function Hd(z) in 'z' domain
trans_function = tf(numd,dend,0.1,'Variable','z^-1')
```

Output:

```
ENTER THE ORDER OF THE FILTER:
ENTER THE CUTOFF FREQUENCY(in Hz):
------DESIGNING OF DIGITAL FILTER STARTED------
ENTER THE SAMPLING FREQUENCY(in Hz):
numd=
dend=
```

Result: Designed IIR BUTTERWORTH low pass filter for the given specifications such as order of the filter (N) and cutoff frequency (Hz) using MATLAB.





VIVA -VOCE QUESTIONS:

1. Define IIR filter?
2. Explain different types of IIR filters?
3. Compare IIR / FIR filters?
4. Compare BUTTERWORTH / CHEBYSHEV filter?
5. What are conditions for an analog filter to be stable and causal?
6. What are conditions for a digital filter to be stable and causal?
7. What are two techniques used to obtain digital filter transfer function from analog filter transfer function?
8. What is frequency warping?
9. What is the relation between analog and digital frequency of IIT method?
10. What is the relation between analog and digital frequency of BLT method?
11. What is the format in MATLAB?
12. What does format long mean in MATLAB?
13. What is hold on and grid on in MATLAB?
14. What is the use of ABS ()?
15. What is butter function in MATLAB?

Lab No.:08 Exp No.: 5b Frequency Response of IIR high pass Butterworth Filter

Aim: Write a program MATLAB to design IIR BUTTERWORTH high pass filter for the given specifications pass band frequency, stop band frequency, pass band ripple and stop band ripple.
Toolboxes: Basic Mathematics and Graphics Toolboxes of MATLAB
List of commands / functions: input( ), buttord( ), butter( ), log10( ), angle( ), disp( ), freqz( ), subplot( ), plot( ), title( ), xlabel( ), ylabel( ), grid on
Theory:

Procedure:
1. Calculate the minimum order (N) and cut-off frequency (wc) required to meet the given specifications using MATLABs in-built function 'buttord'.
2. Analog Butterworth filter can be designed using MATLAB in-built functions ***'butter'***. Give the specifications, such as order of the filter (N) and cutoff frequency (rad/sec), of Lowpass Butterworth filter. The function ***'butter'*** computes the numerator & denominator coefficients and store them into output vectors ***'b'*** and ***'a'***.
3. **'bilinear'** function computes the numerator and denominator coefficients of the corresponding digital filter and store them into output vectors ***'numd'*** and ***'denmd'*** .
4. Calculate and plot its magnitude and phase response using functions 'abs' and 'angle'.

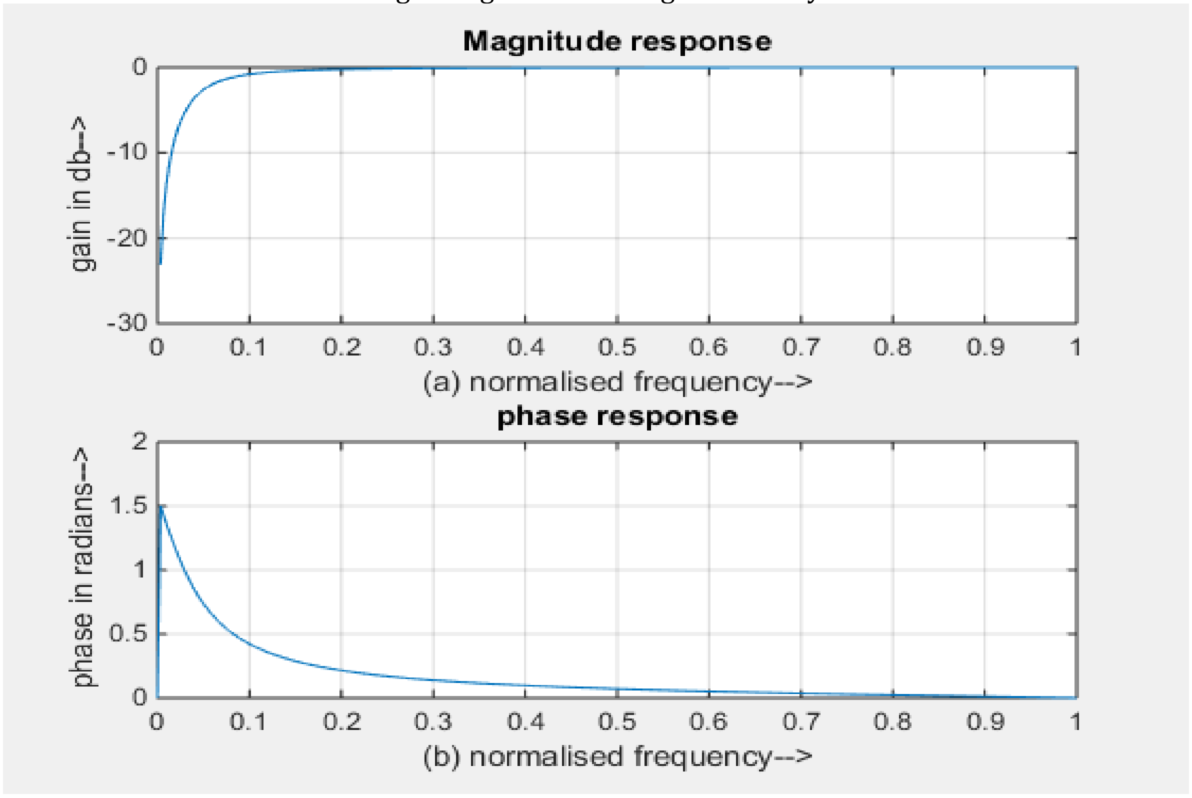
Program:
```
%% Lab 08:
%DESIGN OF AN DIGITAL BUTTERWORTH IIR LOWPASS FILTER TO MEET
%THE GIVEN SPECIFICATIONS
clear all
close all
clc
alphap=input('enter the pass band ripple:  ');
alphas=input('enter the stop band ripple:  ');
fp=input('enter the pass band frequency:  ');
fs=input('enter the stop band frequency:  ');
f=input('enter the sampling frequency:  ');
omp=2*pi*fp/f;
oms=2*pi*fs/f;
[N,wc]=buttord(omp,oms,alphap,alphas);
disp('Order of the filter is:  ');
disp(N);
[b,a]=butter(N,wc,'high');
[numd,dend]=bilinear(b,a,f);
format long
% PRINT NUMERATOR COEFFICIENTS IN COLUMN VECTOR
disp('numd=');
disp(numd')
```

