Experiment No. 7

Aim: Write a MATLAB Program to illustrate Zero Padding

Problem Statement: Obtain Linear Convolution using the method of Circular Convolution by using Zero Padding

Tool used: MATLAB

Theory: Convolution, one of the most important concepts in electrical engineering, can be used to determine the output a system produced for a given input signal. It can be shown that a linear time invariant system is completely characterized by its impulse response. The sifting property of the discrete time impulse function tells us that the input signal to a system can be represented as a sum of scaled and shifted unit impulses. Thus, by linearity, it would seem reasonable to compute the output signal as the sum of scaled and shifted unit impulse responses. That is exactly what the operation of convolution accomplishes. Hence, convolution can be used to determine a linear time invariant system's output from knowledge of the input and the impulse response.

Discrete time Linear convolution is an operation on two discrete time signals defined by the integral

$$(f*g)[n]=\sum_{k=-\infty}^{\infty} f(k)g[n-k]$$

Circular convolution, also known as **cyclic convolution**, is a special case of **periodic convolution**, which is the convolution of two periodic functions that have the same period. Periodic convolution arises, for example, in the context of the discrete-time Fourier transform (DTFT). In particular, the DTFT of the product of two discrete sequences is the periodic convolution of the DTFTs of the individual sequences. And each DTFT is a periodic summation of a continuous Fourier transform function. Although DTFTs are usually continuous functions of frequency, the concepts of periodic and circular convolution are also directly applicable to discrete sequences of data. In that context, circular convolution plays an important role in maximizing the efficiency of a certain kind of common filtering operation.

Linear and circular convolution are fundamentally different operations. However, there are conditions under which linear and circular convolution are equivalent. Establishing this equivalence has important implications. For two vectors, x and y, the circular convolution is equal to the inverse discrete Fourier transform (DFT) of the product of the vectors' DFTs. Knowing the conditions under which linear and circular convolution are equivalent allows you to use the DFT to efficiently compute linear convolutions.

The linear convolution of an N-point vector, x, and an L-point vector, y, has length $N + L - 1$.

For the circular convolution of x and y to be equivalent, one must pad the vectors with zeros to length at least $N + L - 1$ before you take the DFT. After you invert the product of the DFTs, retain only the first $N + L - 1$ elements.

Code:

MATLAB Program:

```
% To illustarte Zero Padding
% Obtain linear Convolution using Circular Convolution by Zero Padding
% Technique
x1=input('Enter the First Sequence x1(n)=');
x2=input('Enter the Second Sequence x2(n)=');
l1=length(x1);
l2=length(x2);
N=l1+l2-1;
%Zero Padding
xz1=[x1,zeros(1,N-l1)];
xz2=[x2,zeros(1,N-l2)];
% DFT of xz1 and xz2 is Xz1k and Xz2k respectively
n=[0:1:N-1];
k=[0:1:N-1];
nk=n'*k;
WN=exp(-1j*2*pi/N);
WNnk=WN.^nk;
Xz1k=xz1*WNnk;
Xz2k=xz2*WNnk;
% Multiplication of Two DFT Xz1k and Xz2k
Xz12k=Xz1k.*Xz2k;
%IDFT of Xz12k
WNI=exp(1j*2*pi/N);
WNInk=WNI.^nk;
y=(Xz12k*WNInk)/N;
disp('Linear Convolution using Circular Convolution by Zero Padding
Technique=')
disp(abs(y));
subplot(3,2,1);
stem(x1);
title('First Input Sequence x1(n)');
subplot(3,2,2);
stem(x2);
title('Second Input Sequence x2(n)');
subplot(3,2,3);
stem(xz1);
title('First Input Sequence with Zero Padding xz1(n)');
subplot(3,2,4);
stem(xz2);
title('Second Input Sequence with Zero Padding xz2(n)');
subplot(3,2,5);
stem(abs(y));
title('Linear Convolution using Circular Convolution by Zero Padding
Technique');
```

Input Variables:

1. Input Discrete Sequence $x_1[n]$
2. Input Discrete Sequence $x_2[n]$

Results and Observations:

