7

EXPERIMENT

# IIR Filter Design

## AIM

(a) To design a Low Pass Butterworth IIR filter and to plot its frequency response.

(b) To design a High Pass Butterworth IIR and to plot its frequency response.

(c) To Demonstrate the designed filters by filtering a composite frequency signal and plotting the filtered output

## THEORY

A low-pass Butterworth filter is an analog all-pole filter with a squared magnitude response given by,

$$\left|H_C(j\Omega)\right|^2 = \frac{1}{1+\left(\frac{j\Omega}{j\Omega_C}\right)^{2N}} = \frac{1}{1+\epsilon^2\left(\frac{j\Omega}{j\Omega_P}\right)^{2N}}$$

Where $N$ is the order of the filter (number of poles in the system function), $\Omega_C$ is the 3-dB cut-off frequency and $\Omega_P$ is the passband edge frequency

$$\epsilon = \left(\frac{\Omega_P}{\Omega_C}\right)^N$$

As the order ($N$) increases, the transition band becomes narrower. i.e, the filter characteristics becomes sharper. The frequency response of the Butterworth filter decreases monotonically with increasing $\Omega$. $\left|H_C(j\Omega)\right|^2$ is monotonic in both the passband and stopband.

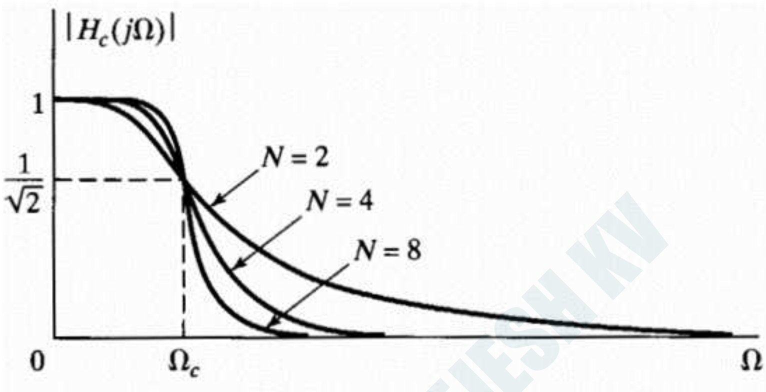



Figure 7.1: Magnitude Response of an analog Butterworth filter

The poles of the Butterworth filter are symmetrically located with respect to the imaginary axis. A pole never falls on the imaginary axis, and one occurs on the real axis for odd N odd, but not for even N.

The poles of the magnitude-squared function always occur in pairs; i.e., if there is a pole at $s = s_k$, then a pole also occurs at $s = -s_k$. Therefore, to construct $H_C(s)$ from the magnitude- squared function, we would choose one pole from each such pair: To obtain a stable and causal filter, we should choose the poles on the left-half-plane part of the $s$-plane. The angular spacing between the poles on the circle is $\pi/N$ radians For example, for $N = 3$, the poles are spaced by $\pi/3$ radians as shown below.

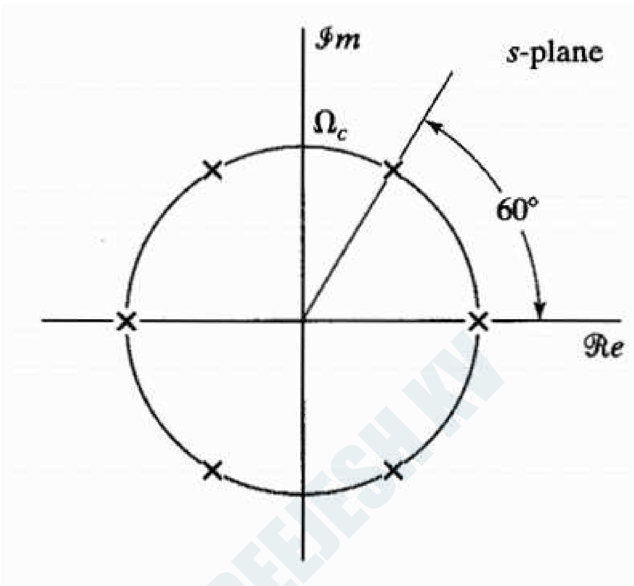



Figure 7.2: Pole locations of $H_C(s)H_C(-s)$ for a third order analog Butterworth filter

### Design of IIR digital filters from analog filters

The traditional approach for designing discrete-time IIR filters is through transformation of a continuous-time filter into a discrete-time filter meeting prescribed specifications.

