BMS INSTITUTE OF TECHNOLOGY & MANAGEMENT

Doddaballapur Road, Avalahalli, Yelahanka, Bengaluru, Karnataka 560064

DEPARTMENT OF ELECTRONICS & TELECOMMUNICATION ENGINEERING

Academic Year 2022-23

REPORT

On

"Implementation of DSP concepts using MATLAB"

Submitted By

USN	Name of the Student	Marks Awarded Max Marks: 05
1BY20ET048	S VARSHA	
Signature of faculty		

Course: Digital Signal Processing

Course Code: 18EC52

Course Outcome: Perform in a group to design and execute an application of any digital signal processing operations using modern tools.

Under the guidance of

Dr. Thejaswini S

Asst. Professor, Dept. of ETE

DESIGNING A FILTER USING FILTER DESIGNER APP

- Filter Designer is a powerful graphical user interface (GUI) in Signal Processing Toolbox for designing and analyzing filters.
- Filter Designer enables you to quickly design digital FIR or IIR filters by setting filter performance specifications, by importing filters from your MATLAB workspace or by adding, moving, or deleting poles and zeros.
- Filter Designer also provides tools for analyzing filters, such as magnitude and phase response plots and pole-zero plots.

Design a low pass filter that passes all frequencies less than or equal to 20% of the Nyquist frequency (half the sampling frequency) and attenuates frequencies greater than or equal to 50% of the Nyquist frequency. Design an FIR Equiripple filter with these specifications:

Passband attenuation 1 dB

Stopband attenuation 80 dB

A passband frequency 0.2 [Normalized (0 to 1)]

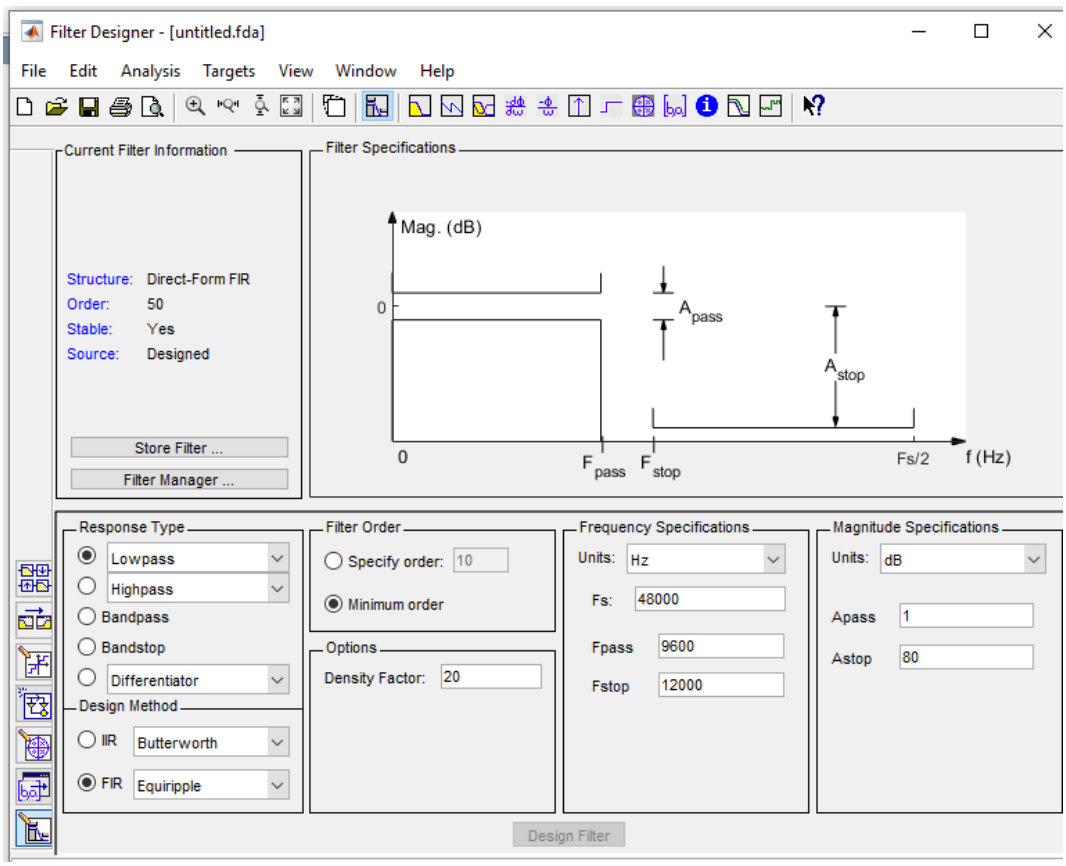
A stopband frequency 0.5 [Normalized (0 to 1)]

