DIGITAL SIGNAL PROCESSING LAB

EC 352 R20

III/IV B.TECH – I SEM, R-20

LABORATORY MANUAL

Prepared By

B.Siva Kumari , M.Tech

Assistant Professor, Dept. of ECE, BWEC

Department of Electronics and Communications Engineering

Bapatla Women's Engineering College

(Affiliated to Acharya Nagarjuna University,Guntur)

Bapatla-522101

BAPATLA WOMEN'S ENGINEERING COLLEGE::BAPATLA

AN ISO 9001-2015 CERTIFIED INSTITUTION

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

INSTITUTE VISION AND MISSION

Vision

Bapatla Women's Engineering College has been established with forethought to organize the growing needs of higher technical education for WOMEN, in order to provide them a competative edge and a unique niche in the society.

Mission

Our mission is to impart the quality education at par with global standards to the students from all over India and in particular those from the local and rural areas.We continuously try to maintain high standards so as to make them technologically competent and ethically strong individuals who shall be able to improve the quality of life and economy of our country.

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

VISION

To produce highly professional and socially responsible women engineers through world class quality technical education and research to meet industry and Society needs in the field of Electronics and Communication Engineering.

MISSION

M1- To inculcate quality education for women through innovative learning practices with the use of modern tools.

M2- To impart professional ethics and moral values through co-curricular andextra-curricular activities as individual and team member.

M3- Empowering graduates through industrial trainings/on-job training in collaboration with industries with-a-view to life-long learning.

Programme Educational Objectives (PEO's)

PEO1: To develop and implementation of Electronics and Communication systems by imparting technical knowledge, professional leadership and communication skills as an individual and team in a multi-disciplinary stream.

PEO2: To train Graduates in solving complex problems, analyzing, designing and imparting research-based knowledge by using modern scientific tools to develop least cost-effective and innovative products.

PEO3: To produce women graduates through life-long learning, who ensure ethical and moral behavior.

Programme Specific Outcomes(PSO's)

PSO1: To understand and analyze basic concepts of electronics and communication in various areas like signal processing, VLSI, embedded systems, communications, digital and analog devices.

PSO2: Ability to design and develop computing systems using concepts of mathematics, electronics engineering and other related inter disciplines to asses and resolve societal, environmental, health safety and cultural issues with professional ethics.

Programme Outcomes (POs)

- **PO1: Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
- **PO2: Problem analysis:** Identify, formulate, review research literature, and analyse complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
- **PO3: Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
- **PO4: Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
- **PO5: Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools prediction and modeling to complex engineering activities with an understanding of the limitations.
- **PO6: The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice
- **PO7: Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and the need for sustainable development.
- **PO8: Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
- **PO9: Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
- **PO10: Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and receive clear instructions.

- **PO11: Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in team, to manage projects and in multidisciplinary environments.
- **PO12: Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

COURSE OBJECTIVES:

The course should enable the students

1. To Make familiar with the practical implementation of the Digital Signal Processing
2. To Use Computational tools to do basic Operations of Signal Processing.
3. To Develop DSP algorithms for convolution, correlation, DFT etc.
4. To implement FIR and IIR filters.
5. To Make familiar with the practical implementation of the Digital Signal Processing
6. To Use Computational tools to do basic Operations of Signal Processing

COURSE OUTCOMES:

On successful completion of the course, students will be able to

CO No.	Course Outcome Statement	Bloom's Taxonomy Level
C352.1	Generate the waveforms for Amplitude Modulation (AM) and Frequency Modulation (FM)	Apply
C352.2	Obtain the Frequency response for Low Pass Filter (LPF) and High Pass Filter (HPF)	Apply
C352.3	Obtain Fourier transform, convolution and correlation of two signals	Analyze
C352.4	Describe M-ary PSK	Analyze
C352.5	Evaluate DFT and IDFT of 16 sample sequence using DIT and DIF algorithm	Evaluate
C352.6	Design IIR Butter worth filter using Impulse Invariant Method	Create

Mapping of course outcomes with PO's and PSO's:

PO / CO	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11	PO 12	PSO 1	PSO 2	PSO3
C352.1	1	1	1	1	1	1	2	-	2	1	1	2	1	1	1
C352.2	1	1	1	1	1	1	2	-	2	1	1	2	1	1	1
C352.3	2	2	2	2	1	1	3	-	3	2	2	3	1	2	2
C352.4	1	1	1	1	1	1	2	-	2	1	1	2	1	1	1
C352.5	1	1	1	1	1	1	2	-	2	1	1	2	1	1	1
C352.6	2	2	-	-	-	-	1	-	1	-	-	1	-	-	-
Average	1.3	1.3	1	1	0.8	0.8	2	-	2	1	1	2	0.8	1	1

Correlation Levels:---1: Low 2: Medium 3: Strong, '-' No correlation

LIST OF EXPERIMENTS

1. Simulation of AM.
2. Simulation of FM
3. Simulation of LPF and HPF.
4. Fourier Transforms
5. Simulation of M-ary PSK.
6. Simulation of DPCM.
7. Evaluation of DFT and IDFT of 16 sample sequence using DIT Algorithm.
8. Evaluation of DFT and IDFT of 16 sample sequence using DIF Algorithm.
9. Design of IIR Butterworth Filter using Impulse Invariant Method.
10. Design of FIR Filter using Windowing Technique.
11. Convolution of Two signals.
12. Correlation of Two signals.
13. DFT Analysis of a Noise Corrupted Signal.

Experiment(s) beyond the syllabus

1. DESIGN OF CHEBYSHEV-I& CHEBYSHEV-II IIR FILTER

Exp.No.	Name of the Experiment	Page No.
1	Simulation of AM.	9
2	Simulation of FM	13
3	Simulation of LPF and HPF.	17
4	Fourier Transforms	23
5	Simulation of M-ary PSK.	27
6	Simulation of DPCM.	30
7	Evaluation of DFT and IDFT of 16 sample sequence using DIT Algorithm.	33
8	Evaluation of DFT and IDFT of 16 sample sequence using DIF Algorithm.	39
9	Design of IIR Butterworth Filter using Impulse Invariant Method.	45
10	Design of FIR Filter using Windowing Technique.	50
11	Convolution of Two signals.	58
12	Correlation of Two signals.	64
13	DFT Analysis of a Noise Corrupted Signal.	77
14	Design Of Chebyshev-I& Chebyshev-II IIR Filter	80

1. SIMULATION OF AM

Aim: To simulate amplitude modulation

Software:- MATLAB 7.11.0.584(R2010b) version

Theory:-

Modulation is defined as the process by which some characteristics of a carrier signal is varied in accordance with a modulating signal. The base band signal is referred to as the modulating signal and the output of the modulation process is called as the modulation signal.

Amplitude Modulation is defined as a process in which the amplitude of the carrier wave c(t) is varied linearly with the instantaneous amplitude of the message signal m(t).The standard form of an amplitude modulated (AM) wave is defined by

$$S(t) = [1 + K_a m(t) \cos(2\pi f_c t)]$$

Where Ka is a constant called the amplitude sensitivity of the modulator

The envelope of the modulating wave has the same shape as the base band signal provided the following two requirements are satisfied

1. The carrier frequency fc must be much greater then the highest frequency components fm of

the message signal m (t) i.e. fc >> fm

2. The modulation index must be less than unity. if the modulation index is greater than unity,

the carrier wave becomes over modulated.

Procedure:-

1. Click on the MATLAB icon, then **command window** will be opened.
2. Go to file and open new **Script**.
3. Save the program using specified name other than program name with **.m** extension.
4. **Debug** the program for errors.
5. If errors are present correct them.
6. **Run** the program.
7. Observe for resultant **graphs.**

Calculations:-

Message signal, $m=a_m\cos(2\pi f_m t)$

Carrier signal, $c= a_c\cos(2\pi f_c t)$

Amplitude modulated wave, $s=a_c(1+km)\cos(2\pi f_c t)$

$$s=c(1+km)$$

Modulation index, $k=a_m/a_c$

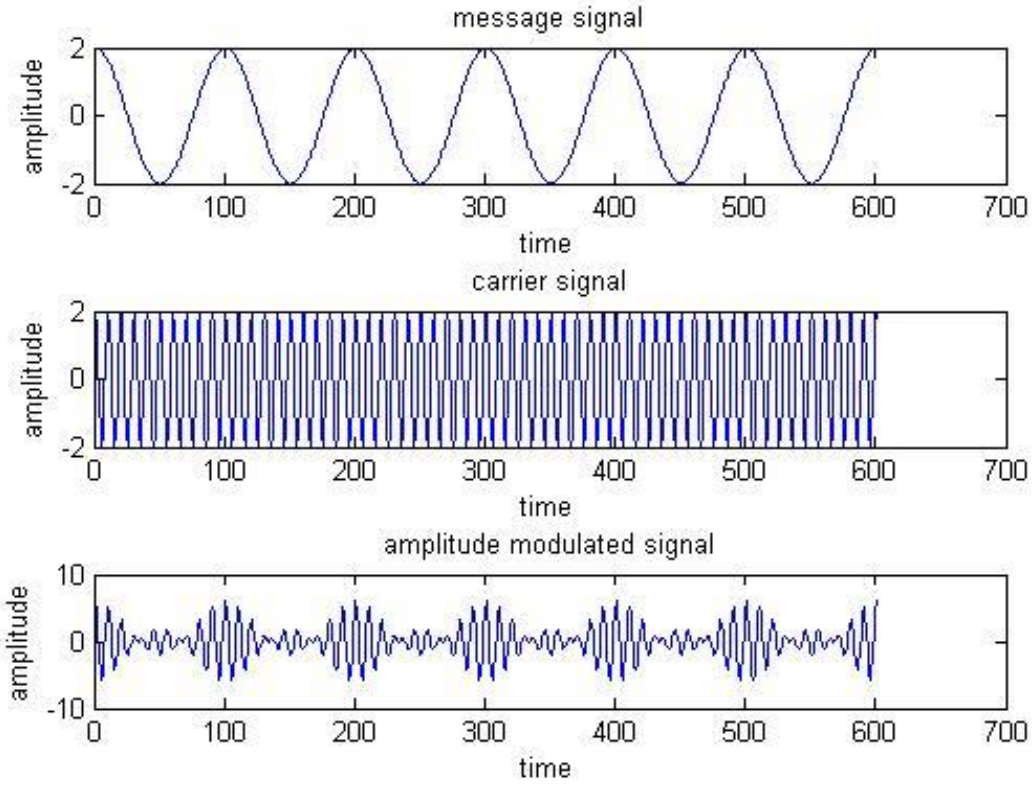
Program:-

```
am = input('enter the amplitude of message signal');
ac = input('enter the amplitude of carrier signal');
fm = input('enter the frequency  of message signal');
fc = input('enter the frequency of carrier signal');
n=input('enter no of cycles');
t = 1/fm;
t = 0:t/100:n*t;
m = am*cos(2*pi*fm*t);
c = ac*cos(2*pi*fc*t);
subplot(3,1,1);
```

```
plot(m);
xlabel('time');
ylabel('amplitude');
title('message signal');
subplot(3,1,2);
plot(c);
xlabel('time');
ylabel('amplitude');
title('carrier signal');
k=am/ac;
s=c.*(1+ka*m);
subplot(3,1,3);
plot(s);
xlabel('time');
ylabel('amplitude');
title('amplitude modulated signal');
```

Output:-

```
enter the amplitude of message signal 2
enter the amplitude of carrier signal 2
enter the frequency of message signal 100
enter the frequency of message signal 1000
enter no of cycles 6
```



Viva-Voce Questions:-

1) Define AM and what are its applications?
2) What is the need for modulation?
3) Define depth of modulation.
4) What are the degrees of modulation?
5) What are the types of AM modulators?

2.SIMULATION OF FM

Aim: To simulate frequency modulation

Software:- MATLAB 7.11.0.584(R2010b) version

Theory:-

Modulation is defined as the process by which some characteristics of a carrier signal is varied in accordance with a modulating signal. The base band signal is referred to as the modulating signal and the output of the modulation process is called as the modulation signal.