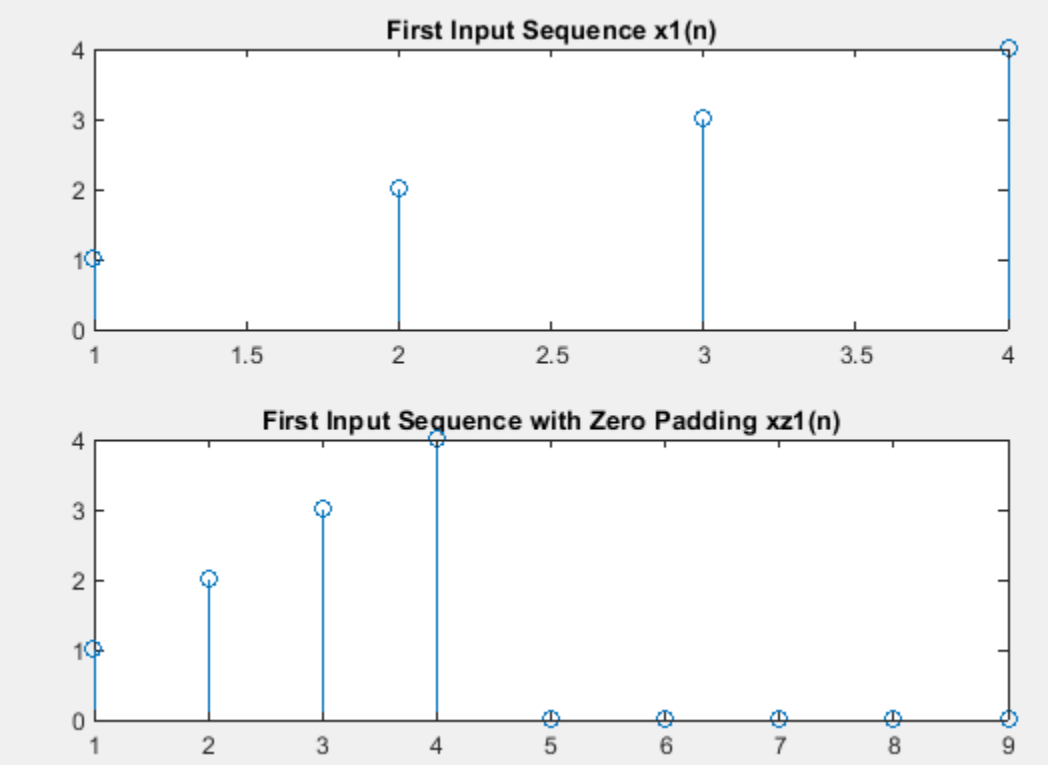


Fig 1. Plot of Discrete Sequence $x_1[n]$ and Discrete Sequence $x_1[n]$ after zero padding

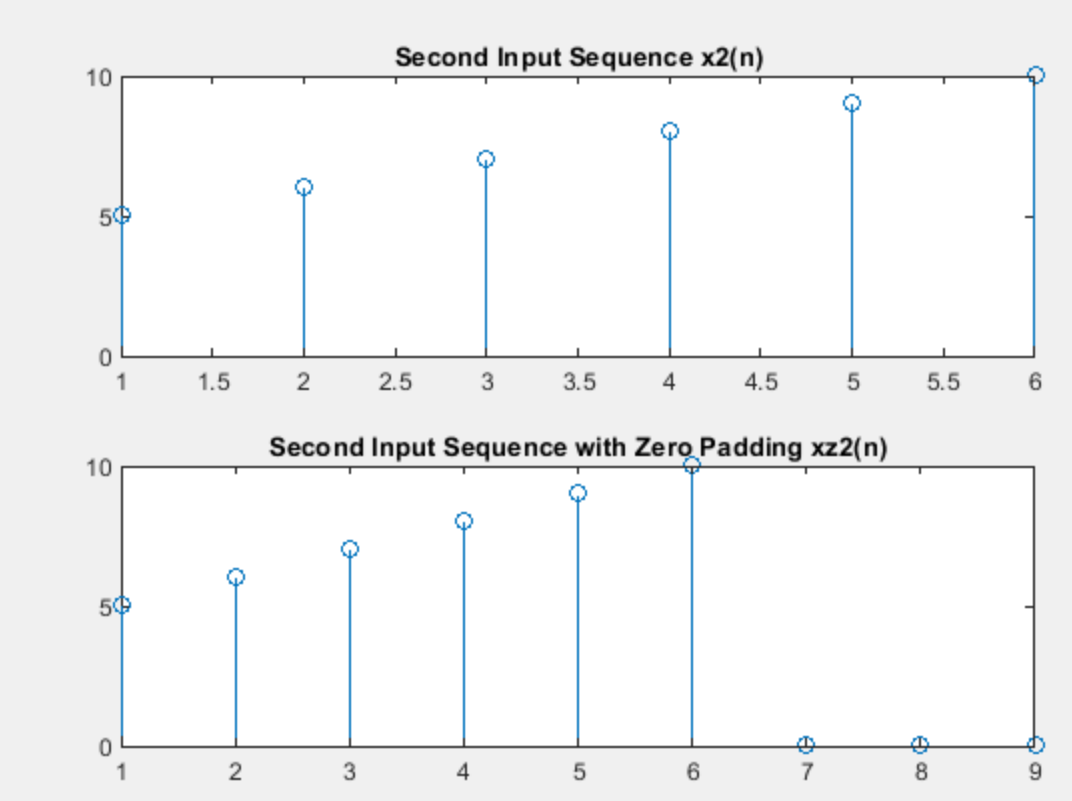


Fig 2. Plot of Discrete Sequence $x_2[n]$ and Discrete Sequence $x_2[n]$ after zero padding