DESIGN STEPS:

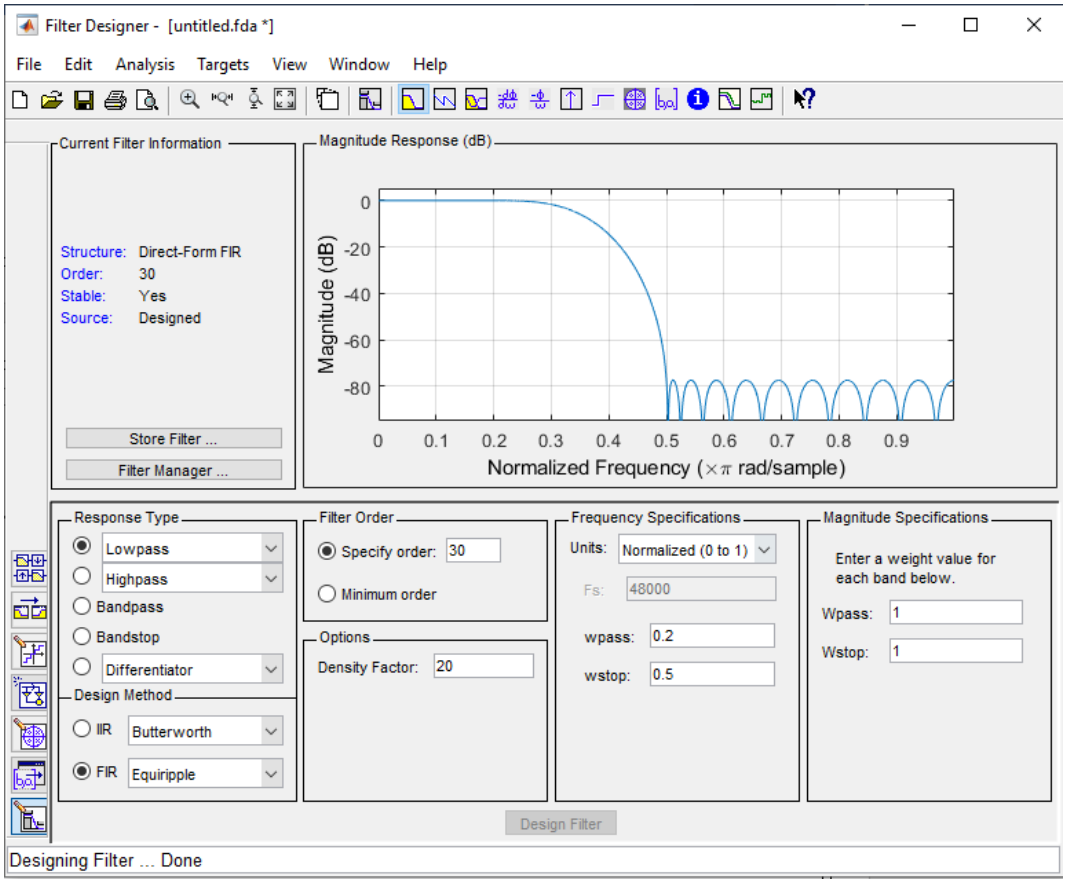
1. Type “filterDesigner” at the MATLAB command prompt:

```
>> filterDesigner
```