Frequency Modulation is defined as a process in which the frequency of the carrier wave c(t) is varied linearly with the instantaneous frequency of the message signal m(t).The standard form of an frequency modulated (FM) wave is defined by

$$s(t) = a_c \cos\left(2\pi f_c t + 2\pi k_f \int_0^t m(t)dt\right)$$

Where, m(t) = modulating signal

f_c = carrier frequency

c = carrier amplitude

If we let the modulating signal be a pure sinusoid, $m(t) = Am \cos 2\pi f_m t$, then equation becomes

$$s(t) = a_c \cos\left(2\pi f_c t + \frac{k_f a_m}{f_m} \sin 2\pi f_m t\right)$$

$$s(t) = a_c \cos\left(2\pi f_c t + \beta \sin 2\pi f_m t\right)$$

Where, $k_f a_m = \Delta f$ = frequency deviation

β = Modulation index

Procedure:-

1. Click on the MATLAB icon, then **command window** will be opened.
2. Go to file and open new **Script**.
3. Save the program using specified name other than program name with **.m** extension.
4. **Debug** the program for errors.
5. If errors are present correct them.
6. **Run** the program.
7. Observe for resultant **graphs**

Calculations:-

Message signal, $m=a_m\cos(2\pi f_m t)$

Carrier signal, $c= a_c\cos(2\pi f_c t)$

Frequency modulated signal, $s = a_c \cos(2\pi f_c t + \beta \sin 2\pi f_m t)$

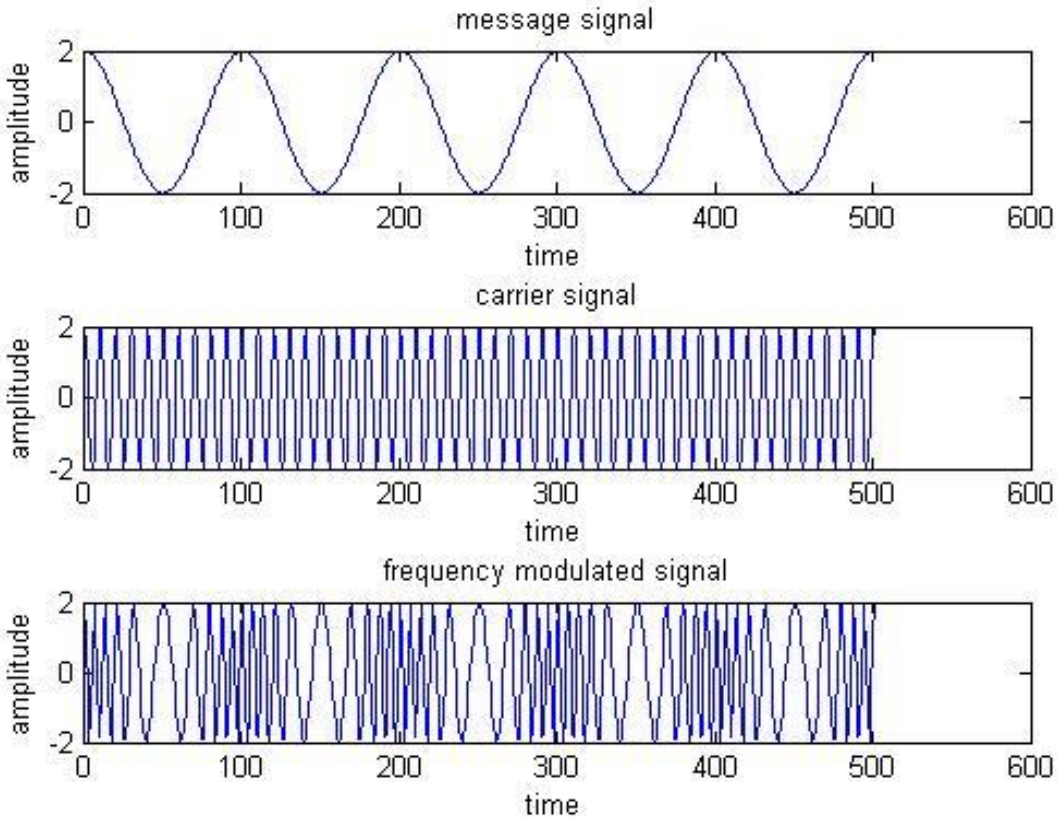
Program:-

```
am = input('enter the amplitude of message signal');
ac = input('enter the amplitude of carrier signal');
fm = input('enter the frequency  of message signal');
fc = input('enter the frequency of carrier signal');
kf = input('enter frequency sensitivity');
n=input('enter no of cycles');
t = 1/fm;
t = 0:t/100:n*t;
m = am*cos(2*pi*fm*t);
subplot(3,1,1);
plot(m);
```

```
xlabel('time');
ylabel('amplitude');
title('message signal');
c = ac*cos((2*pi*fc*t);
subplot(3,1,2);
plot(c);
xlabel('time');
ylabel('amplitude');
title('carrier signal');
fd=kr*am;
b=fd/fm;
s=ac.*cos((2*pi*fc*t)+b.*sin(2*pi*fm*t));
subplot(3,1,3);
plot(s);
xlabel('time');
ylabel('amplitude');
title('frequency modulated signal');
```

Output:-

```
enter the amplitude of message signal      2
enter the amplitude of carrier signal      2
enter the frequency  of message signal     100
enter the frequency  of message signal     1000
enter frequency sensitivity                300
enter no of cycles                         5
```



Viva-Voce Questions:-

1. What is frequency modulation?
2. Define modulation index.
3. What is frequency deviation?
4. Define demodulation.
5. Why FM is better than AM?

3.SIMULATION OF LPF AND HPF

3(a) Simulation of Low Pass Filter

Aim: To simulate Low Pass Filter.

Software:- MATLAB 7.11.0.584(R2010b) version

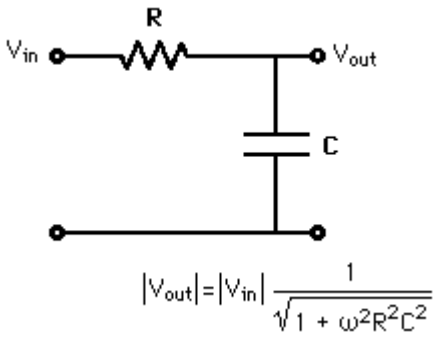
Theory:-

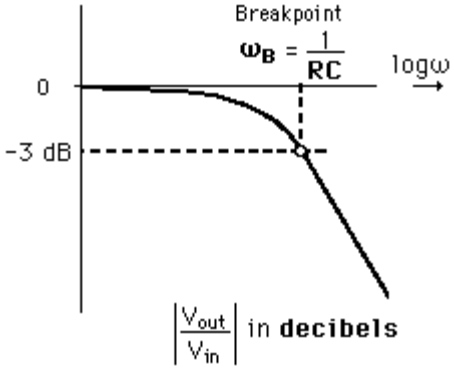
A filter is a device or process that removes from a signal some unwanted component or feature. A low-pass filter passes relatively low frequency components in the signal but stops the high frequency components. The so-called cutoff frequency divides the pass band and the stop band. In other words, the frequency components higher than the cutoff frequency will be stopped by a low-pass filter. This type of filter is especially useful since the random errors involved in the raw position data obtained through reconstruction are characterized by relatively high frequency contents.

Procedure:-

1. Click on the MATLAB icon, then **command window** will be opened.
2. Go to file and open new **Script**.
3. Save the program using specified name other than program name with **.m** extension.
4. **Debug** the program for errors.
5. If errors are present correct them.
6. **Run** the program.
7. Observe for resultant **graphs**

Calculations:-





Magnitude response = $\frac{1}{\sqrt{1+\left(\frac{f}{f_c}\right)^2}} = \frac{1}{1+j\left(\frac{f}{f_c}\right)}$

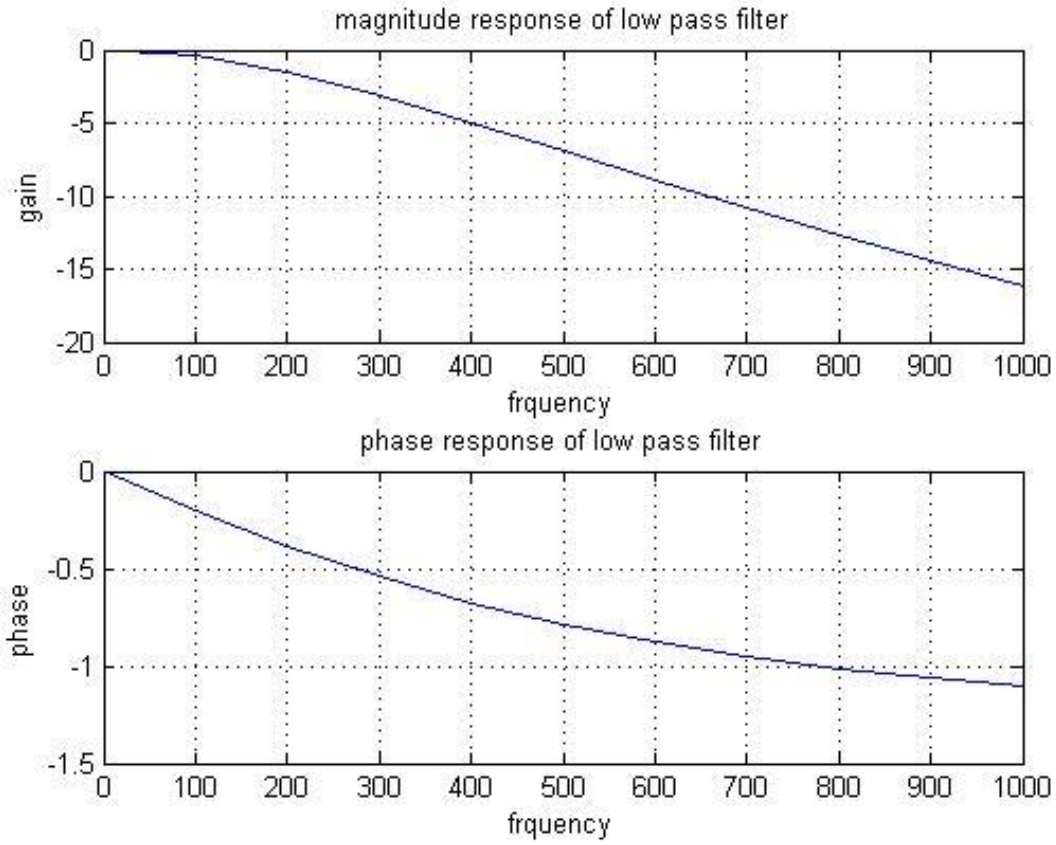
Phase response = $\tan^{-1}\left(\frac{f}{f_c}\right)$

Program:-

```
clc
clear all;
close all;
j=sqrt(-1);
f=0:100:1000;
fc=500;
a=1./(1+j*(f/fc));
h=abs(a);
g=20*log(h);
subplot(2,1,1);
plot(f,g);
```

```
grid on;
xlabel('frquency');
ylabel('gain');
title('magnitude response of low pass filter');
p=angle(a);
subplot(2,1,2);
plot(f,p);
xlabel('frquency');
ylabel('phase');
title('phase response of low pass filter');
grid on;
```

Output:-



3.(b) Simulation of High Pass Filter

Aim: To simulate High pass filter

Software:- MATLAB 7.11.0.584(R2010b) version

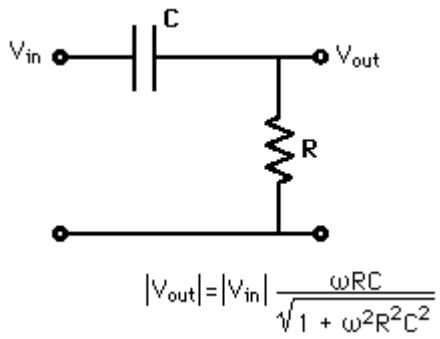
Theory:-

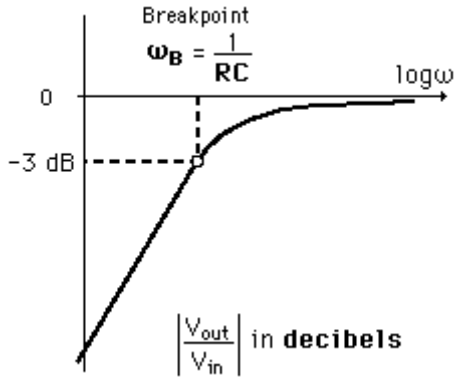
A filter is a device or process that removes from a signal some unwanted component or feature. A high-pass filter (HPF) is an electronic filter that passes high-frequency signals but attenuates (reduces the amplitude of) signals with frequencies lower than the cutoff frequency. The actual amount of attenuation for each frequency varies from filter to filter. A high-pass filter is usually modeled as a linear time-invariant system. It is sometimes called a low-cut filter or bass-cut filter. High-pass filters have many uses, such as blocking DC from circuitry sensitive to non-zero average voltages or RF devices. They can also be used in conjunction with a low-pass filter to make a band pass filter.