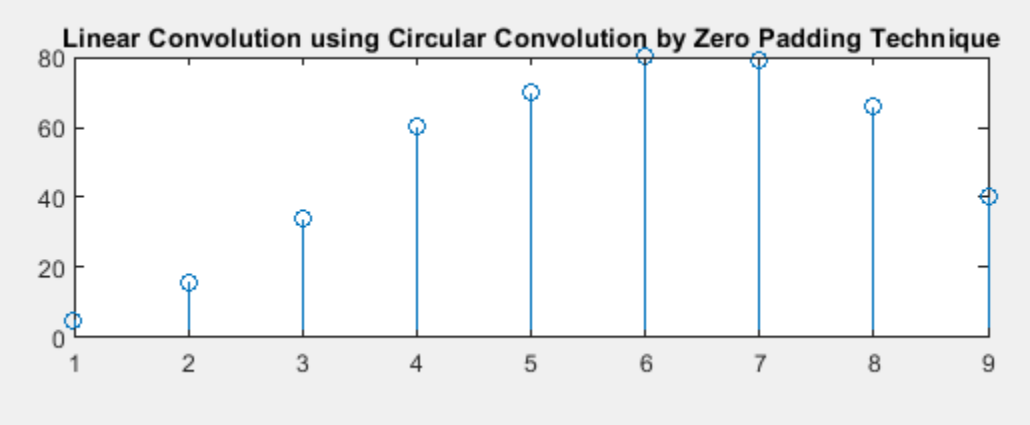


Fig3 Linear Convolution of Sequence $x_1[n]$ and $x_2[n]$ obtained from Circular Convolution by Zero Padding Technique

Conclusion: Thus it can be concluded from Fig 1, Fig 2 and Fig 3 that Linear Convolution can be obtained from Circular Convolution by Zero Padding Technique

Experiment No. 8

Aim: Write a MATLAB Program to design Butterworth Low Pass Filter and High Pass Filter

Tool used: MATLAB

Theory: The Butterworth filter is an analog filter design which produces the best output response with no ripple in the pass band or the stop band resulting in a maximally flat filter response but at the expense of a relatively wide transition band.

In applications that use filters to shape the frequency spectrum of a signal such as in communications or control systems, the shape or width of the roll-off is called the "transition band". For simple first-order filters this transition band maybe too long or too wide, so active filters designed with more than one "order" are required. These types of filters are commonly known as "High-order" or "n^{th}-order" filters.

An ideal electrical filter should not only completely reject the unwanted frequencies but should also have uniform sensitivity for the wanted frequencies".

Such an ideal filter cannot be achieved, but Butterworth showed that successively closer approximations were obtained with increasing numbers of filter elements of the right values. At the time, filters generated substantial ripple in the passband, and the choice of component values was highly interactive. Butterworth showed that a low-pass filter could be designed whose cutoff frequency was normalized to 1 radian per second. The frequency response of Low Pass Butterworth Filter is given by

$$G(w) = \frac{1}{\sqrt{1+W^{2n}}}$$

where w is the angular frequency in radians per second and n is the number of poles in the filter which is equal to the number of reactive elements in a passive filter.

A high pass filter is a circuit that attenuates all the signals below a specified cut off frequency denoted as f_L. Thus, a high pass filter performs the opposite function to that of low pass filter. Hence, the First Order High Pass Butterworth Filter circuit can be obtained by interchanging frequency determining resistances and capacitors in low pass filter circuit.

The frequency at which the gain is 0.707 times the gain of filter in pass band is called as low cut off frequency and denoted as f_L. So, all the frequencies greater than f_L are allowed to pass but the maximum frequency which is allowed to pass is determined by the closed loop bandwidth of the circuit used.

Code:

MATLAB Program:

a) Butterworth Low Pass Filter

```
% Design of Butterworth Low Pass Filter
% Passband Ripple=rp=0.15
% Stopband Ripple=rs=60
% Passband Frequency=wp=1500
% Stopband Frequency=ws=3000
```

```
% Sampling Frequency=fs=7000
rp=input('Enter the Passband Ripple =');
rs=input('Enter the Stopband Ripple =');
wp=input('Enter the Passband Frequency =');
ws=input('Enter the Stopband Frequency =');
fs=input('Enter the Sampling Frequency =');
w1=2*wp/fs;
w2=2*ws/fs;
[n,wn]=buttord(w1,w2,rp,rs,'s');
[z,p,k]=butter(n,wn);
sos=zp2sos(z,p,k);
freqz(sos);
```

a) Butterworth High Pass Filter

```
% Design of Butterworth High Pass Filter (Analog)
% Passband Ripple=rp=0.2
% Stopband Ripple=rs=40
% Passband Frequency=wp=2000
% Stopband Frequency=ws=3500
% Sampling Frequency=fs=8000
rp=input('Enter the Passband Ripple =');
rs=input('Enter the Stopband Ripple =');
wp=input('Enter the Passband Frequency =');
ws=input('Enter the Stopband Frequency =');
fs=input('Enter the Sampling Frequency =');
w1=2*wp/fs;
w2=2*ws/fs;
[n,wn]=buttord(w1,w2,rp,rs,'s');
[z,p,k]=butter(n,wn,'high');
sos=zp2sos(z,p,k);
freqz(sos);
```