```
% PRINT DENOMINATOR COEFFICIENTS IN COLUMN VECTOR
disp('dend=');
disp(dend')
% DETERMINE THE FREQUENCY RESPONSE
[Hd,F]=freqz(numd,dend,512,f);
mag=20.*log10(abs(Hd));
an=angle(Hd);
disp('press enter for magnitude response');
subplot(2,1,1);
semilogx(2*pi*F,mag,'LineWidth',2);
grid on
title('Magnitude response');
xlabel('(a) Frequency, [in radians/second]-->');
ylabel('Gain in db-->');
disp('press enter for phase response');
subplot(2,1,2);
semilogx(2*pi*F,an,'LineWidth',2);
grid on
title('Phase response');
xlabel('(b) Frequency, [in radians/second]-->');
ylabel('Phase in radians-->');

figure,zplane(numd,dend)
% Disply the Tranfer function Hd(z) in 'z' domain
trans_function = tf(numd,dend,0.1,'Variable','z^-1')
```

Output:

```
Enter the pass band ripple:  0.6
Enter the stop band ripple:  0.8
Enter the pass band frequency:  100
Enter the stop band frequency:  50
Enter the sampling frequency:  1000
Order of the filter is:
Press enter for magnitude response
Press enter for phase response
```

Result: Designed IIR BUTTERWORTH high pass filter for the given specifications pass band frequency, stop band frequency, pass band ripple and stop band ripple using MATLAB.



VIVA -VOCE QUESTIONS:

1. Define IIR filter?
2. Explain different types of IIR filters?
3. Compare IIR / FIR filters?
4. Compare BUTTERWORTH / CHEBYSHEV filter?
5. What are conditions for an analog filter to be stable and causal?
6. What are conditions for a digital filter to be stable and causal?
7. What are two techniques used to obtain digital filter transfer function from analog filter transfer function?
8. What is frequency warping?
9. What is the relation between analog and digital frequency of IIT method?
10. What is the relation between analog and digital frequency of BLT method?
11. What is hold on and grid on in MATLAB?
12. What is the use of ABS ()?
13. What is butter function in MATLAB?
14. What is buttord function in MATLAB?

Lab No.: 09 Exp No.: 4a Frequency Response of FIR low pass/high pass Filter using Rectangle Window