Procedure:-

1. Click on the MATLAB icon, then **command window** will be opened.
2. Go to file and open new **Script**.
3. Save the program using specified name other than program name with **.m** extension.
4. **Debug** the program for errors.
5. If errors are present correct them.
6. **Run** the program.
7. Observe for resultant **graphs**

Calculations:-





Magnitude response = $\frac{(f / fc)2}{\sqrt{1+(f / fc)2}}$

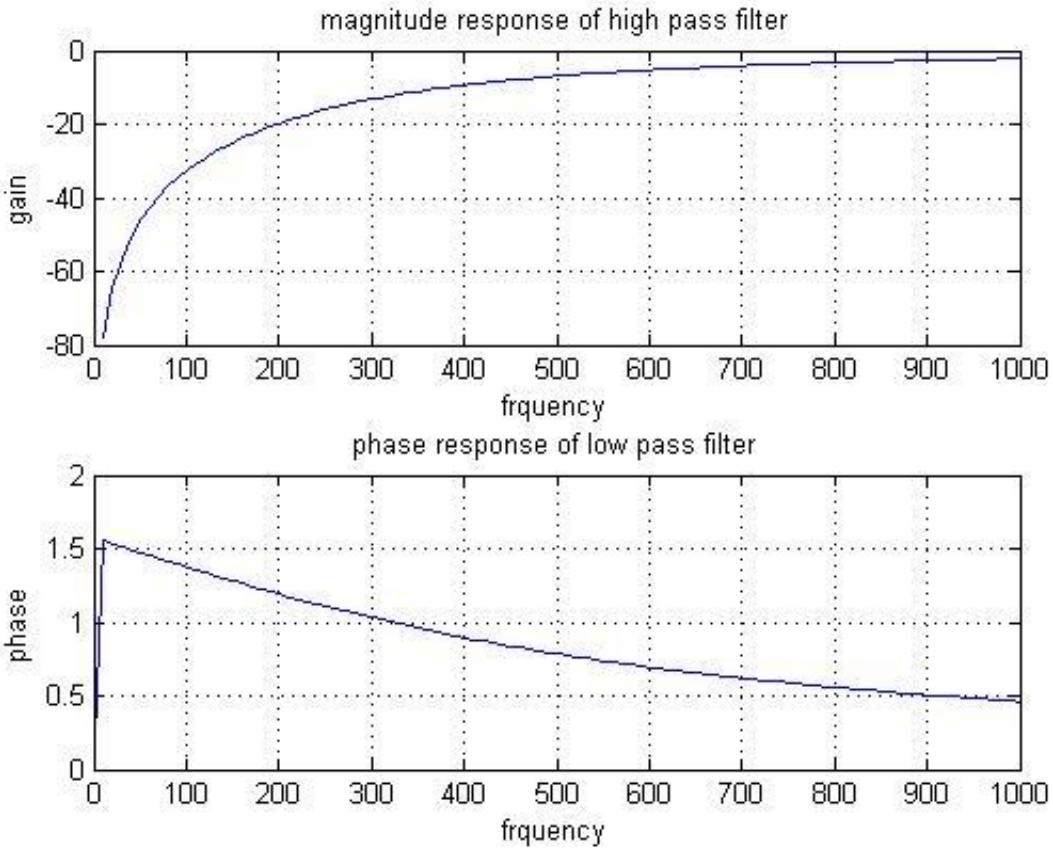
Phase response = $\tan^{-1}\left(\frac{f}{fc}\right)$

Program:-

```
clc
clear all;
close all;
j=sqrt(-1);
f=0:10:1000;
fc=500;
a=(j*(f/fc))./(1+j*(f/fc));
h=abs(a);
g=20*log(h);
subplot(2,1,1);
plot(f,g);
grid on;
xlabel('frquency');
```

```
ylabel('gain');
title('magnitude response of high pass filter');
p=angle(a);
subplot(2,1,2);
plot(f,p);
xlabel('frquency');
ylabel('phase');
title('phase response of low pass filter');
grid on;
```

Output:-



Viva-Voce Questions:-

1. Define filter.
2. List types of filter and define LPF & HPF.
3. Differentiate ideal filter and practical filter responses.
4. What is meant by magnitude response?
5. What is meant by phase response?

4. FOURIER TRANSFORMS

4.(a) Simulation of Discrete Fourier Transform

Aim: To simulate Discrete Fourier Transform

Software:- MATLAB 7.11.0.584 (R2010b) version

Theory:-

The **discrete Fourier transform** (**DFT**) converts a finite list of equally spaced samples of a function into the list of coefficients of a finite combination of complex sinusoids, ordered by their frequencies, that has those same sample values. It can be said to convert the sampled function from its original domain to the frequency domain.

The input samples are complex numbers (in practice, usually real numbers), and the output coefficients are complex as well. The frequencies of the output sinusoids are integer multiples of a fundamental frequency, whose corresponding period is the length of the sampling interval. The combination of sinusoids obtained through the DFT is therefore periodic with that same period. The DFT differs from the discrete-time Fourier transform (DTFT) in that its input and output sequences are both finite; it is therefore said to be the Fourier analysis of finite-domain discrete-time functions.

The DFT is the most important discrete transform, used to perform Fourier analysis in many practical applications. In digital signal processing, the function is any quantity or signal that varies over time, such as the pressure of a sound wave, a radio signal, or daily temperature readings, sampled over a finite time interval. In image processing, the samples can be the values of pixels along a row or column of a raster image. The DFT is also used to efficiently solve partial differential equations, and to perform other operations such as convolutions or multiplying large integers.

Procedure:-

1. Click on the MATLAB icon, then **command window** will be opened.
2. Go to file and open new **Script**.
3. Save the program using specified name other than program name with **.m** extension.
4. **Debug** the program for errors.
5. If errors are present correct them.
6. **Run** the program.
7. Observe for resultant **graphs**

Calculations:-

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{\frac{-j2\pi kn}{N}} \quad , 0 \leq k \leq N-1$$

Program:-

```
clc
clear all;
close all;
x=input('enter the sequence');
N=length(x);
j=sqrt(-1);
y=zeros(1,N);
for k=1:N
    for n=1:N
y(k)=y(k)+(x(n)*exp((-j*2*pi*(k-1)*(n-1))/N));
    end;
 end;
disp(y);
```

Output:-

```
enter the sequence[ 1 2 3 4]
  10.0000          -2.0000 + 2.0000i     -2.0000 - 0.0000i     -2.0000 - 2.0000i
```

4.(b) Simulation of Inverse Discrete Fourier Transform

Aim: To simulate Inverse Discrete Fourier Transform

Software:- MATLAB 7.11.0.584(R2010b) version

Theory:- The Fourier transform takes a signal in the so called time and maps it, without loss of information, into the frequency domain. The frequency domain representation is exactly the same signal, in a different form. The inverse Fourier transform maps the signal back from the frequency domain into the time domain.

A time domain signal will usually consist of a set of real values, where each value has an associated time (e.g., the signal consists of a time series). The Fourier transform maps the time series into a a frequency domain series, where each value is a complex number that is associated with a given frequency. The inverse Fourier transform takes the frequency series of complex values and maps them back into the original time series. Assuming that the original time series consisted of real values, the result of the IDFT will be complex numbers where the imaginary part is zero.

Procedure:-

1. Click on the MATLAB icon, then **command window** will be opened.
2. Go to file and open new **Script**.
3. Save the program using specified name other than program name with **.m** extension.
4. **Debug** the program for errors.
5. If errors are present correct them.
6. **Run** the program.
7. Observe for resultant **graphs**.

Calculations:

$$g(k) = \frac{1}{N}\sum_{n=0}^{N-1} x(n)e^{\frac{j2\pi kn}{N}} \quad ,0 \le k \le N-1$$

Program:-

```
clc
clear all;
close all;
x=input('enter the sequence');
N=length(x);
j=sqrt(-1);
y=zeros(1,N);
for k=1:N
   for n=1:N
      y(k)=y(k)+(x(n)*exp((j*2*pi*(k-1)*(n-1))/N));
   end;
end;
disp(y/N);
```

Output:-

```
Enter the sequence [10  -2+2i  -2-0i  -2-2i]
1.0000          2.0000 + 0.0000i      3.0000 - 0.0000i      4.0000 - 0.0000i
```

Viva-Voce Questions:-

1. What is signal?
2. Define transform. What is the need for transform?
3. Differentiate Fourier transform and discrete Fourier transform.
4. Explain the mathematical formula for calculation of DFT.
5. Explain the mathematical formula for calculation of IDFT.

5.SIMULATION OF M-ARY PSK

Aim: To simulate M-ary PSK

Software:- MATLAB 7.11.0.584(R2010b) version

Theory:- Phase-shift keying (PSK) is a digital modulation scheme that conveys data by changing, or modulating, the phase of a reference signal (the carrier wave). Any digital modulation scheme uses a finite number of distinct signals to represent digital data. PSK uses a finite number of phases, each assigned a unique pattern of binary digits. Usually, each phase encodes an equal number of bits. Each pattern of bits forms the symbol that is represented by the particular phase.

M-ary Phase Shift Keying or MPSK is a modulation where data bits select one of M phase shifted versions of the carrier to transmit the data. Thus, the M possible waveforms all have the same amplitude and frequency but different phases. M-ary PSK refers to multi-level modulation. Multi-level modulation techniques permit high data rates within fixed bandwidth constraints.

Procedure:-

1. Click on the MATLAB icon, then **command window** will be opened.
2. Go to file and open new **Script**.
3. Save the program using specified name other than program name with **.m** extension.
4. **Debug** the program for errors.
5. If errors are present correct them.
6. **Run** the program.
7. Observe for resultant **graphs**.

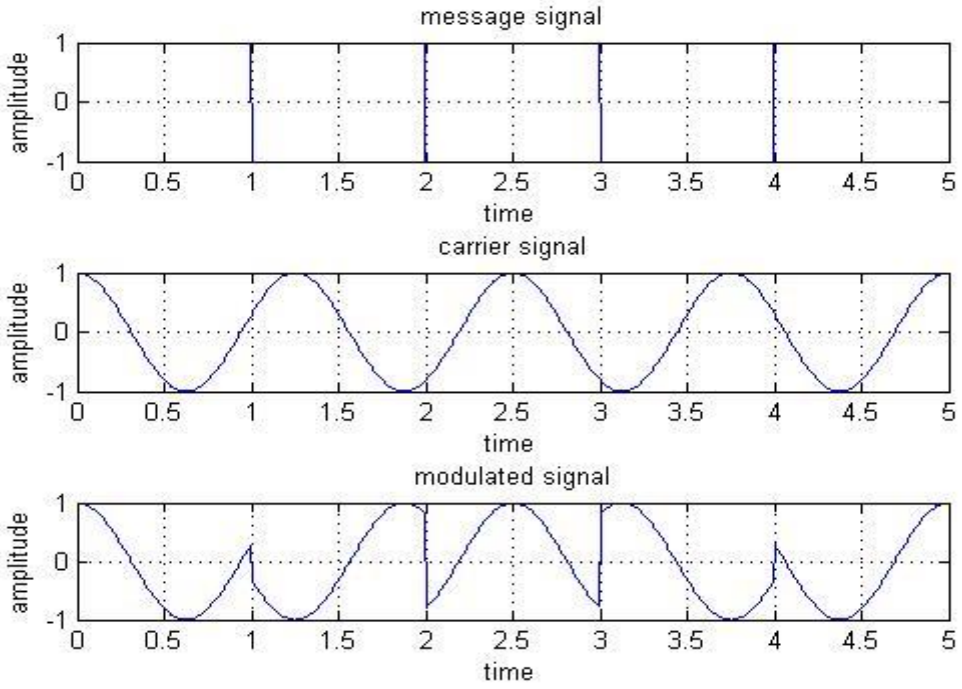
Program:-

```
clc;
clear all;
close all;
b=input('enter the binary sequence');
fc=input('enter carrier frequency');
n=length(b);
t=0:0.01:n;
x=1:1:(n+1)*100;
for i=1:n
   for j=1:0.01:i+1;
      bw(x(i*100:(i+1)*100))=b(i);
   end;
end;
bw=bw(100:end);
carrier=cos(2*pi*fc*t);
modulated=bw.*carrier;
subplot(3,1,1);
plot(t,bw);
grid on;
xlabel('time');
ylabel('amplitude');
title('message signal;');
subplot(3,1,2);
plot(t,carrier);
grid on;
xlabel('time');
ylabel('amplitude');
title('carrier signal');
```

```
subplot(3,1,3);
plot(t,modulated');
grid on;
xlabel('time');
ylabel('amplitude');
title('modulated signal');
```

Output:-

```
enter the binary sequence [1 -1 1 -1 1]
enter carrier frequency 0.8
```



Viva-Voce Questions:-

1. What are digital modulation techniques?
2. What is PSK?
3. Define ‘M’ in M-ary PSK.
4. What are the advantages of M-ary techniques?
5. Mathematically, represent M-ary PSK.

6. Simulation of Differential Pulse Code Modulation

Aim: To simulate Differential Pulse Code Modulation

Software:- MATLAB 7.11.0.584 (R2010b) version

Theory:- Differential pulse-code modulation (**DPCM**) is a signal encoder that uses the baseline of pulse-code modulation (PCM) but adds some functionalities based on the prediction of the samples of the signal. The input can be an analog signal or a digital signal.

If the input is a continuous-time analog signal, it needs to be sampled first so that a discrete-time signal is the input to the DPCM encoder

1. Take the values of two consecutive samples; if they are analog samples, quantize them; calculate the difference between the first one and the next; the output is the difference, and it can be further entropy coded.

2. Instead of taking a difference relative to a previous input sample, take the difference relative to the output of a local model of the decoder process; in this option, the difference can be quantized, which allows a good way to incorporate a controlled loss in the encoding.

Applying one of these two processes, short-term redundancy (positive correlation of nearby values) of the signal is eliminated; compression ratios on the order of 2 to 4 can be achieved if differences are subsequently entropy coded, because the entropy of the difference signal is much smaller than that of the original discrete signal treated as independent samples.

Procedure:-

1. Click on the MATLAB icon, then **command window** will be opened.
2. Go to file and open new **Script**.
3. Save the program using specified name other than program name with **.m** extension.
4. **Debug** the program for errors
5. If errors are present correct them.
6. **Run** the program.
7. Observe for resultant **graphs**.