Input Variables:

For Butterworth Low Pass Filter

1.Passband Ripple=rp=0.15
2.Stopband Ripple=rs=60
3.Passband Frequency=wp=1500
4.Stopband Frequency=ws=3000
5.Sampling Frequency=fs=7000

For Butterworth high Pass Filter

- 1.Passband Ripple=rp=0.2
  2.Stopband Ripple=rs=40

3.Passband Frequency=wp=2000
4.Stopband Frequency=ws=3500
5.Sampling Frequency=fs=8000

Results and Observations:

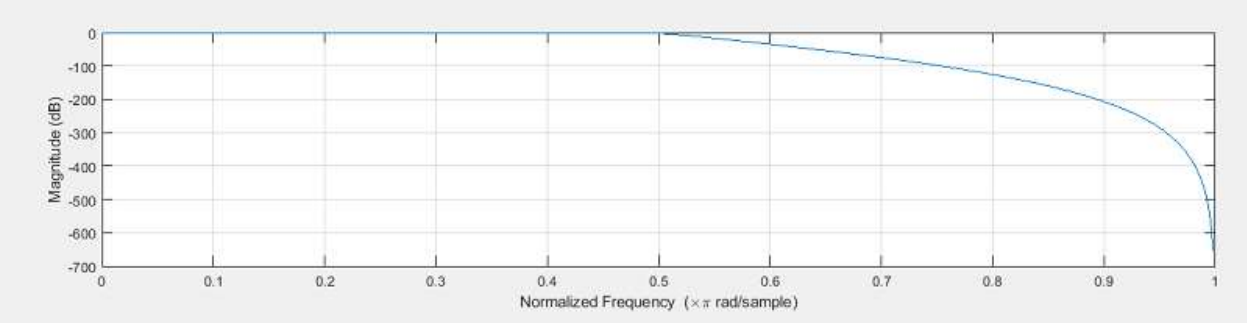


Fig 1. Magnitude Response of Butterworth Low Pass Filter

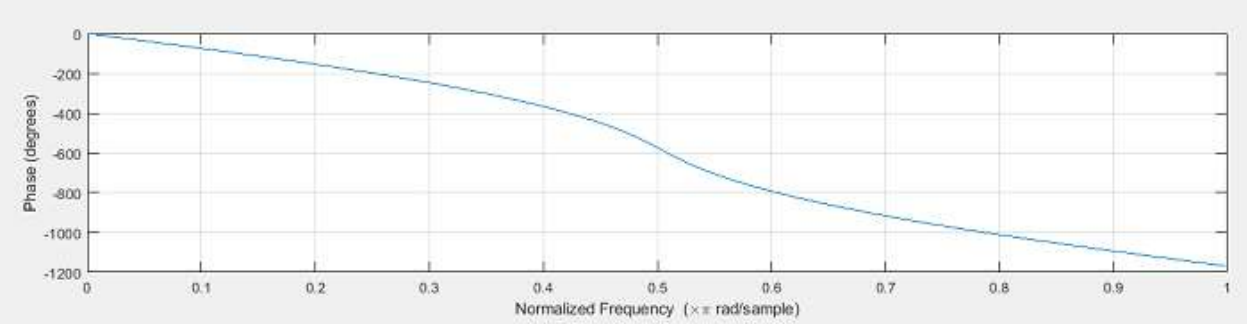


Fig 2 Phase Response of Butterworth Low Pass Filter

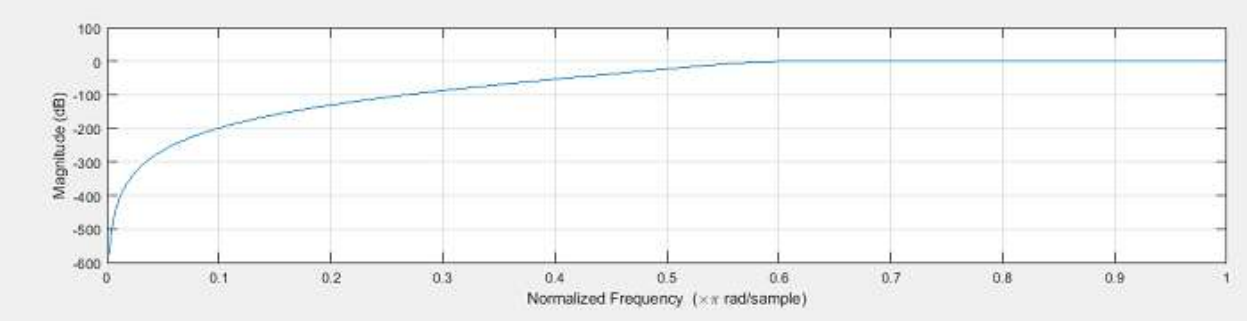


Fig 3. Magnitude Response of Butterworth High Pass Filter

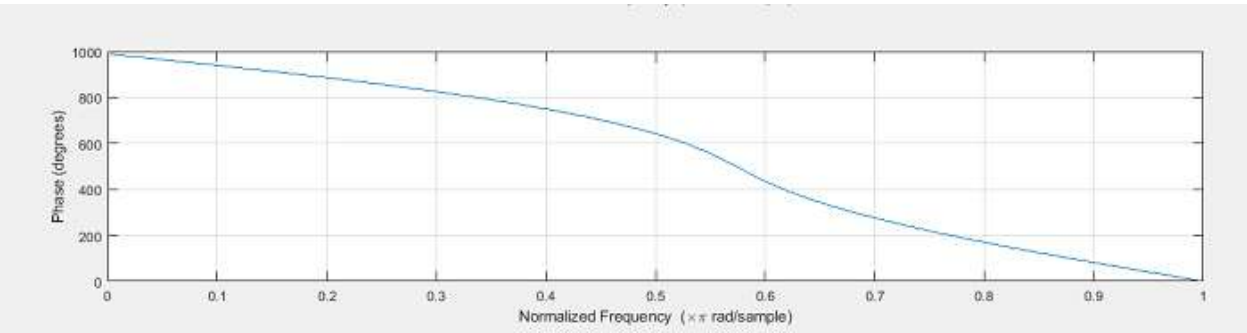


Fig 4. Phase Response of Butterworth High Pass Filter

Conclusion: Thus Magnitude Response and Phase Response of Butterworth Low Pass Filter and Butterworth High Pass Filter have been plotted.

Experiment No. 9

Aim: Write a MATLAB Program to find the autocorrelation of a given sequence and verification of its properties.

Tool used: MATLAB

Theory: A signal operation similar to signal convolution, but with completely different physical meaning, is signal correlation. The signal correlation operation can be performed either with one signal (autocorrelation) or between two different signals (crosscorrelation). Physically, signal autocorrelation indicates how the signal energy (power) is distributed within the signal, and as such is used to measure the signal power. Typical applications of signal autocorrelation are in radar, sonar, satellite, and wireless communications systems. Devices that measure signal