In designing a discrete-time filter by transforming a prototype continuous-time filter, the specifications for the continuous-time filter are obtained by a transformation of the specifications for the desired discrete-time filter.

Then the system function $H_C(s)$ or the impulse response $h_C(t)$ of the continuous-time filter is obtained through one of the established approximation methods.
Next, the system function $H(z)$ or impulse response $h[n]$ for the discrete-time filter is obtained by applying a transformation to $H_C(s)$ or $h_C(t)$.
In such transformations, the following essential properties of the continuous-time frequency response should be preserved in the frequency response of the resulting discrete-time filter.

- The imaginary axis of the s-plane should map onto the unit circle of the z-plane
- A stable continuous-time filter should be transformed to a stable discrete-time filter. i.e., if the continuous-time system has poles only in the left half of the $s$-plane, then the discrete-time filter must have poles only inside the unit circle in the $z$-plane

**Bilinear transformation**

This transformation avoids aliasing by using an algebraic transformation between the variables $s$ and $z$ that maps the entire $j\Omega$-axis in the $s$-plane to one revolution of the unit circle in the $z$-plane.
The transformation is non-linear, since $-\infty \le \Omega \le \infty$ maps to $-\pi \le \omega \le \pi$. Therefore, Bilinear transformation is used only when warping of frequency axis is acceptable. The transformation is done by replacing $s$ with,

$$s = \frac{2}{T_d}\left(\frac{1-z^{-1}}{1+z^{-1}}\right)$$

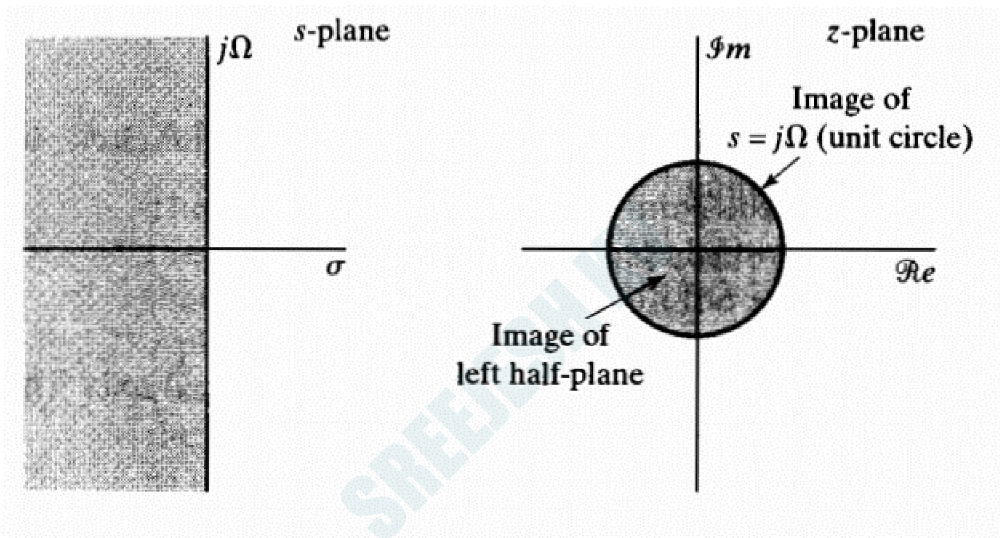



Figure 7.3: Bilinear Transformation

Thus, BLT avoids aliasing by mapping the entire imaginary axis of $s$ plane to one complete revolution of the unit circle of $z$-plane. The price paid for this is the non-linear compression of frequency axis.