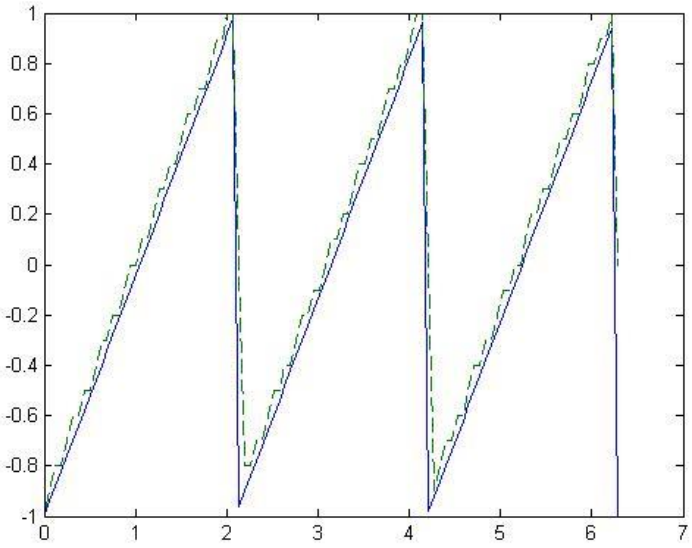
Program:-

A simple special case of DPCM quantizes the difference between the signal's current value and its value at the previous step. Thus the predictor is just $y(k) = x(k - 1)$. The code below implements this scheme. It encodes a saw-tooth signal, decodes it, and plots both the original and decoded signals. The solid line is the original signal, while the dashed line is the recovered signals. The example also computes the mean square error between the original and decoded signals.

```
predictor = [0 1]; % y(k)=x(k-1)
partition = [-1:.1:.9];
codebook = [-1:.1:1];
t = [0:pi/50:2*pi];
x = sawtooth(3*t); % Original signal
% Quantize x using DPCM.
encodedx = dpcmenco(x,codebook,partition,predictor);
% Try to recover x from the modulated signal.
decodedx = dpcmdeco(encodedx,codebook,predictor);
plot(t,x,t,decodedx,'--')
distor = sum((x-decodedx).^2)/length(x) % Mean square error
```

Output:-

distor = 0.032 distor =0.032 distor = 0.0327



Viva-Voce Questions:-

1. What are digital pulse modulation techniques?
2. Define DPCM.
3. Why DPCM is better than PCM?
4. What are the applications of DPCM?
5. Define step size.

7. Evaluation of DFT and IDFT of 16 sample sequence using DIT Algorithm

7(a) Evaluation of DFT of 16 sample sequence using DIT Algorithm

Aim: To simulate Discrete Fourier Transform of 16-sampled sequence using Decimation in Time.

Software:- MATLAB 7.11.0.584(R2010b) version

Theory:-

The discrete Fourier transform (DFT) is one of the specific forms of Fourier analysis. It transforms one function into another, which is called the *frequency domain* representation, or simply the *DFT*, of the original function (which is often a function in the time domain). But the DFT requires an input function that is *discrete* and whose non-zero values have a limited (*finite*) duration. Such inputs are often created by sampling a continuous function

An *N*-point DFT is expressed as an *N*-by-*N* matrix multiplication as $X = Wx$, where x is the original input signal, and X is the DFT of the signal.

$$X_k = \sum_{n=0}^{N-1} x_n e^{-\frac{2\pi i}{N} nk} \qquad k = 0, \ldots, N-1.$$

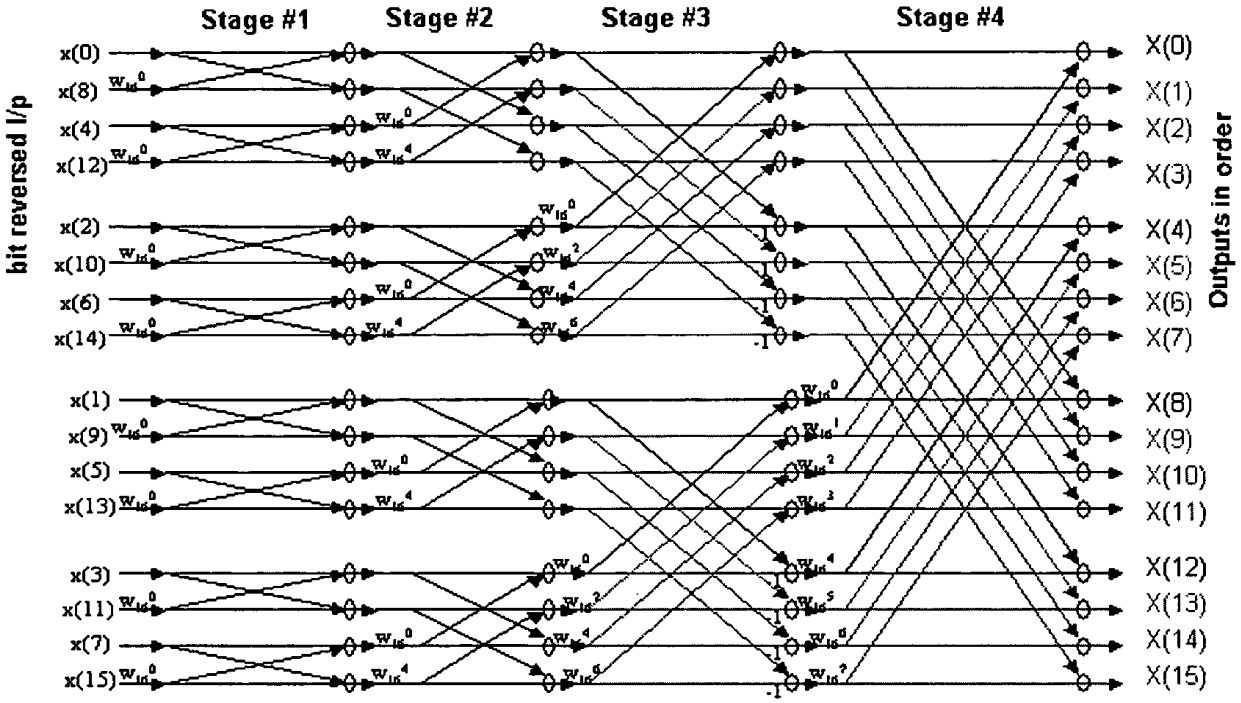
Procedure:-

1. Click on the MATLAB icon, then **command window** will be opened.
2. Go to file and open new **Script**.
3. Save the program using specified name other than program name with **.m** extension.
4. **Debug** the program for errors.
5. If errors are present correct them.
6. **Run** the program.
7. Observe for resultant **graphs**.

Calculations:

$$\mathbf{X(k)} = \sum_{n=0}^{N-1} x(n)\, \mathbf{e}^{-j2\pi kn/N}$$

$k=Nt/2^n$ N=16, n=4, t= 0,1,2,…….2^{m-1}



Butterfly distribution for 4-processor configuration. N=16, P=4.

Program:

```
clc;
clear all;
close all;
a1=input('enter the sequence:');
a=bitrevorder(a1);
N=length(a1);
n=log2(N);
b=a;
i=sqrt(-1);
for m=1:n
   for t=1:2^(m-1)
      k(t)=(N*(t-1))/2^m;
      w(t)=exp(-1*i*(2*pi*k(t))/N);
   end
   for p=1:2^m:(N-2^(m-1))
      x=p;
      r=1;
      for q=(x+2^(m-1)):((x+2^m)-1)
         b(p)=b(p)+b(q)*w(r);
         b(q)=b(p)-2*b(q)*w(r);
         p=p+1;
         r=r+1;
      end
      p=x;
   end
end
disp(b);
```

Output:-

enter the sequence: [1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0]

8 0 0 0 0 0 0 0 8 0 0 0 0 0 0 0

7(b) Evaluation of IDFT of 16 sample sequence using DIT Algorithm

Aim: To simulate Discrete Fourier Transform of 16-sampled sequence using Decimation in Time.

Software:- MATLAB 7.11.0.584(R2010b) version

Theory: Fast Fourier transform algorithm can be used to compute an inverse discrete Fourier transform without any change in the algorithm. The inverse discrete Fourier transform of an N-point sequence X(k) where k=0,1,---N-1 is defined as

$$X(n) = \frac{1}{N}\sum_{k=0}^{n-1} x(k)\, w_N^{-nk} \quad \text{where} \quad w_N = e^{-j2\pi/N}$$

The desired output sequence x(n) can be obtained by complex conjugating the DFT and dividing by N. The input sequence must be normal order and output sequence is in bit reversal order.

Procedure:-

1. Click on the MATLAB icon, then **command window** will be opened.
2. Go to file and open new **Script**.
3. Save the program using specified name other than program name with **.m** extension.
4. **Debug** the program for errors.
5. If errors are present correct them.
6. **Run** the program.
7. Observe for resultant **graphs**.

Calculations:

$$X(n) = \frac{1}{N}\sum_{k=0}^{n-1} x(k)\, w_N^{-nk} \quad \text{where} \quad w_N = e^{-j2\pi/N}$$

$k=Nt/\ 2^n$, N= 16, n=4 ,t=0,1,-----2^{m-1}-1

Program:-

```
clc;
clear all;
close all;
a1=input('enter the sequence:');
a=bitrevorder(a1);
N=length(a1);
n=log2(N);
b=a;
i=sqrt(-1);
for m=1:n
   for t=1:2^(m-1)
      k(t)=(N*(t-1))/2^m;
      w(t)=exp(-1*i*(2*pi*k(t))/N);
   end
   for p=1:2^m:(N-2^(m-1))
      x=p;
      r=1;
      for q=(x+2^(m-1)):((x+2^m)-1)
         b(p)=b(p)+b(q)*w(r);
         b(q)=b(p)-2*b(q)*w(r);
         p=p+1;
         r=r+1;
      end
      p=x;
   end
end
disp(b/N);
```

Output:

enter the sequence:[8 0 0 0 0 0 0 0 8 0 0 0 0 0 0 0]

1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0

Viva-Voce Questions:-

1. Define DFT of a discrete time sequence.
2. List any four properties of DFT.
3. What is FFT and what is its importance?
4. Compare FFT and DFT.
5. What are the various algorithms to calculate FFT?

8 . Evaluation of DFT and IDFT of 16 sample sequence using DIF Algorithm

8(a) Evaluation of DFT of 16 sample sequence using DIF Algorithm

Aim: To simulate Discrete Fourier Transform of 16-sampled sequence using Decimation in Frequency.

Software:- MATLAB 7.11.0.584(R2010b) version

Theory:-

The Fast fourier transform is a highly efficient procedure for computing the DFT of finite series and requires less number of computations than that of direct evaluation of DFT.This DFT is a powerful tool which is calculated as

$$X(k) = \sum_{n=0}^{N-1} x(n)\, w_N^K \quad \text{where } 0<k<N-1$$

The factor w_N^K is twiddle factor and is given by $e^{-j2\pi n/N}$

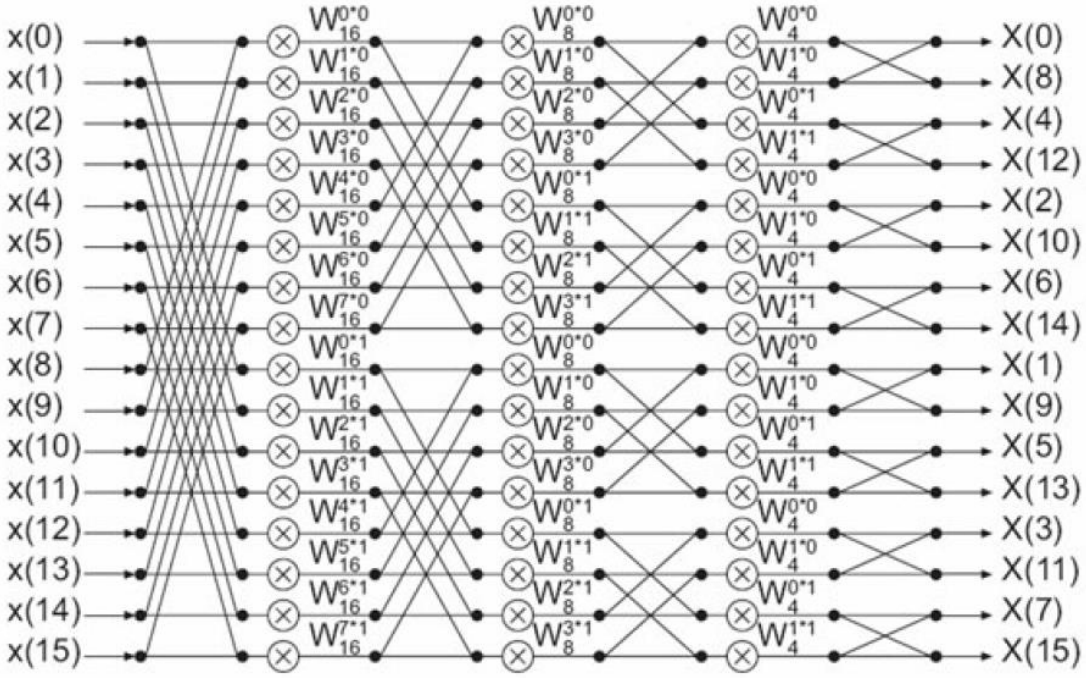
Procedure:-

1. Click on the MATLAB icon, then **command window** will be opened.
2. Go to file and open new **Script**.
3. Save the program using specified name other than program name with **.m** extension.
4. **Debug** the program for errors.
5. If errors are present correct them.
6. **Run** the program.
7. Observe for resultant **graphs**.