power using signal correlation are known as signal correlators. There are also many applications of signal crosscorrelation in signal processing systems, especially when the signal is corrupted by another undesirable signal (noise) so that the signal estimation (detection) from a noisy signal has to be performed. Signal crosscorrelation can be also considered as a measure of similarity of two signals.

Given two discrete-time real signals (sequences) x[k] and y[k] . The autocorrelation and croosscorrelation functions are respectively defined by

$$R_{xx}[k] = \sum_{m=-\infty}^{\infty} x[m]x[m-k], \quad R_{yy}[k] = \sum_{m=-\infty}^{\infty} y[m]y[m-k]$$

$$R_{xy}[k] = \sum_{m=-\infty}^{\infty} x[m]y[m-k], \quad R_{yx}[k] = \sum_{m=-\infty}^{\infty} y[m]x[m-k]$$

where the parameter k is any integer from $-\infty<k<\infty$

Code:

```
% Matlab Code for Cross-Correlation of two sequence x(n) & h(n)
x=input('Enter the First Sequence x(n) =');
h=input('Enter the Second Sequence h(n) =');
lx=length(x);
lh=length(h);
L=lx+lh-1;
lc=-(L-1)/2:1:(L-1)/2;
y=xcorr(x,h);
subplot(2,2,1);
stem(x);
title('First Input Sequence x(n)');
subplot(2,2,2);
stem(h);
title('Second Input Sequence h(n)');
```

```
subplot(2,2,3);
stem(lc,y);
title('Cross-Correlation of x(n) with h(n)');
disp('Cross-Correlation of x(n) with h(n) is given as =');
disp(y);
y1=xcorr(h,x);
subplot(2,2,4);
stem(lc,y1);
title('Cross-Correlation of h(n) with x(n)');
disp('Cross-Correlation of h(n) with x(n) is given as =');
disp(y1);
```

Input Variables:

1. First Input Discrete Sequence x[n]=[1 2 3 4 5]
2. Second Input Discrete Sequence h[n]=[1 2 3 4 5]

Results and Observations:

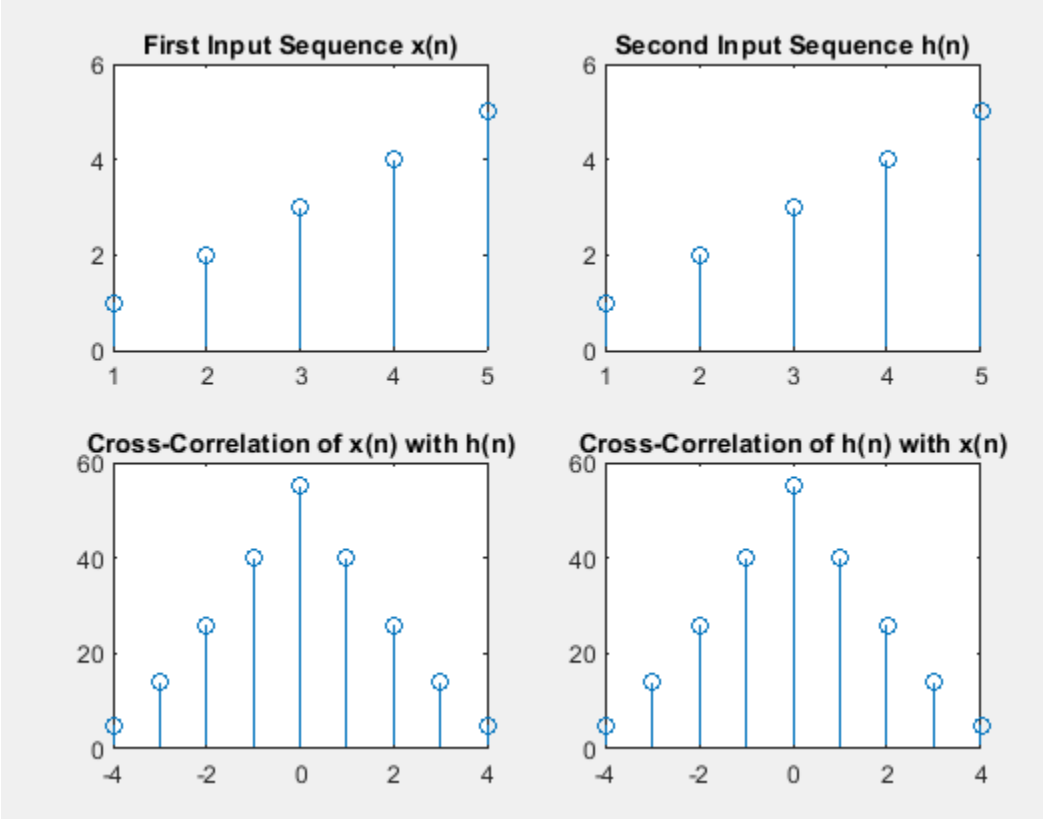


Fig 1 First Input Sequence x[n],Fig 2 Second Input Sequence h[n]
Fig 3 Cross Correlation of x[n] with h[n], Fig 4 Cross Correlation of h[n] with x[n]

Conclusion: Thus Autocorrelation of two similar discrete sequences has been plotted and properties of Autocorrelation have been verified..

Experiment No. 10

Aim: Write a MATLAB Program to design Chebyshev Low Pass Filter & High Pass Filter

Tool used: MATLAB

Theory: Chebyshev filters are used for distinct frequencies of one band from another. They cannot match the windows-sink filter's performance and they are suitable for many applications. The main feature of Chebyshev filter is their speed, normally faster than the windowed-sinc. Because these filters are carried out by recursion rather than convolution. The designing of the Chebyshev and Windowed-Sinc filters depends on a mathematical technique called as the Z-transform.

Type-I Chebyshev Filters

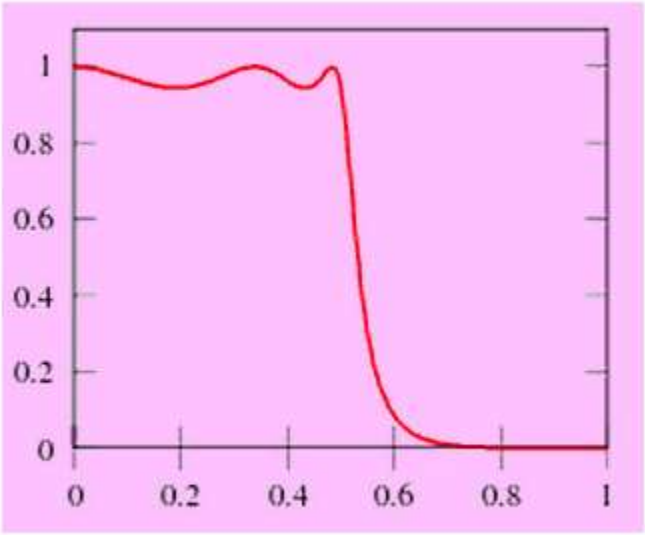
This type of filter is the basic type of Chebyshev filter. The amplitude or the gain response is an angular frequency function of the nth order of the LPF (low pass filter) and is equal to the total value of the transfer function Hn (jw)

$$Gn(w)=|Hn\ (j\omega)|=1\sqrt{(1+\epsilon 2Tn2()\ \omega/\omega o)}$$

Where,ε = ripple factor
ωo= cutoff frequency
Tn= Chebyshev polynomial of the nth order