Aim: Write a program in MATLAB to design FIR Low pass/High pass filter for a given specifications using Rectangular Window.
Toolboxes: Basic Mathematics and Graphics Toolboxes of MATLAB
List of commands / functions: input( ), log10( ), sqrt( ), ceil( ), rem( ), boxcar( ), fir1( ), freqz( ), subplot( ), plot ( ), title( ), ylabel( ), xlabel( ), figure( ),display( ), disp( ).
Theory:

Procedure:
1) Enter the passband ripple (rp) and stopband ripple (rs). Enter the passband frequency (fp) and stopband frequency (fs). Enter the sampling frequency (f).

2) Calculate the analog passband edge frequency (wp) and stop band edge frequency (ws)

```
wp=2*pi*fp/f
ws=2*pi*fs/f
```

3) Calculate the order of the filter using the following formula,

$$n = \frac{(-20log_{10}(rp.rs) - 13)}{14.6(fs - fp)/f}$$

[Use 'ceil( )' for rounding off the value of 'n' to the nearest integer]
if 'n' is an odd number, then reduce its value by '1'.

4) Generate (n+1)th point window coefficients. For example boxcar(n+1) generates a rectangular window.

```
y=boxcar(n+1)
```

5) Design an nth order FIR filter using the previously generated (n+1) length window function.

```
b=fir1(n,wp,y)
```

6) Find the frequency response of the filter by using 'freqz( )' function.

```
[h,o]=freqz(b,a,k)
```

This function returns k-point complex frequency response vector 'h' and k-point frequency vector 'o' in radians/samples of the filter.

$$H(e^{j\omega}) = \frac{B(e^{j\omega})}{A(e^{j\omega})} = \frac{b(1) + b(2)e^{-j\omega} + \cdots + b(m+1)e^{-jm\omega}}{a(1) + a(2)e^{-j\omega} + \cdots + a(n+1)e^{-jn\omega}}$$

Where a, b are vectors containing the denominator and numerator coefficients. Here a(1)=1.

7) Calculate the magnitude of the frequency response in decibels (dB).

m= 20*$\log_{10}$(abs(h))

8) Plot the magnitude response [magnitude in dB Vs normalized frequency (o/pi)]. Give relevant names to x- and y- axes and give an appropriate title for the plot.

Repeat the program for different types of windows available:

a. Rectangular window
b. Triangular window
c. Kaiser window

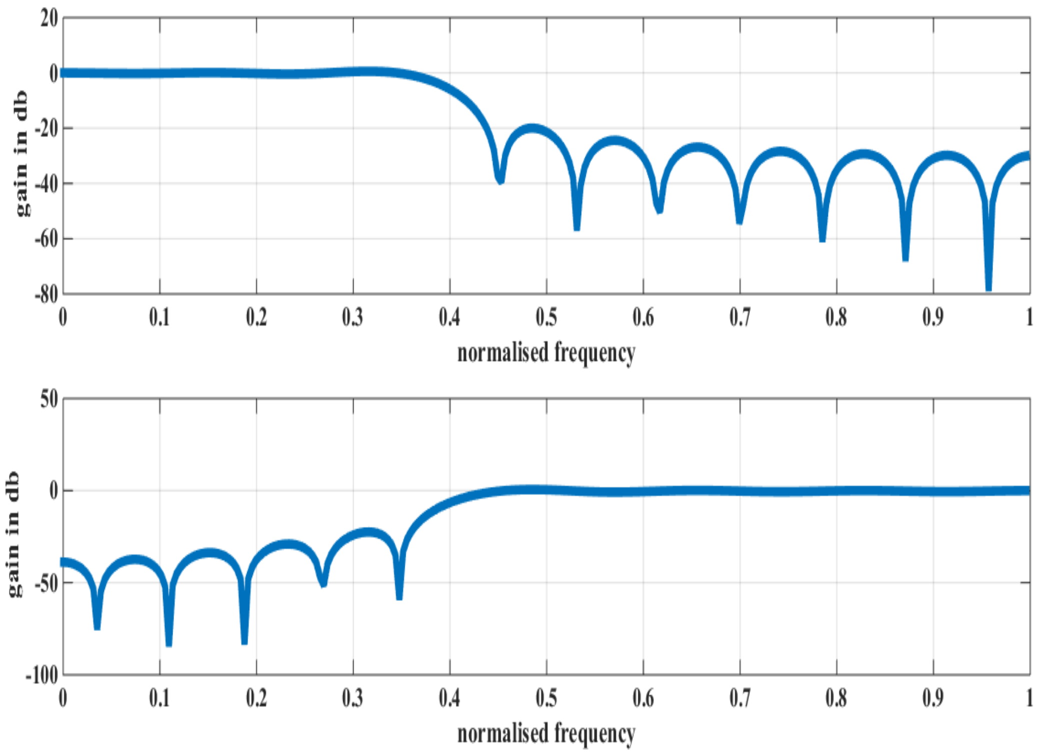
Program:

```
%% Lab 09:
clear all
close all
clc
rp=input('enter the passband ripple');
rs=input('enter  the stopband  ripple');
fp=input('enter  the  passband  frequency');
fs=input('enter the stopband  frequency');
f=input('enter the sampling frequency');
wp=2*pi*fp/f;
ws=2*pi*fs/f;
num=-20*log10(sqrt(rp*rs))-13;
dem=14.6*(fs-fp)/f;
n=ceil(num/dem);
n1=n+1;
if(rem(n,2)~=0)
  n1=n;
  n=n-1;
end
display('Order of the filter is ');
disp(n1);
y=boxcar(n1);
% LOW-PASS FILTER
b=fir1(n,wp,y);
[h,o]=freqz(b,1,256);
m=20*log10(abs(h));
subplot(2,1,1);
plot(o/pi,m);
xlabel('normalised frequency');
ylabel('gain in db');
% HIGH-PASS FILTER
b=fir1(n,wp,'high',y);
[h,o]=freqz(b,1,256);
m=20*log10(abs(h));
subplot(2,1,2);
plot(o/pi,m);
xlabel('normalised frequency');
ylabel('gain in db');
```

Output:

```
Enter the passband ripple 0.04
Enter the stopband ripple 0.01
Enter the passband frequency 1000
Enter the stopband frequency 2500
Enter the sampling frequency 9000
Order of the filter is
```

Result: Designed FIR Low pass/High pass filter for a given specifications using Rectangular Window using MATLAB.



VIVA -VOCE QUESTIONS:

1. Define FIR filter?
2. What are advantages of FIR Filter over IIR Filter?
3. Explain GIBBS Phenomenon?
4. What is the need of Window technique?
5. Which are raised cosine windows?
6. Which one is Barlett window?
7. Which one is Barlett window?
8. How to design FIR Filters?
9. What are characteristics of LPF?
10. What is Ceil function?

Lab No.: 09 Exp No.: 4b Frequency Response of FIR low pass/ high pass Filter using Triangle Window