Calculations:

$$X(k) = \sum_{n=0}^{N-1} x(n)\, w_N^K \quad \text{where } 0<k<N-1$$

$$w_N = e^{-j2\pi n/N}$$



Program:-

```
clc;
clear all;
close all;
a1=input('enter the sequence:');
N=length(a1);
n=log2(N);
b=a1;
for m=n:-1:1
   for t=1:2^(m-1)
```

```
        k(t)=(N*(t-1))/2^m;
        w(t)=exp((-1*1i)*2*pi*k(t)/N);
    end
    for p=1:2^m:N-2^(m-1)
        x=p;
        r=1;
        for q=x+2^(m-1):x+(2^m)-1
            b(p)=b(p)+b(q);
            b(q)=(b(p)-2*b(q))*w(r);
            p=p+1;
            r=r+1;
        end
        p=x;
    end
end
disp(b);
```

Output:-

enter the sequence[1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0]

8 8 0 0 0 0 0 0 0 0 0 0 0 0 0 0

8(b) Evaluation of IDFT of 16 sample sequence using DIF Algorithm

Aim: To simulate Discrete Fourier Transform of 16-sampled sequence using Decimation in Frequency.

Software:- MATLAB 7.11.0.584(R2010b) version

Theory:-

Fast Fourier transform algorithm can be used to compute an inverse discrete Fourier transform without any change in the algorithm. The inverse discrete Fourier transform of an N-point sequence X(k) where k=0,1,---N-1 is defined as

$$X(n) = \frac{1}{N}\sum_{k=0}^{n-1} x(k)\, w_N^{-nk} \quad \text{where} \quad w_N = e^{-j2\pi/N}$$

The w_N^{-nk} is calculated for different values of k. It is called twiddle factor. The desired output x(n) can be obtained by complex conjugate of the discrete Fourier transform.

Procedure:-

1. Click on the MATLAB icon, then **command window** will be opened.
2. Go to file and open new **Script**.
3. Save the program using specified name other than program name with **.m** extension.
4. **Debug** the program for errors.
5. If errors are present correct them.
6. **Run** the program.
7. Observe for resultant **graphs**.

Calculations:

$$X(n) = \frac{1}{N}\sum_{k=0}^{n-1} x(k)\, w_N^{-nk} \quad \text{where} \quad w_N = e^{-j2\pi/N}$$

$$k = \frac{Nt}{2^n}, \quad N = 16,\ n = 4,\ t = 0,1,\text{-----}2^{m-1}-1$$

Program:-

```
clc;
clear all;
close all;
a1=input('enter the sequence:');
N=length(a1);
n=log2(N);
b=a1;
for m=n:-1:1
   for t=1:2^(m-1)
      k(t)=(N*(t-1))/2^m;
      w(t)=exp((-1*1i)*2*pi*k(t)/N);
   end
   for p=1:2^m:N-2^(m-1)
      x=p;
      r=1;
      for q=x+2^(m-1):x+(2^m)-1
         b(p)=b(p)+b(q);
         b(q)=(b(p)-2*b(q))*w(r);
         p=p+1;
         r=r+1;
      end
```

```
            p=x;
        end
    end
    disp(b/N);
```

Output:

enter the sequence:[8 0 0 0 0 0 0 0 8 0 0 0 0 0 0 0]

1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0

Viva-Voce Questions:-

1. What is twiddle factor?
2. What is the formula for calculating N-point DFT?
3. Define symmetry property.
4. What is the difference between DIT and DIF FFT?
5. Represent the basic butterfly computation in the DIF FFT algorithm.

9. Design of IIR Butterworth Filter using Impulse Invariant Method

Aim: To simulate IIR Butterworth filter using Impulse Invariant Method.

Software:- MATLAB 7.11.0.584(R2010b) version

Theory:-

Butterworth filters have very smooth passband, which we pay for with a relatively wide transition region. A Butterworth filter is characterized by its magnitude frequency response,

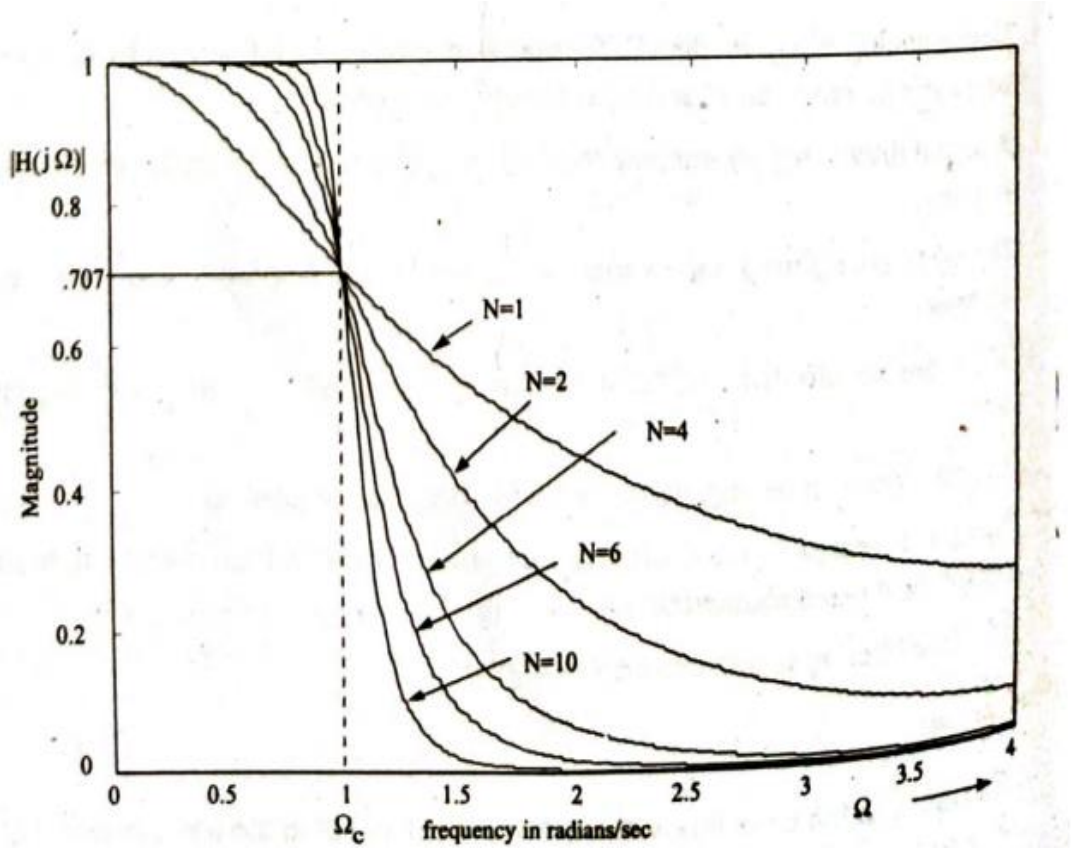
$$| H(j\Omega) | = 1 / [1+(\Omega/\Omega_c)^{2N}]^{1/2}$$

where N is the order of the filter and Ω_c is defined as the cutoff frequency where the filter magnitude is $1/\sqrt{2}$ times the dc gain (Ω=0).

$$N = \frac{\log \sqrt{\frac{10^{0.1\alpha_s} - 1}{10^{0.1\alpha_p} - 1}}}{\log \frac{\Omega_s}{\Omega_p}}$$

N	Denominator of $H(s)$
1	$s+1$
2	$s^2+\sqrt{2}s+1$
3	$(s+1)(s^2+s+1)$
4	$(s^2+0.76537s+1)(s^2+1.8477s+1)$
5	$(s+1)(s^2+0.61803s+1)(s^2+1.61803s+1)$
6	$(s^2+1.931855s+1)(s^2+\sqrt{2}s+1)(s^2+0.51764s+1)$
7	$(s+1)(s^2+1.80194s+1)(s^2+1.247s+1)(s^2+0.445s+1)$

Butterworth Polynomials for various values of N for Ω_c=1rad/sec



Lowpass Butterworth Magnitude Response

Procedure:-

1. Click on the MATLAB icon, then **command window** will be opened.
2. Go to file and open new **Script**.
3. Save the program using specified name other than program name with **.m** extension.
4. **Debug** the program for errors.
5. If errors are present correct them.
6. **Run** the program.
7. Observe for resultant **graphs**.

Program:-

```
clc;
clear all;
close all;
as=input('Enter stop band attenuation:');
ap=input('Enter pass band attenuation:');
us=input('Enter stop band attenuation:');
up=input('Enter pass band attenuation:');
T=input('Enter sampling time period:');
ws=2/(T*(atan(us/(2*T))));
wp=2/(T*(atan(up/(2*T))));
b=(10^(0.1*as)-1)/(10^(0.1*ap)-1);
c=log10(b^(0.5));
d=log(us/up);
n=c/d;
n1=ceil(n);
w=0:100:1000000;
j=sqrt(-1);
m=(10^(0.1*ap)-1)^(1/(2*n1));
wc=up/m;
s=j*w/wc;
h=[];
if(n1==1)
   h=1./(s+1);
   x1=abs(h);
end
if(n1==2)
   h=1./((s^2)+sqrt(2)*s+1);
   x1=abs(h);
```

```
end
if(n1==3)
   h=1./((s+1).*(s^2)+s+1);
   x1=abs(h);
end
if(n1==4)
   h=1./((s^2+0.7653*s+1).*(s^2+1.8978*s+1));
   x1=abs(h);
end;
xlabel('Frequency');
ylabel('Magnitude in dbs');
title('IIR butterworth filter');
semilogx(x1);
```

Output:-

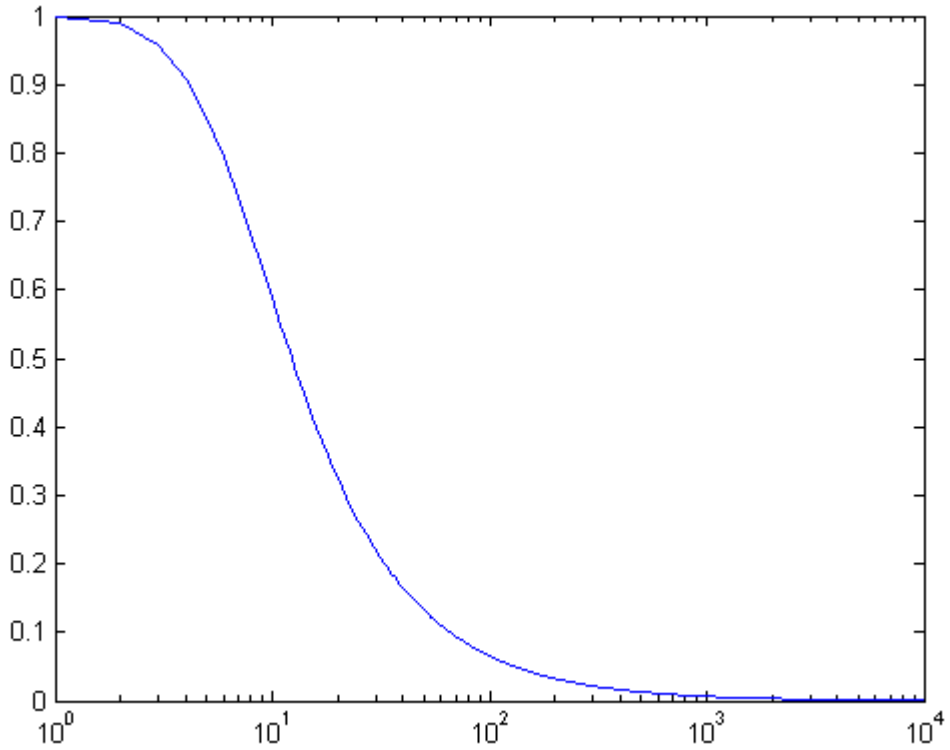
Enter stop band attenuation:10

Enter pass band attenuation:2

Enter stop band attenuation:1000

Enter pass band attenuation:500

Enter sampling time period:10



Viva-Voce Questions:-

1. Define an IIR filter
2. What is the magnitude frequency response of a Butterworth filter of order N?
3. Compare Butterworth and Chebyshev filters.
4. State the limitations of impulse invariant method.
5. Differentiate IIR and FIR filters.

10. Design of FIR Filter using Windowing Technique

Aim: To simulate FIR filter using Windowing Technique.

Software:- MATLAB 7.11.0.584(R2010b) version

Theory:-

A linear-phase is required throughout the passband of the filter to preserve the shape of the given signal in the passband. A causal IIR filter cannot give a linear-phase characteristics and only special types of FIR filters that exhibit center symmetry in its impulse response give the linear-space. An FIR filter with impulse response h(n) can be obtained as follows:

$$h(n) = h_d(n) \quad 0 \le n \le N-1$$
$$= 0 \quad \text{otherwise} \qquad \text{.....(a)}$$

The impulse response $h_d(n)$ is truncated at n = 0, since we are interested in causal FIR Filter. It is possible to write above equation alternatively as

$$h(n) = h_d(n)w(n) \qquad \text{.....(b)}$$

where w(n) is said to be a rectangular window defined by

$$w(n) = 1 \quad 0 \le n \le N-1$$
$$= 0 \quad \text{otherwise}$$

Taking DTFT on both the sides of equation (b), we get

$$H(\omega) = H_d(\omega) * W(\omega)$$

Procedure:-

1. Click on the MATLAB icon, then **command window** will be opened.
2. Go to file and open new **Script**.
3. Save the program using specified name other than program name with **.m** extension.
4. **Debug** the program for errors.