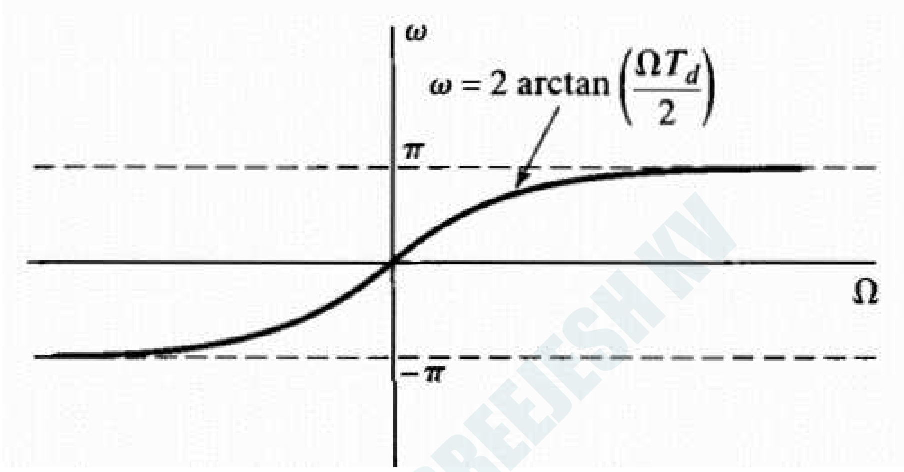



Figure 7.4: Mapping of the continuous-time frequency axis onto the discrete-time frequency axis by bilinear transformation

The relation between continuous-time frequency($\Omega$) and discrete-time frequency ($\omega$) for bilinear transformation is given by,

$$\omega = 2\arctan\left(\frac{\Omega T_d}{2}\right)$$

### Prewarping in Bilinear Transformation

The relationship between the analog frequency $\Omega$ and the digital frequency $\omega$ is almost linear for small values of $\omega$, but becomes nonlinear for large values of $\omega$ leading to a distortion (or warping) of the digital frequency response.

This effect is normally compensated for by **prewarping** the analog filter before applying the bilinear transformation. i.e., we prewarp one or more critical frequencies before applying the Bilinear Transformation. For example, for a lowpass filter we often prewarp the cutoff frequency as follows:

$$\Omega = \frac{2}{T_d}\tan(\omega/2)$$

## MATLAB FUNCTIONS USED

**buttord**

Butterworth filter order and cutoff frequency

```
[n,Wn] = buttord(Wp,Ws,Rp,Rs) returns the lowest order, n,
of the digital Butterworth filter with no more than Rp dB
of passband ripple and at least Rs dB of attenuation in the
stopband.
Wp and Ws are respectively the passband and stopband edge
frequencies of the filter, normalized from 0 to 1, where
1 corresponds to π rad/sample.  The scalar (or vector) of
corresponding cutoff frequencies, Wn, is also returned.
```

To design a Butterworth filter, use the output arguments n and Wn as inputs to **butter**

**butter**

Butterworth filter design

[b,a] = butter(n,Wn) returns the transfer function coefficients of an nth-order lowpass digital Butterworth filter with normalized cutoff frequency Wn.
[b,a] = butter(n,Wn,ftype) designs a lowpass, highpass, bandpass, or bandstop Butterworth filter, depending on the value of ftype and the number of elements of Wn. The resulting bandpass and bandstop designs are of order 2n.

**freqz**

Frequency response of digital filter

[h,w] = freqz(b,a,n) returns the n-point frequency response vector h and the corresponding angular frequency vector w for the digital filter with transfer function coefficients stored in b and a.
freqz(___) with no output arguments plots the frequency response of the filter.
[h,f]=freqz(___,n,fs) returns the frequency response vector h and the corresponding physical frequency vector f for a digital filter designed to filter signals sampled at a rate fs.

**Note:**

For digital filter design, **butter** uses **bilinear** to convert the analog filter into a digital filter through a bilinear transformation with frequency prewarping

## ALGORITHM

Step 1. Start

Step 2. Input the pass band ripple rp, stop band ripple rs, pass band frequency fp and stop band frequency fs and the sampling frequency fsamp

Step 3. Convert the frequencies to digital and normalize them.