The pass-band shows equiripple performance. In this band, the filter interchanges between -1 & 1 so the gain of the filter interchanges between max at G = 1 and min at G =1/√(1+ε2) . At the cutoff frequency, the gain has the value of 1/√(1+ε2) and remains to fall into the stop band as the frequency increases. The behavior of the filter is shown below. The cutoff frequency at -3dB is generally not applied to Chebyshev filters.

The pass-band shows equiripple performance. In this band, the filter interchanges between -1 & 1 so the gain of the filter interchanges between max at G = 1 and min at G =1/√(1+ε2) . At the cutoff frequency, the gain has the value of 1/√(1+ε2) and remains to fail into the stop band as the frequency increases. The behavior of the filter is shown below. The cutoff frequency at -3dB is generally not applied to Chebyshev filters.



Type-I Chebyshev Filter

Code:

a) Chebyshev Low Pass Filter

```
%Design of Chebyshev Type - II Low Pass Filter
%Design 6th order Low pass Chebyshev Type-II filter with 40 dB of stopband
%attenuation and stopband edge frequency of 300 Hz and data sampled at1000
%Hz. Plot its Magnitude and Phase response
%Stopband Attenuation=rs=40
%Order of the Filter=n=6
%Stopband Frequency=ws=300
%Sampling Frequency=fs=1000
n=input('Enter the Order of the Filter =');
rs=input('Enter the Stopband Attenuation in dB =');
ws=input('Enter the Stopband Edge Frequency in Hz=');
fs=input('Enter the Sampling Frequency in Hz =');
w2=2*ws/fs;
[b,a]=cheby2(n,rs,w2);
freqz(b,a);
```

b) Chebyshev High Pass Filter

```
% Design of Chebyshev Type - II High Pass Filter
%Design 9th order high pass Chebyshev Type-II filter with 20 dB of stopband
%attenuation and stopband edge frequency of 300 Hz and data sampled at 1000
%Hz. Convert the zero, pole and gain to second order section.Plot Magnitude
%and Phase response.
```

```
% Stopband Attenuation=rs=20
% Order of the Filter=n=9
% Stopband Frequency=ws=300
% Sampling Frequency=fs=1000
n=input('Enter the Order of the Filter =');
rs=input('Enter the Stopband Attenuation in dB =');
ws=input('Enter the Stopband Edge Frequency in Hz=');
fs=input('Enter the Sampling Frequency in Hz =');
w2=2*ws/fs;
[z,p,k]=cheby2(n,rs,w2,'high');
sos=zp2sos(z,p,k);
freqz(sos);
%fvtool(sos,'Analysis','freq');
```

Input Variables:

For Chebyshev Low Pass Filter

1. Stopband Attenuation=rs=40
2. Order of the Filter=n=6
3. Stopband Frequency=ws=300
4. Sampling Frequency=fs=1000

For Chebyshev High Pass Filter

1.Stopband Attenuation=rs=20
2.Order of the Filter=n=9
3.Stopband Frequency=ws=300
4.Sampling Frequency=fs=1000

Results and Observations:

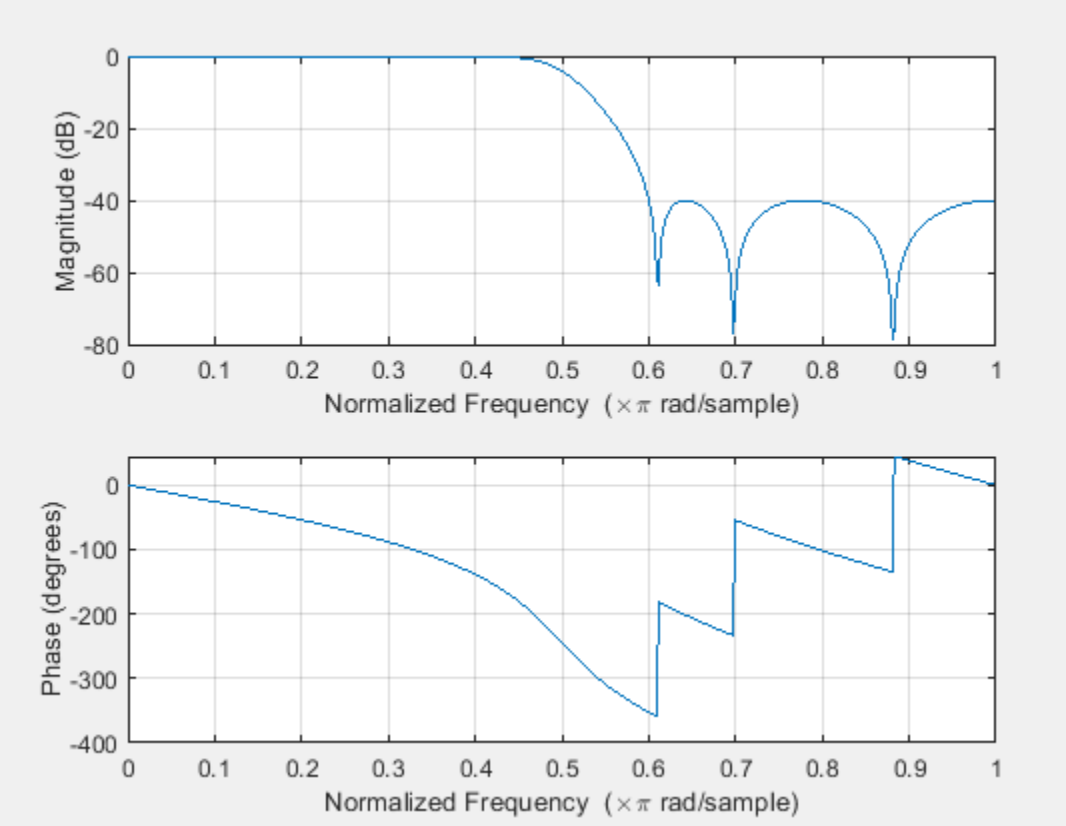


Fig 1 Magnitude Response of Chebyshev Low Pass Filter

Fig 2 Phase Response of Chebyshev Low Pass Filter

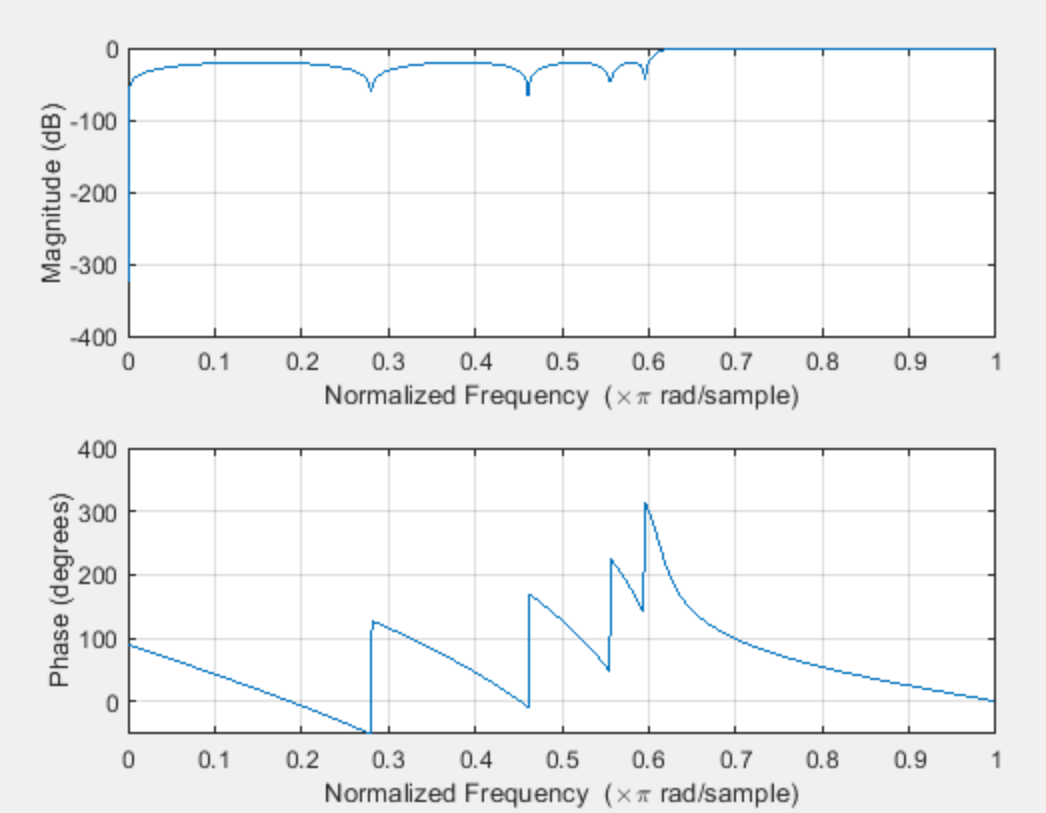


Fig 1 Magnitude Response of Chebyshev High Pass Filter

Fig 2 Phase Response of Chebyshev High Pass Filter

Conclusion: Thus the Magnitude Response and Phase Response of Chebyshev low pass and high pass Filter has been plotted.