5. If errors are present correct them.
6. **Run** the program.
7. Observe for resultant **graphs**.

Program:-

Traingular window:

```
clc;
close all;
clear all;
format long;
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*(fp/f);
ws=2*(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;
y=triang(n1);
%Lowpass 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);
ylabel('gain in db---->');
xlabel('Normalised frequency---->');
title('FIR filter using Triangular window of LPF ----');
grid on;
%Highpass 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);
ylabel('gain in db---->');
xlabel('Normalised frequency---->');
title('FIR filter using Triangular window of HPF ----');
grid on ;
```

Kaiser window:

```
clc;
close all;
clear all;
format long;
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');
```

```
beta=input('enter the beta value');
wp=2*(fp/f);
ws=2*(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;
y=kaiser(n1,beta);
%Lowpass 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);
ylabel('gain in db---->');
xlabel('Normalised frequency---->');
title('FIR filter using Kaiser window of LPF ----');
grid on;
```

Rectangular window:

```
clc;
close all;
clear all;
format long;
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*(fp/f);
ws=2*(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;
y=boxcar(n1);
%Lowpass 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);
ylabel('gain in db---->');
xlabel('Normalised frequency---->');
title('FIR filter using Rectangular window of LPF ----');
grid on;
%Highpass 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);
ylabel('gain in db---->');
xlabel('Normalised frequency---->');
title('FIR filter using Rectangular window of HPF ----');
title('Magnitude response of high pass filter');
grid on;
```

Output:-

Triangular window:

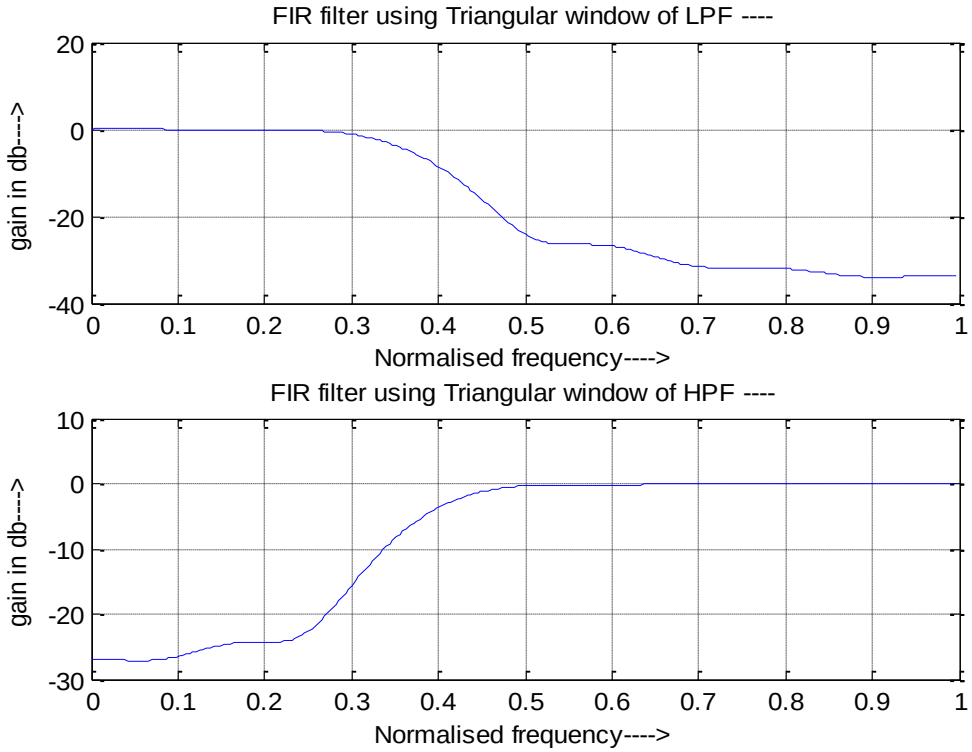
enter the passband ripple0.04

enter the stopband ripple0.02

enter the passband frequency1500

enter the stopband frequency2000

enter the sampling frequency8000



Kaiser window:

enter the passband ripple:0.02

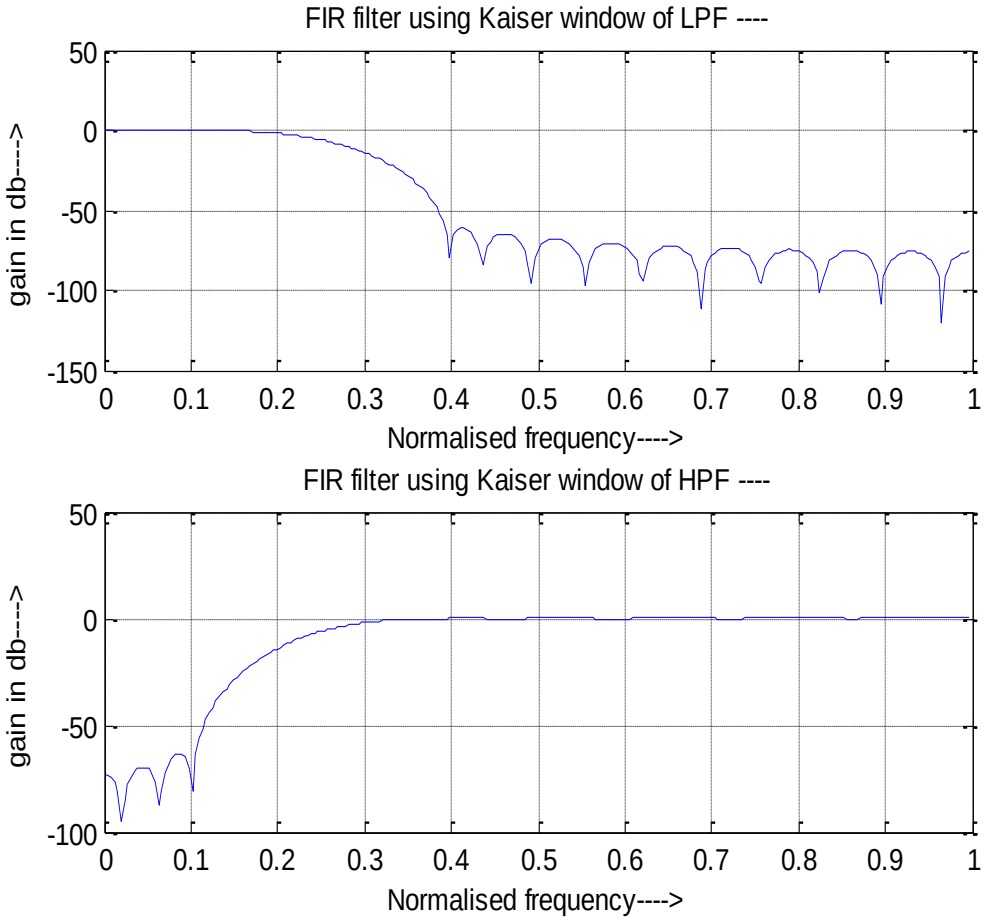
enter the stopband ripple:0.01

enter the passband frequency:1000

enter the stopband frequency:1500

enter the sampling frequency:8000

enter the beta value:5.8



Rectangular window:

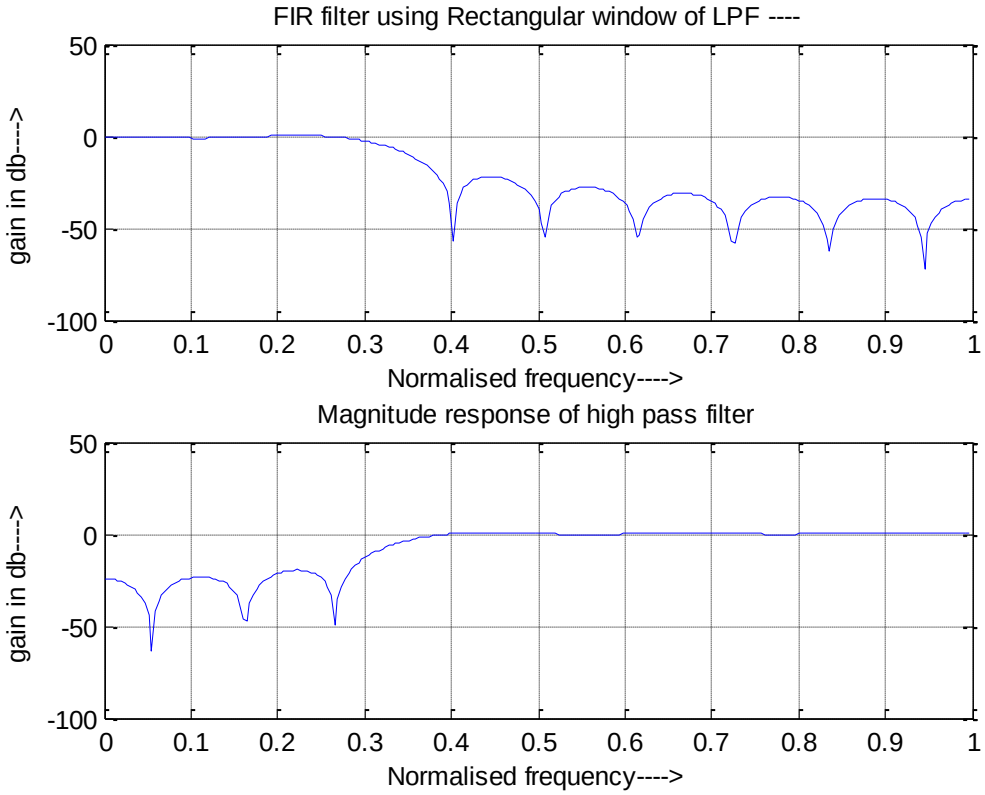
enter the passband ripple:0.04

enter the stopband ripple:0.05

enter the passband frequency:1500

enter the stopband frequency:2000

enter the sampling frequency:9000



Viva-Voce Questions:-

1. What are FIR filters?
2. What are the steps involved in the design FIR filters using Windows?
3. What is Gibb's Phenomenon?
4. What are the advantages and disadvantages of FIR filters?
5. What is the condition for linear phase FIR filter?

11. CONVOLUTION OF TWO SIGNALS

11(a).Linear Convolution

Aim: To simulate linear convolution.

Software:- MATLAB 7.11.0.584(R2010b) version

Theory:-

Convolution is a formal mathematical operation, just as multiplication, addition, and integration. Addition takes two numbers and produces a third number, while convolution takes two signals and produces a third signal. Convolution is used in the mathematics of many fields, such as probability and statistics. In linear systems, convolution is used to describe the relationship between three signals of interest: the input signal, the impulse response, and the output signal.

$$y(n) = \sum_{k=0}^{N-1} x_1(k)\, x_2(n-k) \qquad 0<n<N-1$$

In this equation, $x_1(k)$, $x_2(n-k)$ and y(n) represent the input to and output from the system at time n. Here we could see that one of the input is shifted in time by a value every time it is multiplied with the other input signal. Linear Convolution is quite often used as a method of implementing filters of various types.

Procedure:-

1. Click on the MATLAB icon, then **command window** will be opened.
2. Go to file and open new **Script**.
3. Save the program using specified name other than program name with **.m** extension.
4. **Debug** the program for errors.
5. If errors are present correct them.
6. **Run** the program.
7. Observe for resultant **graphs**.

Calculations:

$$y(n) = x(n) * h(n) = \sum_{k=-\alpha}^{\alpha} x(k)h(n-k)$$

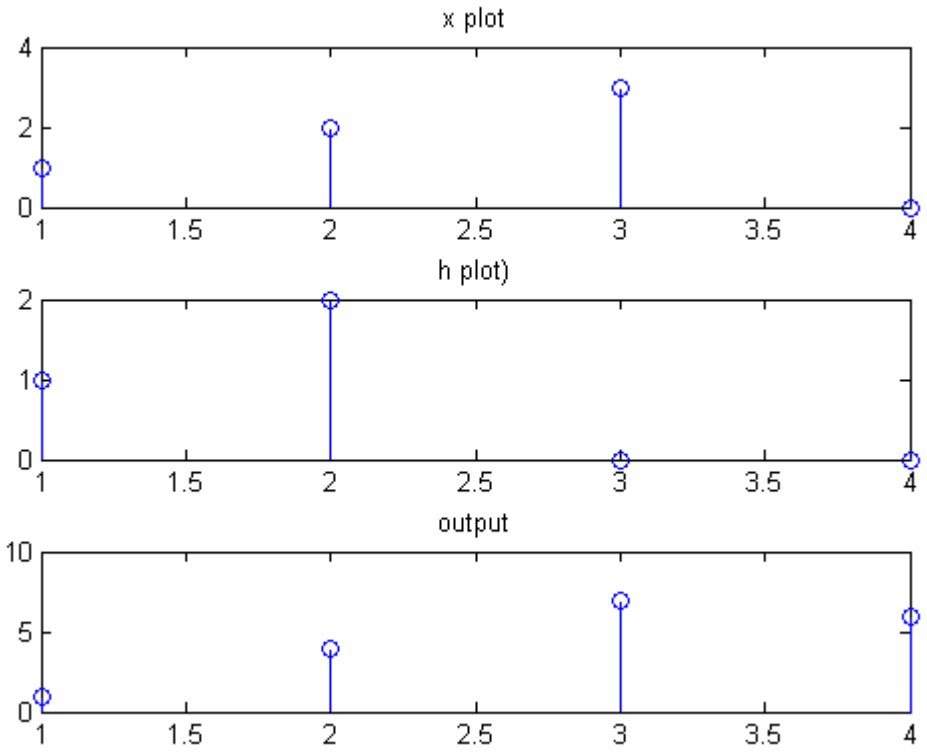
Program:-

```
clc;
clear all;
close all;
x=input('Enter first sequence for x:');
h=input('Enter second sequence for h:');
n=length(x);
m=length(h);
k=n+m-1;
x=[x,zeros(1,m-1)];
  h=[h,zeros(1,n-1)];
  y=zeros(1,k);
  for i=1:k
 for n=1:i
            y(i) = y(i)+x(n)*h(i-n+1);
end;
end;
disp(y);
subplot(3,1,1);
stem(x);
title('x plot');
subplot(3,1,2);
```

```
stem(h);
title('h plot)');
subplot(3,1,3);
stem(y);
title('output');
```

Output:-

```
Enter first sequence for x:[1 2 3]
Enter second sequence for h:[1 2]
   1    4    7    6
```



11(b).Circular Convolution

Aim: To simulate circular convolution.