Step 4. Find the cut off frequency and lowest order N of the filter satisfying the given specifications using MATLAB's **buttord** function

Step 5. Using the above values find the coefficients of the low pass and high pass filters using **butter** function

Step 6. Find the frequency response of the filters and plot them

Step 7. Create a composite signal as the sum of two sinusoids: one having a frequency less than fp and the other having a frequency higher than fs

Step 8. Filter the composite signal using the designed lowpass and highpass filters and observe the output in time domain.

Step 9. Stop

## PROGRAM

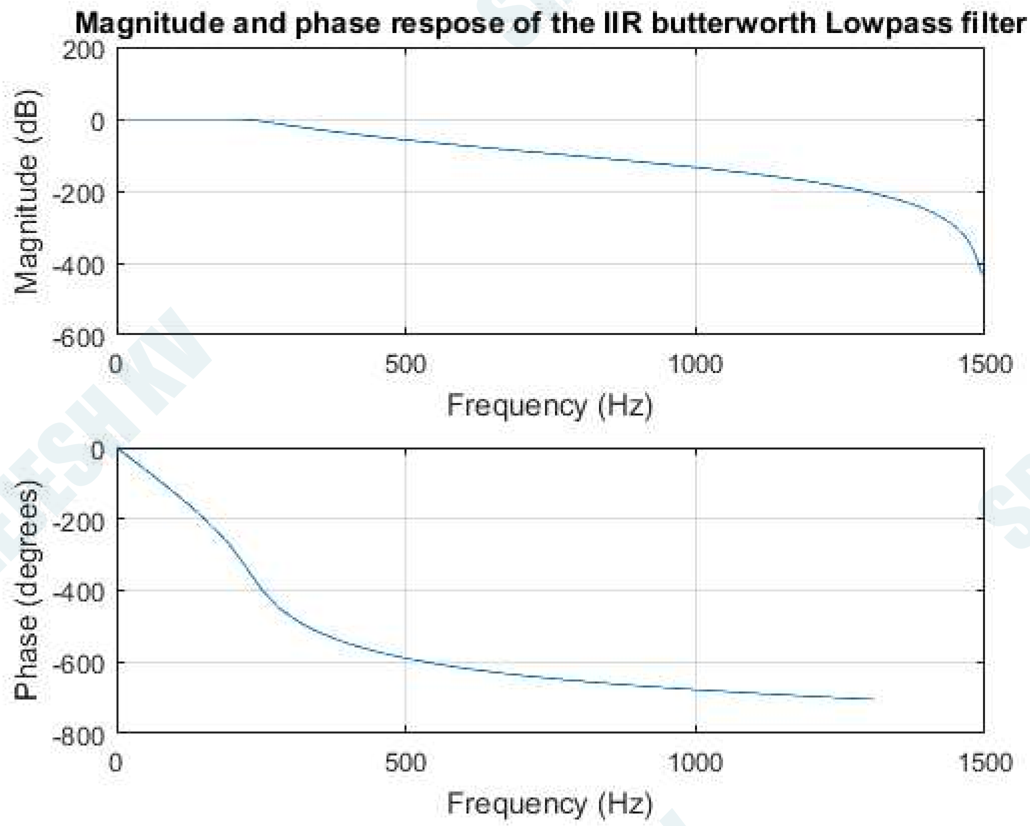
```
%Title: Program to
% 1) Design a Low Pass Butterworth IIR filter and to plot its ...
    frequency response.
% 2) Design a High Pass Butterworth IIR and to plot its frequency ...
    response.
% 3) Demonstrate the designed filters by filtering a composite
% frequency signal and plotting the filtered output

%Author: Sreejesh K V, Dept. of ECE, GCEK
%Date: 05/11/2022

clc;
clear;
close all;

fp=200;%the pass band edge frequency in Hz
fs=400;%the stop band edge frequency in Hz
rp=1;% peak-to-peak passband ripple in dB
rs=40;%the minimum stopband attenuation in dB
fsamp=3000;% Sampling frequency in Hz

wp=2*fp/fsamp;%normalized digital passband edge frequency(=dig ...
    frequency/pi)
ws=2*fs/fsamp;%normalized digital stopband edge frequency

% --- Designing the Low Pass Butterworth Filter --- %
[N,wn]=buttord(wp,ws,rp,rs);% returns the lowest order N of a digital
%Butterworth filter satisfying the given PB & SB ripple conditions.
%Wp and Ws are the PB & SB edge frequencies.
% Wn is the Butterworth natural(3dB) frequency


[b,a]=butter(N,wn,'low');% designs an Nth order lowpass digital
%Butterworth filter and returns the filter coefficients in length
% N+1 vectors B (numerator) and A (denominator)
figure,
freqz(b,a,500,fsamp);%plotting the frequency response of the LPF
```