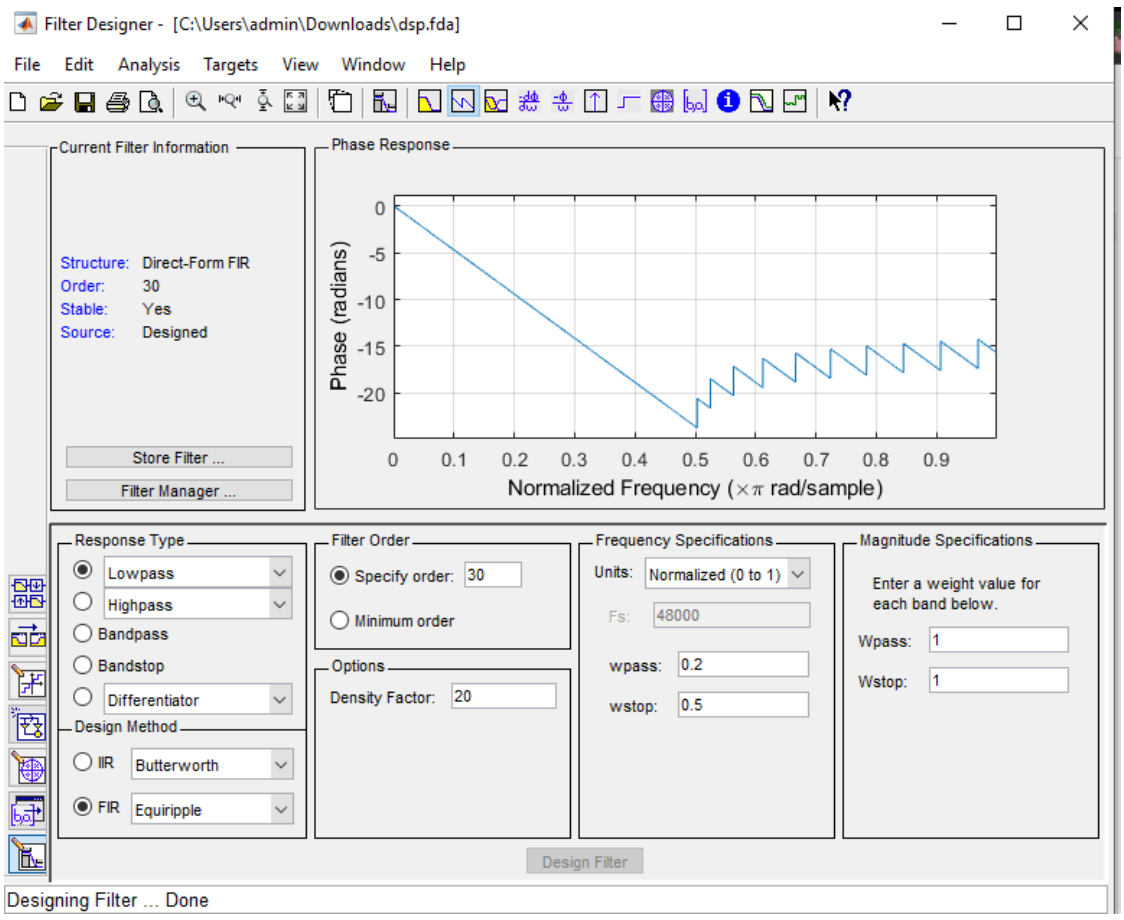
2. Select Lowpass from the dropdown menu under Response Type and Equiripple under FIR Design Method.
3. Select Specify order in the Filter Order area and enter 30.
4. Leave Density Factor value at 20.
5. Select Normalized (0 to 1) in the Units pull down menu in the Frequency Specifications area.
6. Enter 0.2 for wpass and 0.5 for wstop in the Frequency Specifications area.
7. Leave Wpass and Wstop values at 1.
8. After setting the design specifications, click the Design Filter button at the bottom of the GUI to design the filter.