Software:- MATLAB 7.11.0.584(R2010b) version

Theory:-

Circular convolution is another way of finding the convolution sum of two input signals. It resembles the linear convolution, except the sample values of one of the input signals is folded and right shifted before the convolution sum is found.

$$y(n) = x(n) \Theta h(n) = \sum_{i=0}^{N-1} x(i)h(n-i)$$

Procedure:-

1. Click on the MATLAB icon, then **command window** will be opened.
2. Go to file and open new **Script**.
3. Save the program using specified name other than program name with **.m** extension.
4. **Debug** the program for errors.
5. If errors are present correct them.
6. **Run** the program.
7. Observe for resultant **graphs**.

Calculations:

$$y(n) = x(n) \Theta h(n) = \sum_{i=0}^{N-1} x(i)h(n-i)$$

Program:-

```
clc;
clear all;
close all;
a=input('Enter first sequence:');
```

```
b=input('Enter second sequence:');
x=length(a);
y=length(b);
z=max(x,y);
if x>y
        for i=y+1:x
            b(i)=0;
        end
    else if x<y
            for j=x+1:y
                a(j)=0;
            end
        end
    end
    for i=1:z
        c(i)=0;
        for k=1:z
            j=(i-k)+1;
            if(j<=0)
                j=z+j;
            end
            c(i)=c(i)+(a(k)*b(j));
        end
    end
    disp(c);
```

Output:-

Enter first sequence:[1 2 3]

Enter second sequence:[1 2 0]

7 4 7

Viva-Voce Questions:-

1. Explain the significance of convolution.
2. Define linear convolution.
3. Define circular convolution.
4. What is zero padding?
5. What are the properties of linear convolution?
6. List few applications of linear convolution in LTI system design.

12. CORRELATION OF TWO SIGNALS

12(a).Auto Correlation

Aim: To simulate Auto correlation.

Software:- MATLAB 7.11.0.584(R2010b) version

Theory:-Correlation is the mathematical operation tha is related to the convolution.It is basically used to compare the two signals. Correlation is a measure of degree to which two signals are similar.The auto correlation of a sequence is correlation of a sequence with itself. The auto correlation of sequence x(n) is defined by

$$y(l) = \sum_{n=-\alpha}^{\alpha} x(n)x(n-l)$$

Procedure:-

1. Click on the MATLAB icon, then **command window** will be opened.
2. Go to file and open new **Script**.
3. Save the program using specified name other than program name with **.m** extension.
4. **Debug** the program for errors.
5. If errors are present correct them.
6. **Run** the program.
7. Observe for resultant **graphs**.

Calculations:

$$y(l) = \sum_{n=-\alpha}^{\alpha} x(n)x(n-l) \text{ where l=2n-1}$$

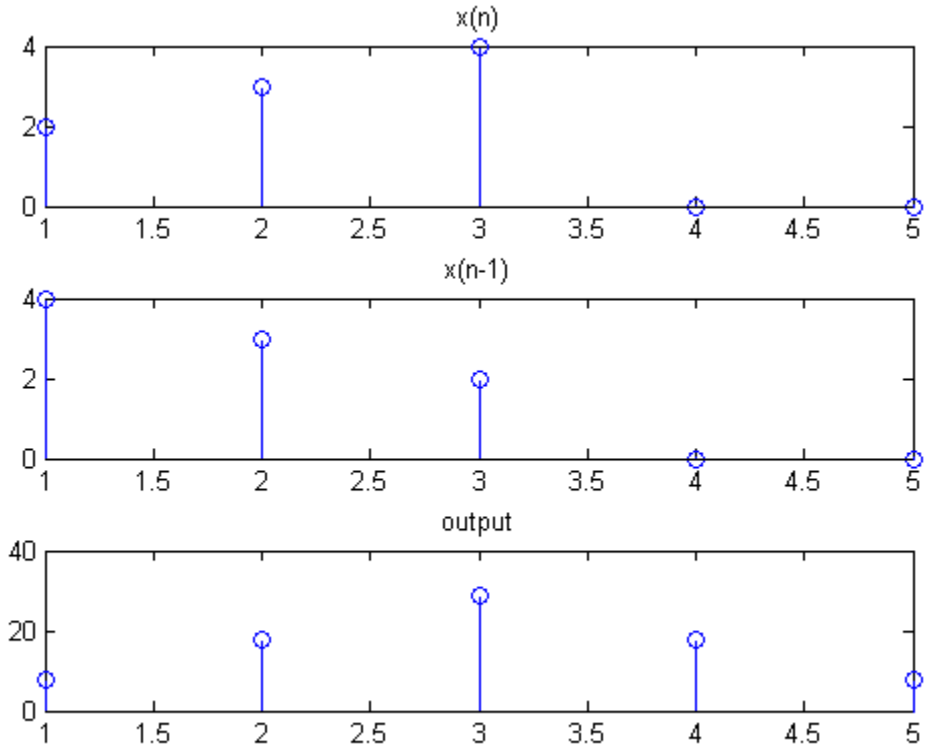
Program:-

```
clear all;
close all;
x= input('enter input sequence:');
n=length(x);
k=2*n-1;
b(1:n)=x(n:-1:1);
x=[x,zeros(1,n-1)];
b=[b,zeros(1,n-1)];
y=zeros(1,k);
    for i=1:k
        for j=1:i
y(i) = y(i)+x(j)*b(i-j+1);
    end;
        end;
    disp(y);
    subplot(3,1,1);
    stem(x);
    title('x(n)');
    subplot(3,1,2);
    stem(b);
    title('x(n-1)');
    subplot(3,1,3);
    stem(y);
    title('output');
```

Output:-

```
Enter input sequence: [2 3 4]

     8    18    29    18    8
```


x(n)
4
2
0
1
1.5
2
2.5
3
3.5
4
4.5
5
x(n-1)
4
2
0
1
1.5
2
2.5
3
3.5
4
4.5
5
output
40
20
0
1
1.5
2
2.5
3
3.5
4
4.5
5

12(b).Cross Correlation

Aim: To simulate Auto correlation.

Software:- MATLAB 7.11.0.584(R2010b) version

Theory:-

Correlation is the mathematical operation tha is related to the convolution.It is basically used to compare the two signals. Correlation is a measure of degree to which two signals are similar. cross correlation between a pair of signals x(n) and h(n) gives the measure of the degree of similarity between them. It is defined as

$$y(l) = \sum_{n=-\alpha}^{\alpha} x(n)h(n-l)$$

Procedure:-

1. Click on the MATLAB icon, then **command window** will be opened.
2. Go to file and open new **Script**.
3. Save the program using specified name other than program name with **.m** extension.
4. **Debug** the program for errors.
5. If errors are present correct them.
6. **Run** the program.
7. Observe for resultant **graphs**.

Calculations:

$$y(l) = \sum_{n=-\alpha}^{\alpha} x(n)h(n-l)$$

Program:-

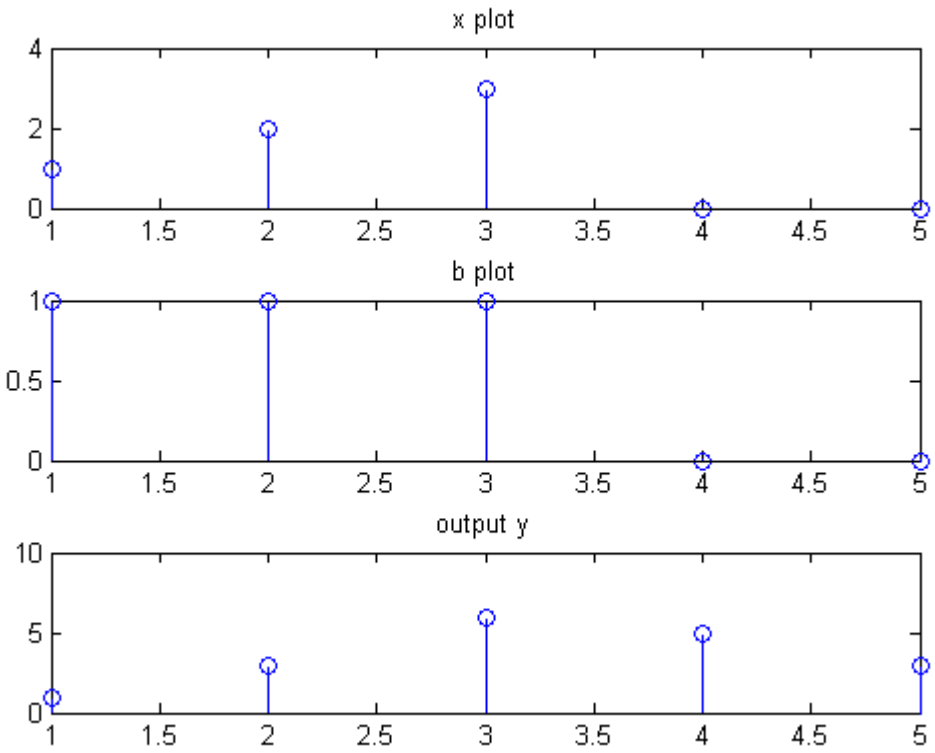
```
clear all;
close all;
x= input('enter input sequence for x:');
b=input('enter input sequence for b:');
n=length(x);
m=length(b);
k=n+m-1;
b(1:m)=b(m:-1:1);
x=[x,zeros(1,m-1)];
b=[b,zeros(1,m-1)];
y=zeros(1,k);
    for i=1:k
      for j=1:i
          y(i) = y(i)+x(j)*b(i-j+1);
      end;
    end;
disp(y);
subplot(3,1,1);
stem(x);
title('x plot');
subplot(3,1,2);
stem(b);
title('b plot');
subplot(3,1,3);
stem(y);
title('output y');
```

Output:-

Enter input sequence for x: [1 2 3]

Enter input sequence for b: [1 1 1]

1 3 6 5 3



Viva-Voce Questions:-

1. Differentiate correlation and convolution.
2. Difference between autocorrelation and cross correlation.
3. What are the properties of auto correlation?
4. Write the mathematical formula to find the cross correlation.
5. List few applications of correlation.

13. DFT Analysis of a Noise Corrupted Signal

Aim: To simulate MATLAB program of DFT analysis of a noise corrupted signal

Software:- MATLAB 7.11.0.584(R2010b) version

Theory:-

Noise is an unwanted signal which interferes with the original message signal and corrupts the parameters of the message signal. The corrupted signal is generated by adding noise to a sine wave. DFT calculates the signal's frequency spectrum. The power spectral density of the signal describes the power present in the signal as a function of frequency. It is used to characterize broadband random signals.

Procedure:-

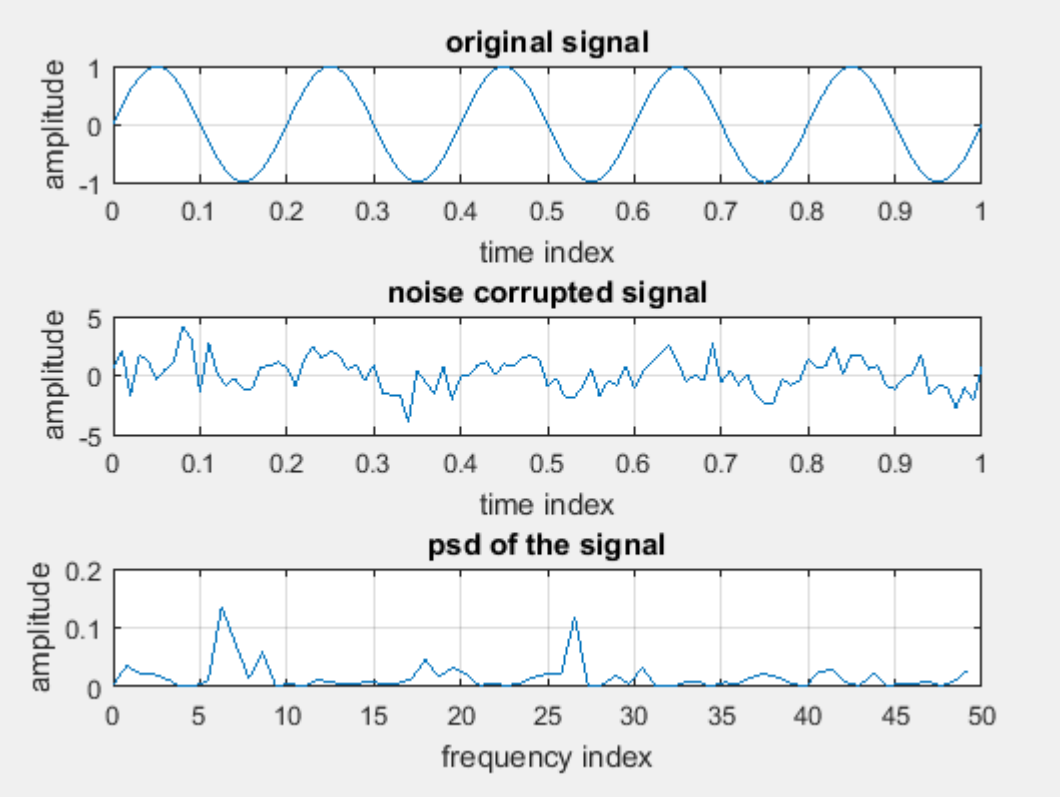
1. Click on the MATLAB icon, then **command window** will be opened.
2. Go to file and open new **Script**.
3. Save the program using specified name other than program name with **.m** extension.
4. **Debug** the program for errors.
5. If errors are present correct them.
6. **Run** the program.
7. Observe for resultant **graphs**.