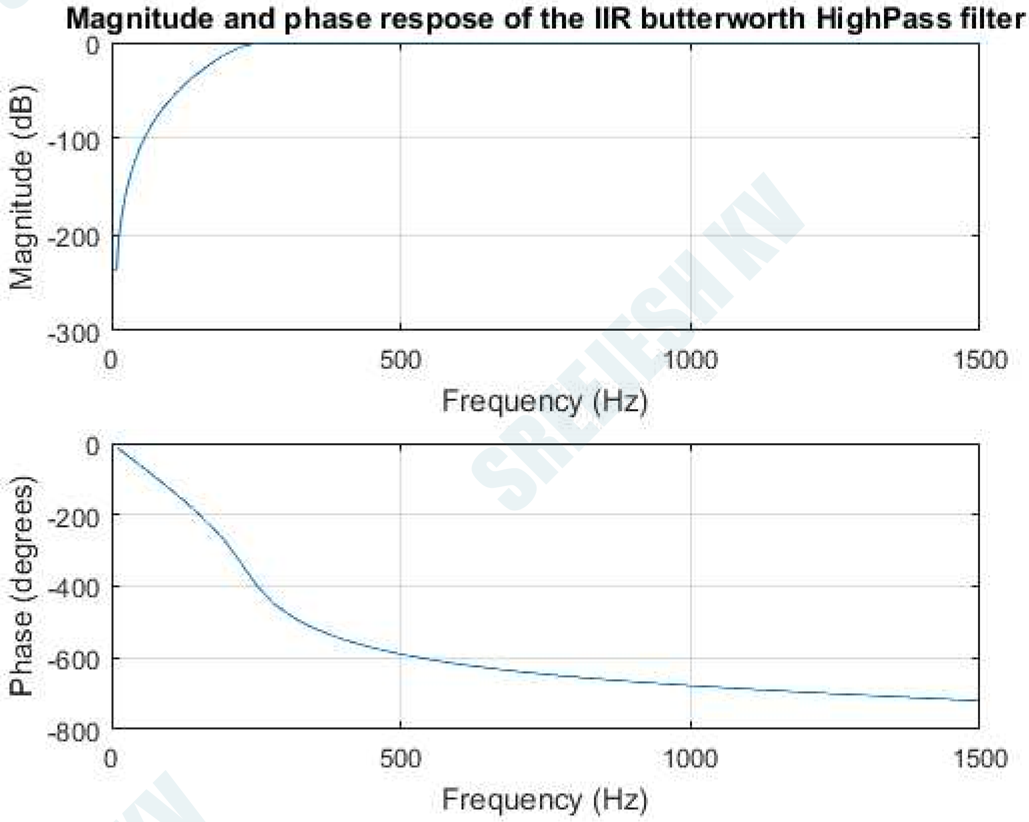
```
title('Magnitude and phase respose of the IIR butterworth Lowpass ...
    filter');

%--- Designing the High Pass Butterworth Filter --- %
[bh,ah]=butter(N,wn,'high');% designs an Nth order lowpass digital
%Butterworth filter and returns the filter coefficients in length
% N+1 vectors B (numerator) and A (denominator)

figure,
freqz(bh,ah,500,fsamp);%plotting the frequency response of the HPF
title('Magnitude and phase respose of the IIR butterworth ...
    HighPass filter');

%--- Demonstration of the Filters --%%
f1=100;%100Hz : in the passband of LPF and in the SB of HPF
f2=500;%500Hz: in the stopband of LPF and in the PB of HPF
T=1/f1;
t=0:1/fsamp:3*T;
s1=sin(2*pi*f1*t);%100Hz signal(approximation using only 3 periods)
s2=sin(2*pi*f2*t);%500Hz signal(approximation)
s=s1+s2;%the composite signal containing 100Hz and 500Hz ...
    components(mainly)

%---Filtering Process--%
y=filter(b,a,s);%low pass filtering
y1=filter(bh,ah,s);%highpass filtering

%---Plotting the signals--%
figure;
subplot(2,2,1);
plot(t,s1,'r','LineWidth',2);
hold on
plot(t,s2);
legend('s1','s2');
title(['Signals s1 ( f = ' num2str(f1) ' Hz) & s2 ( f = ' ...
    num2str(f2) ' Hz)  ']);
xlabel('time');
ylabel('Amplitude');

subplot(2,2,2);
plot(t,s);
title('Composite signal=s1+s2');
xlabel('time');
ylabel('Amplitude');

subplot(2,2,3);
plot(t,y);
title('LowPass Filtered Output');
xlabel('time');
ylabel('Amplitude');