Aim: Write a program in MATLAB to design FIR Low pass/ High pass filter for a given specifications using Triangle Window.

Toolboxes: Basic Mathematics and Graphics Toolboxes of MATLAB

List of commands / functions: input( ), log10( ), sqrt( ), ceil( ), rem( ), triang( ), fir1( ), freqz( ), subplot( ), plot ( ), title( ), ylabel( ), xlabel( ), figure( ),display( ), disp( ).

Theory:

Program:

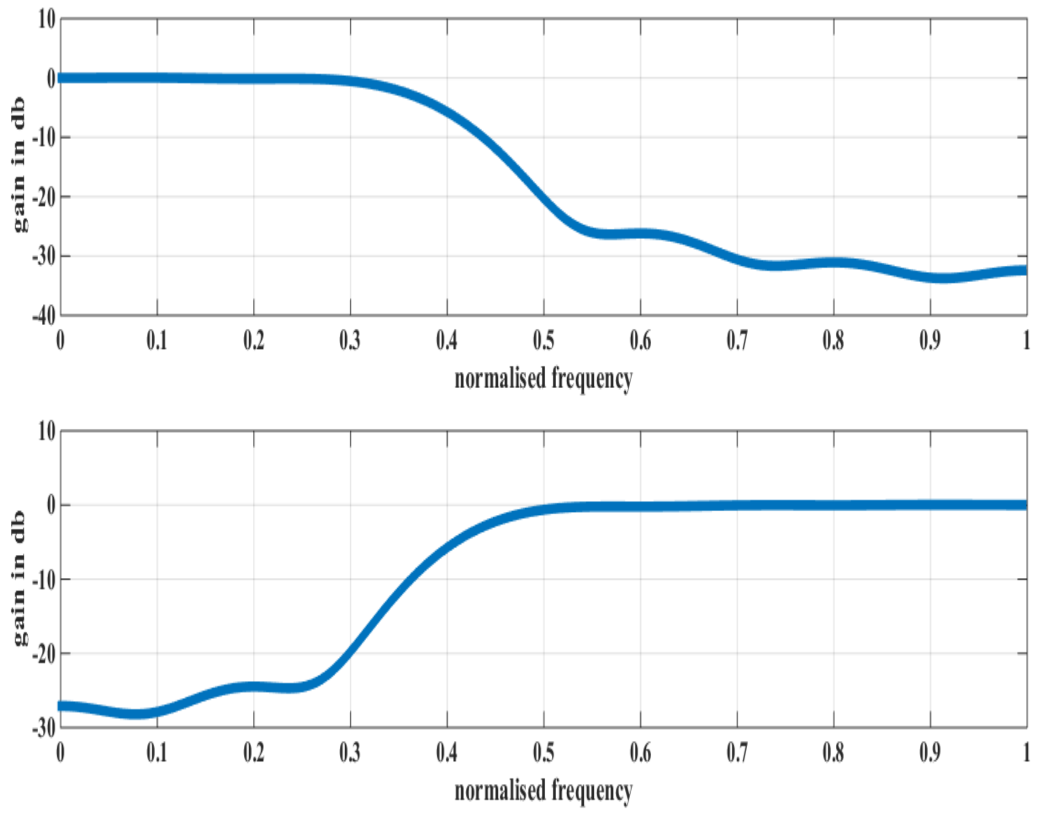
```
%% Lab 09:
clear all
close all
clc
rp=input('enter the passband ripple');
rs=input('enter  the stopband  ripple');
fp=input('enter  the  passband  frequency');
fs=input('enter the stopband  frequency');
f=input('enter the sampling frequency');
wp=2*pi*fp/f;
ws=2*pi*fs/f;
num=-20*log10(sqrt(rp*rs))-13;
dem=14.6*(fs-fp)/f;
n=ceil(num/dem);
n1=n+1;
if(rem(n,2)~=0)
  n1=n;
  n=n-1;
end
display('Order of the filter is ');
disp(n1);
y=triang(n1);
% LOW-PASS FILTER
b=fir1(n,wp,y);
[h,o]=freqz(b,1,256);
m=20*log10(abs(h));
subplot(2,1,1);
plot(o/pi,m);
xlabel('normalised frequency');
ylabel('gain in db');
% HIGH-PASS FILTER
b=fir1(n,wp,'high',y);
[h,o]=freqz(b,1,256);
m=20*log10(abs(h));
subplot(2,1,2);
plot(o/pi,m);
```

```
xlabel('normalised frequency');
ylabel('gain in db');
```

Output:

```
Enter the passband ripple 0.04
Enter the stopband ripple 0.01
Enter the passband frequency 1000
Enter the stopband frequency 2500
Enter the sampling frequency 9000
Order of the filter is
```

Result: Designed FIR Low pass/High pass filter for a given specifications using Triangular Window using MATLAB.


gain in db
normalised frequency
gain in db
normalised frequency