Filter Designer App

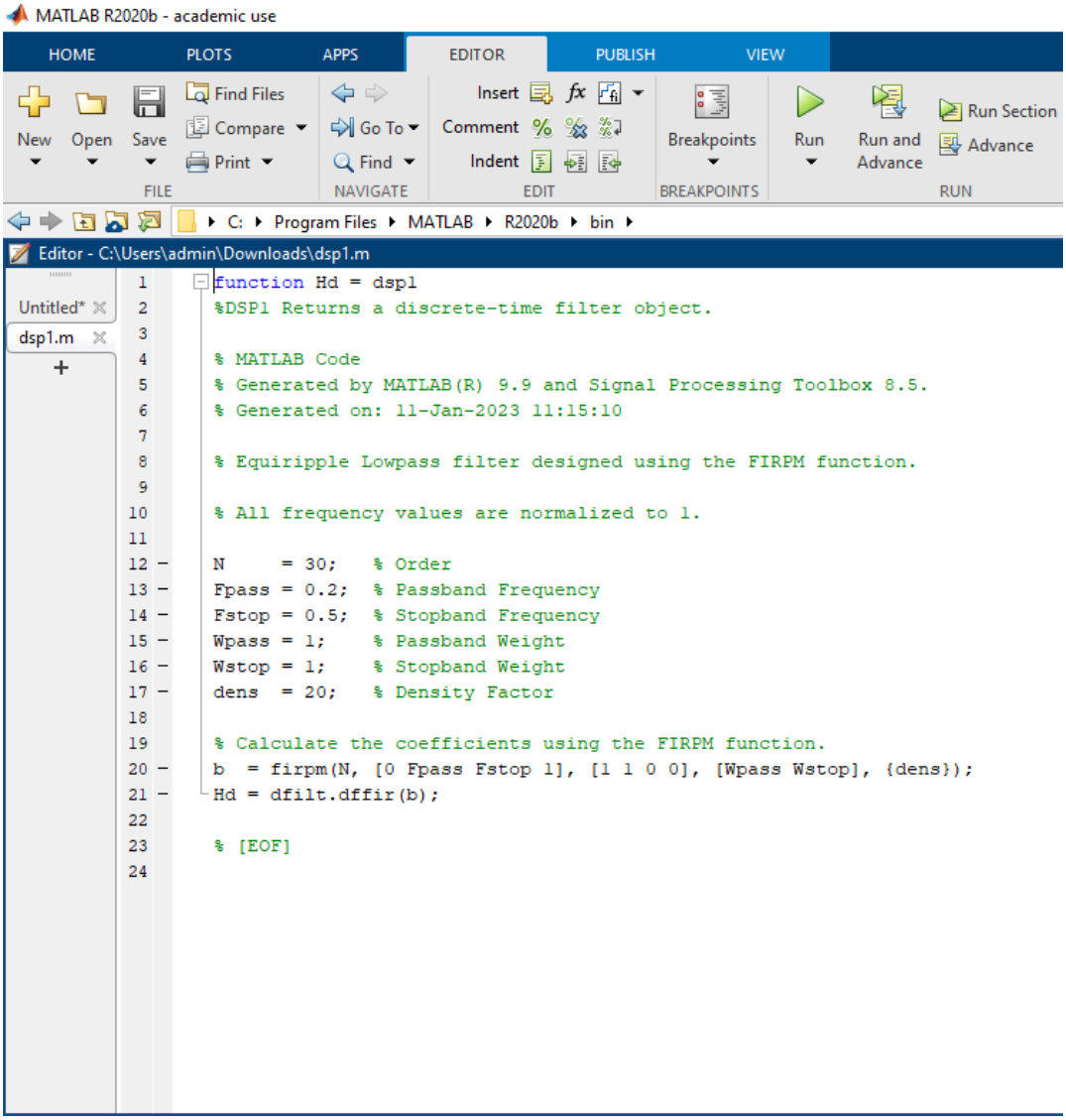


Magnitude response of designed filter



Phase response of designed filter

Generated MATLAB code for the designed filter



VISUALIZATION OF DESIGNED FILTER USING fvtool

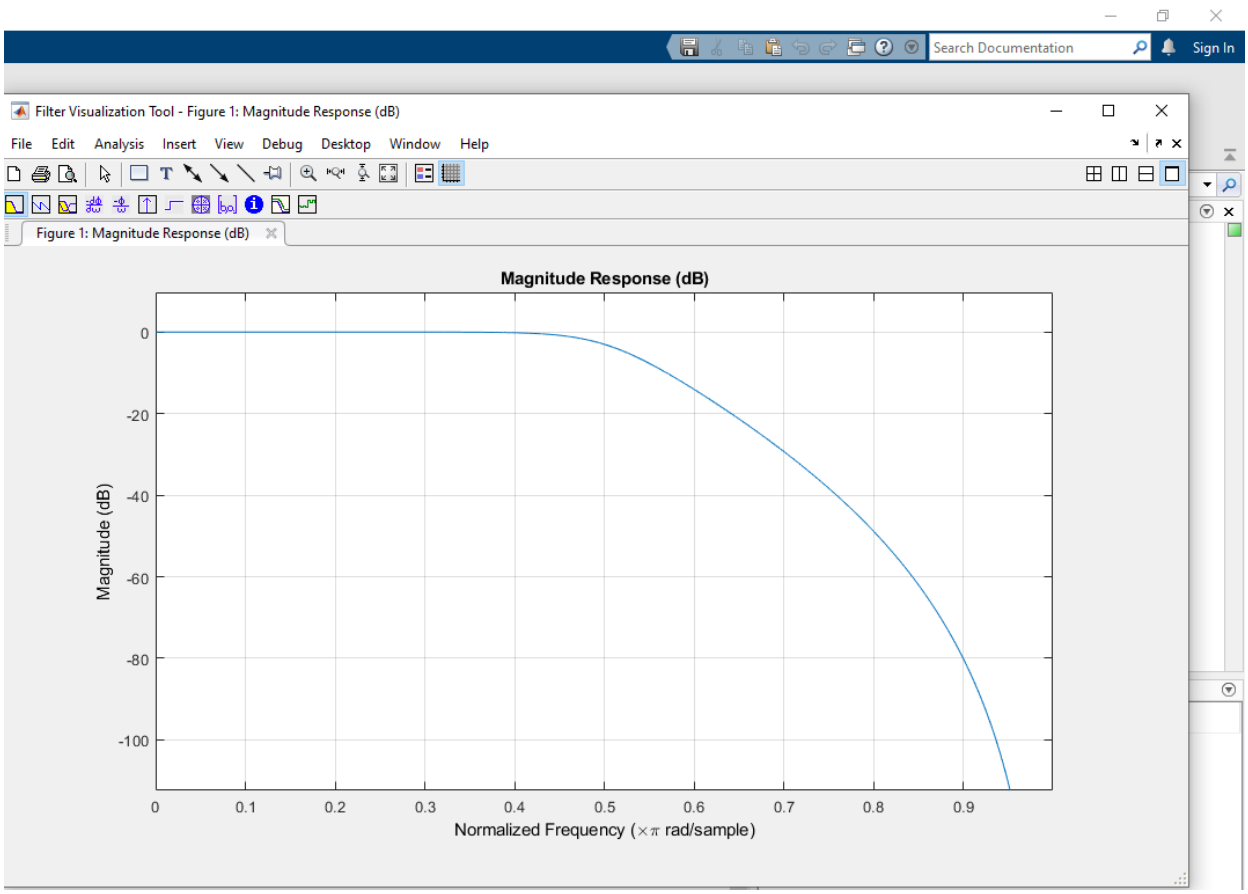
- Filter Visualization Tool is an interactive tool that enables you to display the magnitude, phase response, group delay, impulse response, step response, pole-zero plot, and coefficients of a filter.
- fvtool(b,a) opens FVTool and displays the magnitude response of the digital filter defined with numerator b and denominator a.
- fvtool(d) opens FVTool and displays the magnitude response of a digital filter, d.

MATLAB Code for EXAMPLE 1:

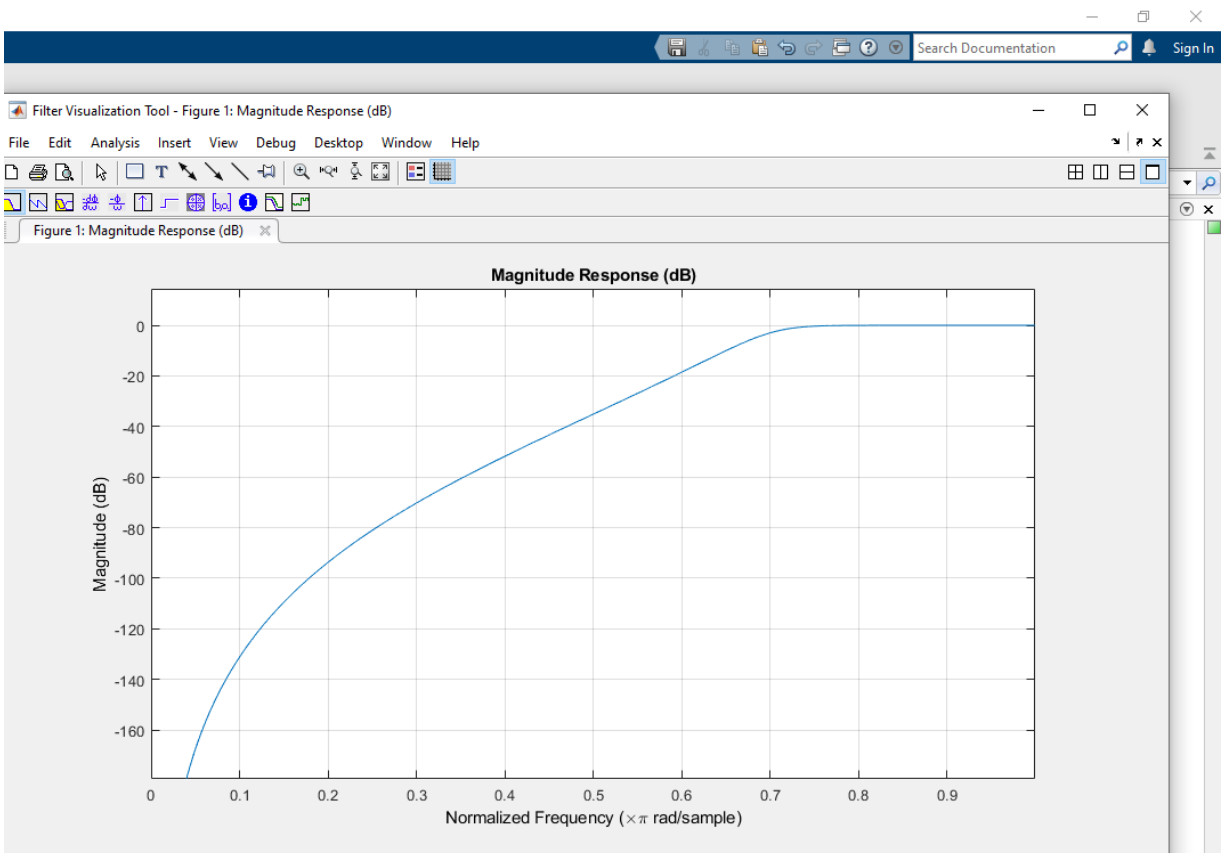
```
% Magnitude Response of an IIR filter.

[b,a] = butter(5,.5);

h1 = fvtool(b,a);

Hd = dfilt.df1(b,a); % Discrete-time filter
(DFILT) object

h2 = fvtool(Hd);
```

MATLAB Code for EXAMPLE 2:

```
Design a Butterworth highpass IIR filter,
represent its coefficients

% using second order sections, and visualize the
filter with fvtool.

[z,p,k] = butter(6,0.7,'high');

SOS = zp2sos(z,p,k);

fvtool(SOS) % Visualize the filter
```



Magnitude Response of an IIR filter (Example 1)



Magnitude Response of a butterworth highpass IIR filter

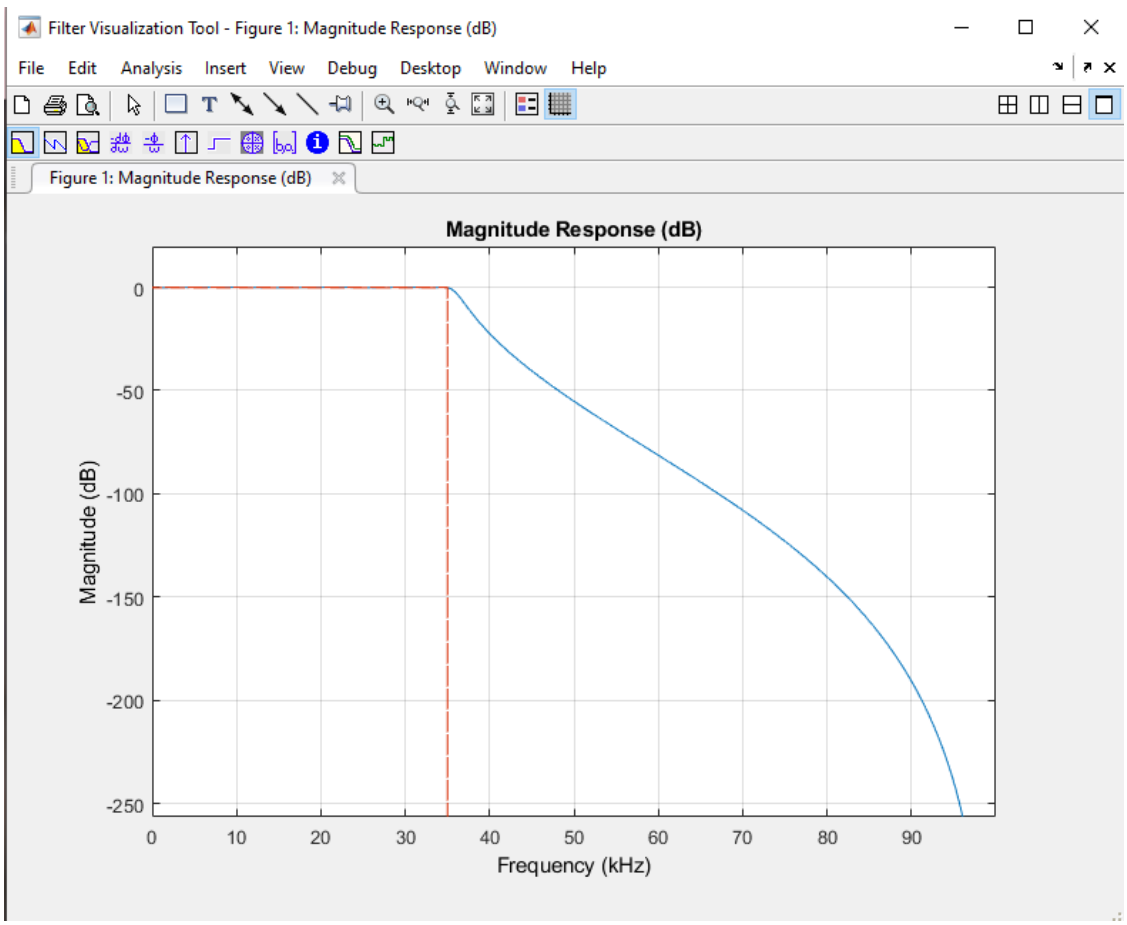
(Example 2)