```

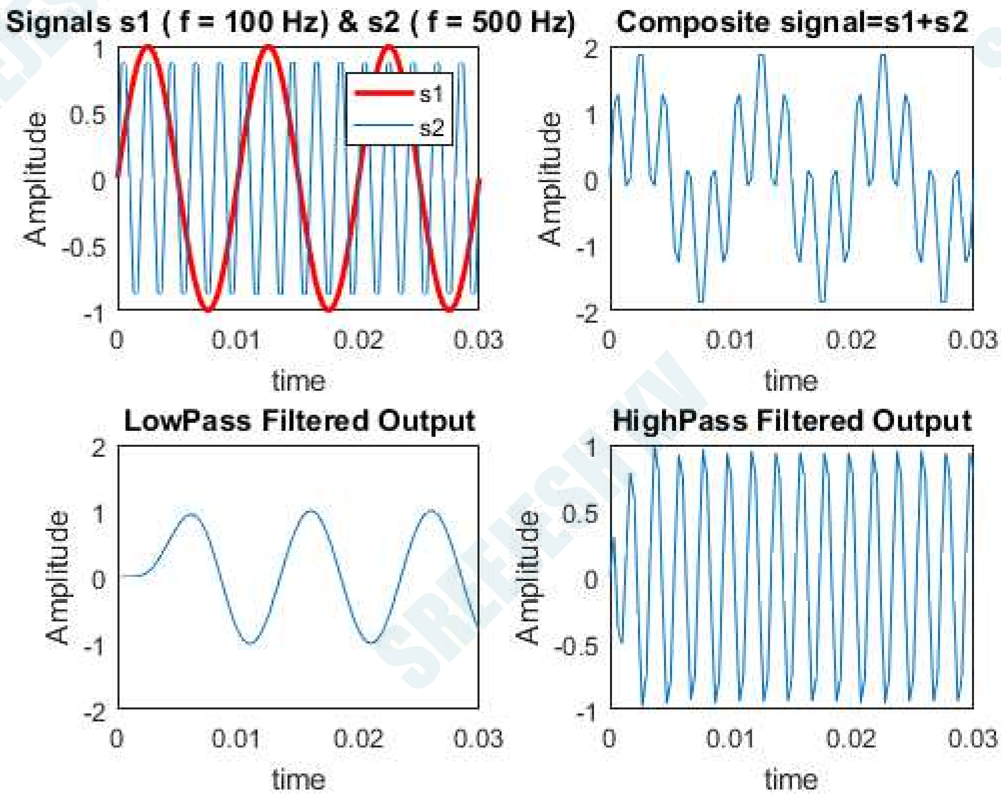
```
subplot(2,2,4);
plot(t,y1);
title('HighPass Filtered Output');
xlabel('time');
ylabel('Amplitude');
```

## OUTPUT & OBSERVATIONS

Figure Window Outputs:






## RESULTS

(a) A Low Pass Butterworth IIR filter was designed using MATLAB and its frequency response was plotted

(b) A High Pass Butterworth IIR filter was designed using MATLAB and its frequency response was plotted

(c) The designed filters were demonstrated by filtering a composite frequency signal and plotting the filtered output