Program:-

```
clc; %clear the command window
clear all; %clear all the variables
close all; %close all values
t=0:0.01:1; %time index array elements
 x=sin(2*pi*5*t); %generation of sine wave
 subplot(3,1,1); %plotting the figure
```

```
 grid on; %grid lines
 plot(t,x); %plot the magnitude response
xlabel('time index'); %x axis index
ylabel('amplitude'); %y axis index
title('original signal'); %title of the plot
 grid on; %grid lines
 y1=randn(1,101); %generation of noise signal
 y=x+y1; %generation of noise corrupted signal
subplot(3,1,2); %plotting the figure
 plot(t,y); %plot the magnitude response
 xlabel('time index'); %x axis index
 ylabel('amplitude'); %y axis index
 title('noise corrupted signal'); %title of the plot
grid on; %grid lines
 Y=fft(y,128); %finding the fast Fourier transform
 Pyy=y.*conj(y)/128; %power spectral density
 f=(100/128) *(0:63); %frequency index array
 subplot(3,1,3); %plotting the figure
plot(f,Pyy(1:64)); %plot the power spectral density response
 xlabel('frequency index'); %x axis index
 ylabel('amplitude'); %y axis index
title('psd of the signal'); %title of the plot
 grid on; %grid lines
```

Output:-



Viva-Voce Questions:-

1. What is noise corrupted signal?
2. What are the techniques to remove the noise?
3. Define power spectral density of a signal.
4. What is the need for transform?
5. What are the applications of DFT

Experiment beyond syllabus

DESIGN OF CHEBYSHEV-I IIR FILTER

AIM: To design a Chebyshev-I IIR filters (LPF/HPF/BPF/BSF) and demonstrates the filteringoperation using MATLAB program.

SOFTWARE REQUIRED:

MATLAB R2014a

ALGORITHM:

- Get the IIR filter specifications.
- Obtain the filter coefficients.
- Plot the frequency response of the filters.

PROGRAM:

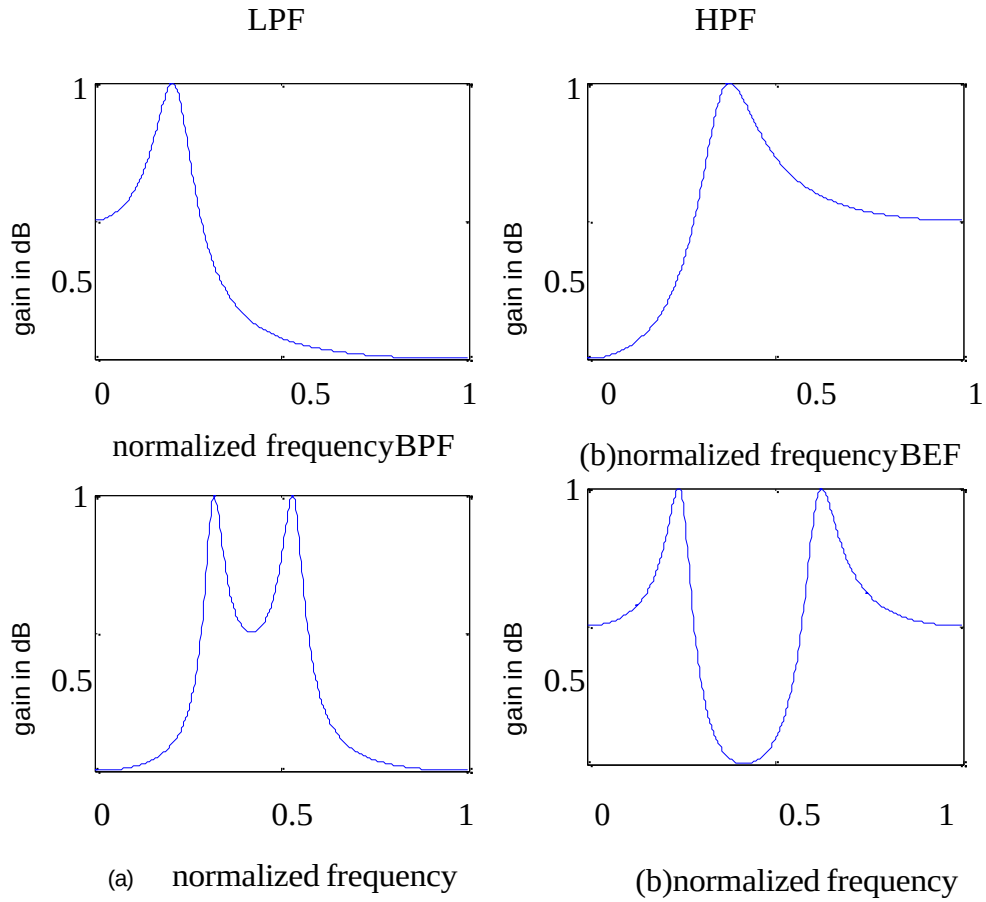
```
clc;
clearall;
closeall;
rp=input('Enter the passband ripple');
rs=input('Enter the stopband ripple');
wp=input('Enter the passband frequency');
ws=input('Enter the stopband frequency');
f=input('Enter the sampling frequency');
w1=2*wp/f;
w2=2*ws/f;
[n,wn]=cheb1ord(w1,w2,rp,rs);
[b,a]=cheby1(n,rp,wn,'low');
[h,w]=freqz(b,a,512);
subplot(2,2,1);
plot(w/pi,abs(h));
ylabel('gain in dB');
```

```
xlabel('(a)normalized frequency');
title('LPF');
[b,a]=cheby1(n,rp,wn,'high');
[h,w]=freqz(b,a,512);
subplot(2,2,2);
plot(w/pi,abs(h));
ylabel('gain in dB');
xlabel('(b)normalized frequency');
title('HPF');
wn1=[w1 w2];
[b,a]=cheby1(n,rp,wn1);
[h,w]=freqz(b,a,512);
subplot(2,2,3);
plot(w/pi,abs(h));
ylabel('gain in dB');
xlabel('(c)normalized frequency');
title('BPF');
wn2=[w1 w2];
[b,a]=cheby1(n,rp,wn2,'stop');
[h,w]=freqz(b,a,512);
subplot(2,2,4);
plot(w/pi,abs(h));
ylabel('gain in dB');
xlabel('(d)normalized frequency');
title('BSF');
```

INPUT

```
Enter the passband ripple 6
Enter the stopband ripple 20
Enter the passband frequency 1000
Enter the stopband frequency 2000
Enter the sampling frequency 7000
```

OUTPUT



RESULT: Thus the Chebyshev-I IIR filter was designed using MATLAB.

DESIGN OF CHEBYSHEV-II IIR FILTER

AIM:

To design a Chebyshev-II IIR filters (LPF/HPF/BPF/BSF) and demonstrates the filtering operation using MATLAB program.

SOFTWARE REQUIRED:

MATLAB R2014a

ALGORITHM:

- Get the IIR filter specifications.
- Obtain the filter coefficients using .
- Plot the frequency response of the filters.

PROGRAM:

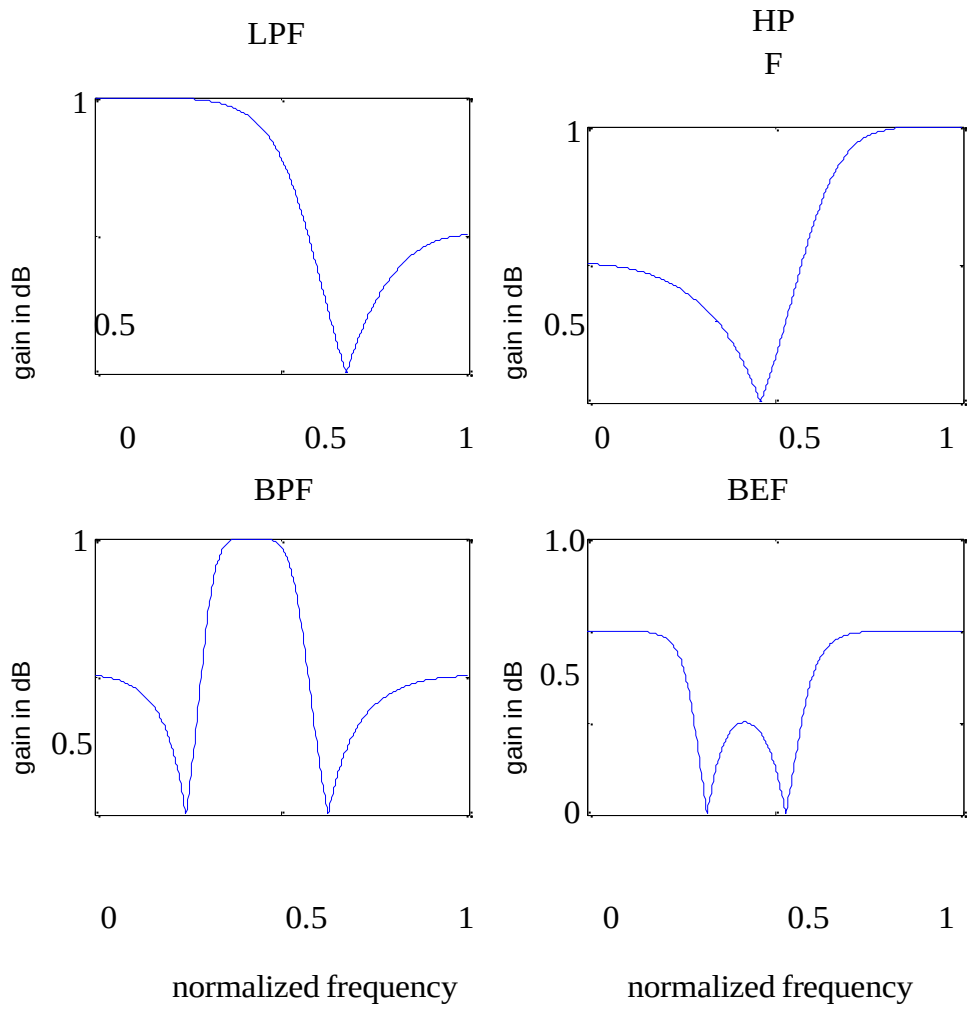
```
clc;
clearall;
closeall;
rp=input('Enter the passband ripple');
rs=input('Enter the stopband ripple');
wp=input('Enter the passband frequency');
ws=input('Enter the stopband frequency');
f=input('Enter the sampling frequency');
w1=2*wp/f;
w2=2*ws/f;
[n,wn]=cheb2ord(w1,w2,rp,rs);
[b,a]=cheby2(n,rp,wn,'low');
[h,w]=freqz(b,a,512);
subplot(2,2,1);
plot(w/pi,abs(h));
ylabel('gain in dB');
xlabel('(a)normalized frequency');
title('LPF');
[b,a]=cheby2(n,rp,wn,'high');
```

```
[h,w]=freqz(b,a,512);
subplot(2,2,2);
plot(w/pi,abs(h));
ylabel('gain in dB');
xlabel('(b)normalized frequency');
title('HPF');
wn1=[w1 w2];
[b,a]=cheby2(n,rp,wn1);
[h,w]=freqz(b,a,512);
subplot(2,2,3);
plot(w/pi,abs(h));
ylabel('gain in dB');
xlabel('(c)normalized frequency');
title('BPF');
wn2=[w1 w2];
[b,a]=cheby2(n,rp,wn2,'stop');
[h,w]=freqz(b,a,512);
subplot(2,2,4);
plot(w/pi,abs(h));
ylabel('gain in dB');
xlabel('(d)normalized frequency');
title('BSF');
```

INPUT

```
Enter the passband ripple 6
Enter the stopband ripple 20
Enter the passband frequency 1000
Enter the stopband frequency 2000
Enter the sampling frequency 7000
```

OUTPUT



RESULT: Thus the Chebyshev-II IIR filter was designed using MATLAB

Viva-Voce Questions:

1.What is the difference between Butterworth ,Chebyshev I &ChebyshevII filters?

2.The chebyshev-I filter is equi-ripple in pass band and monotonic in the stop band.

3.What is the value of chebyshev polynomial of degree 0?

4.The two popular techniques used to approximate the ideal frequency response are?