DESIGNING OF FILTER USING COMMAND designfilt

Design a lowpass IIR filter with order 8, passband frequency 35 kHz, and passband ripple 0.2 dB. Specify a sample rate of 200 kHz. Visualize the magnitude response of the filter.

MATLAB Code:

```
lpFilt =
designfilt('lowpassiir','FilterOrder',8, ...
'PassbandFrequency',35e3,'PassbandRipple',0.2,
...
'SampleRate',200e3);
fvtool(lpFilt)
```



Magnitude Response of a lowpass IIR filter

CASE STUDY

Removing the 60 Hz Hum from a Signal

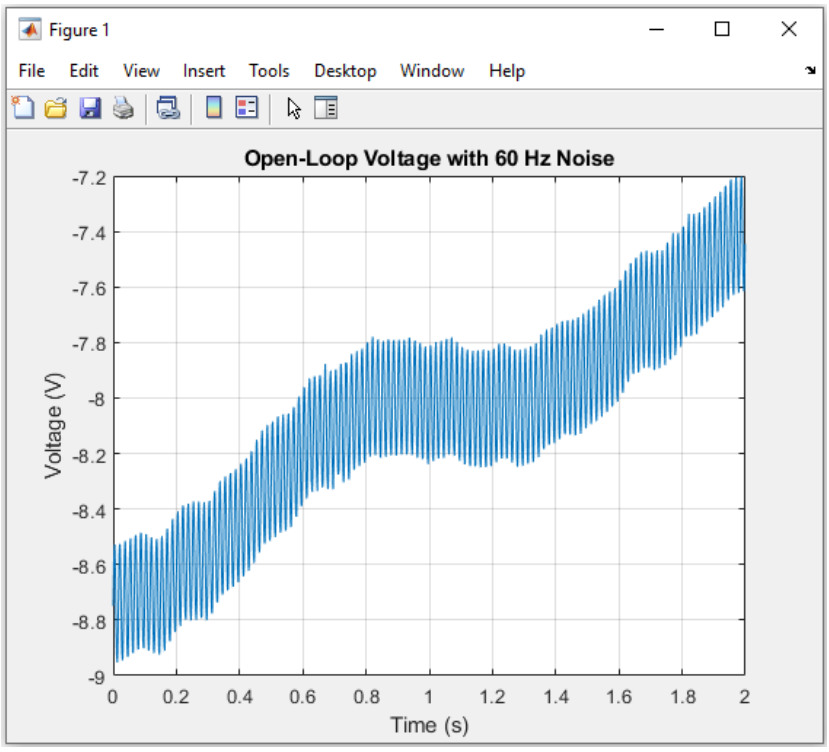
- Alternating current in the United States and several other countries oscillates at a frequency of 60 Hz. Those oscillations often corrupt measurements and have to be subtracted.

STEP 1:

The open-loop voltage across the input of an analog instrument in the presence of 60 Hz power-line noise is plotted. The voltage is sampled at 1 kHz.

MATLAB Code:

```
load openloop60hertz, openLoop = openLoopVoltage;
Fs = 1000;
t = (0:length(openLoop)-1)/Fs;
plot(t,openLoop)
ylabel('Voltage (V)')
xlabel('Time (s)')
title('Open-Loop Voltage with 60 Hz Noise')
grid
```

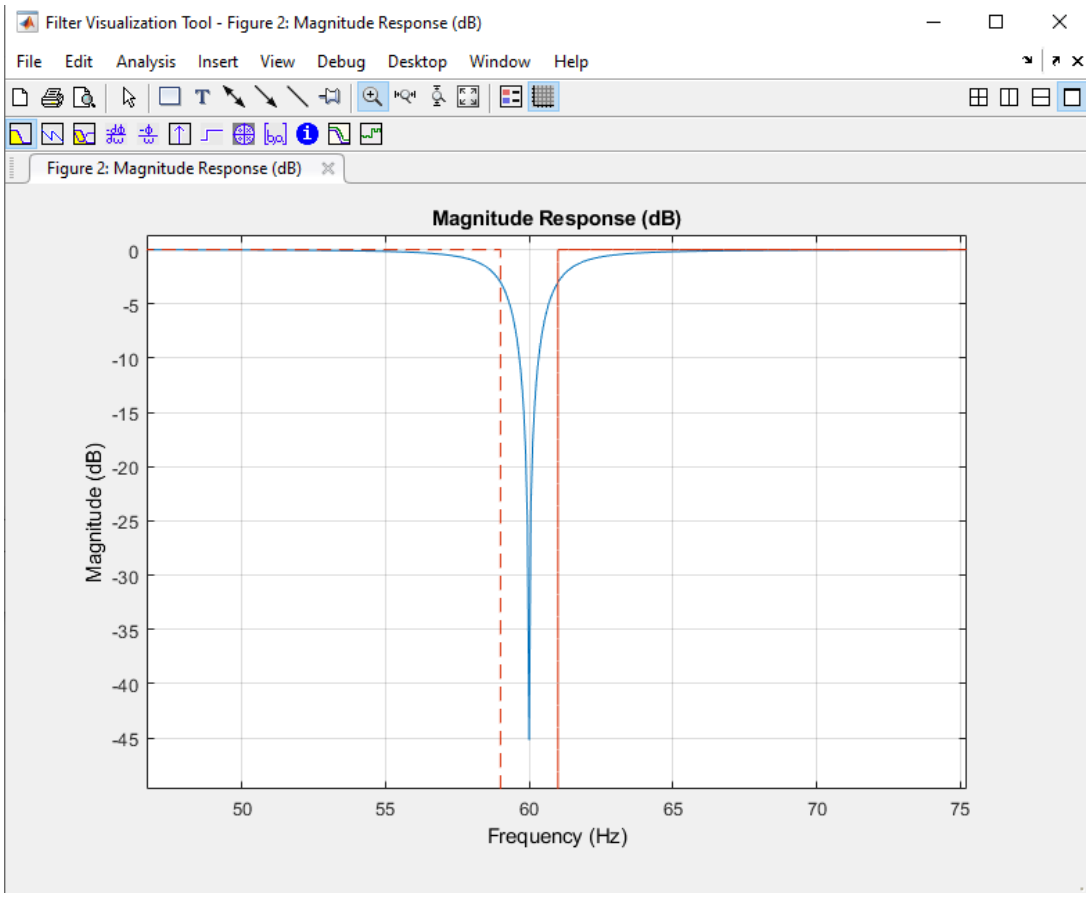


STEP 2:

- Eliminate the 60 Hz noise using a Butterworth notch filter.
  Use designfilt to design the filter. The width of the notch is defined by the 59 to 61 Hz frequency interval. The filter removes at least half the power of the frequency components lying in that range.
- Plot the frequency response of the filter. Note that this notch filter provides up to 45 dB of attenuation.

MATLAB Code:

```
d = designfilt('bandstopiir','FilterOrder',2, ...
'HalfPowerFrequency1',59,'HalfPowerFrequency2',61,
...
'DesignMethod','butter','SampleRate',Fs);
fvtool(d,'Fs',Fs)
```



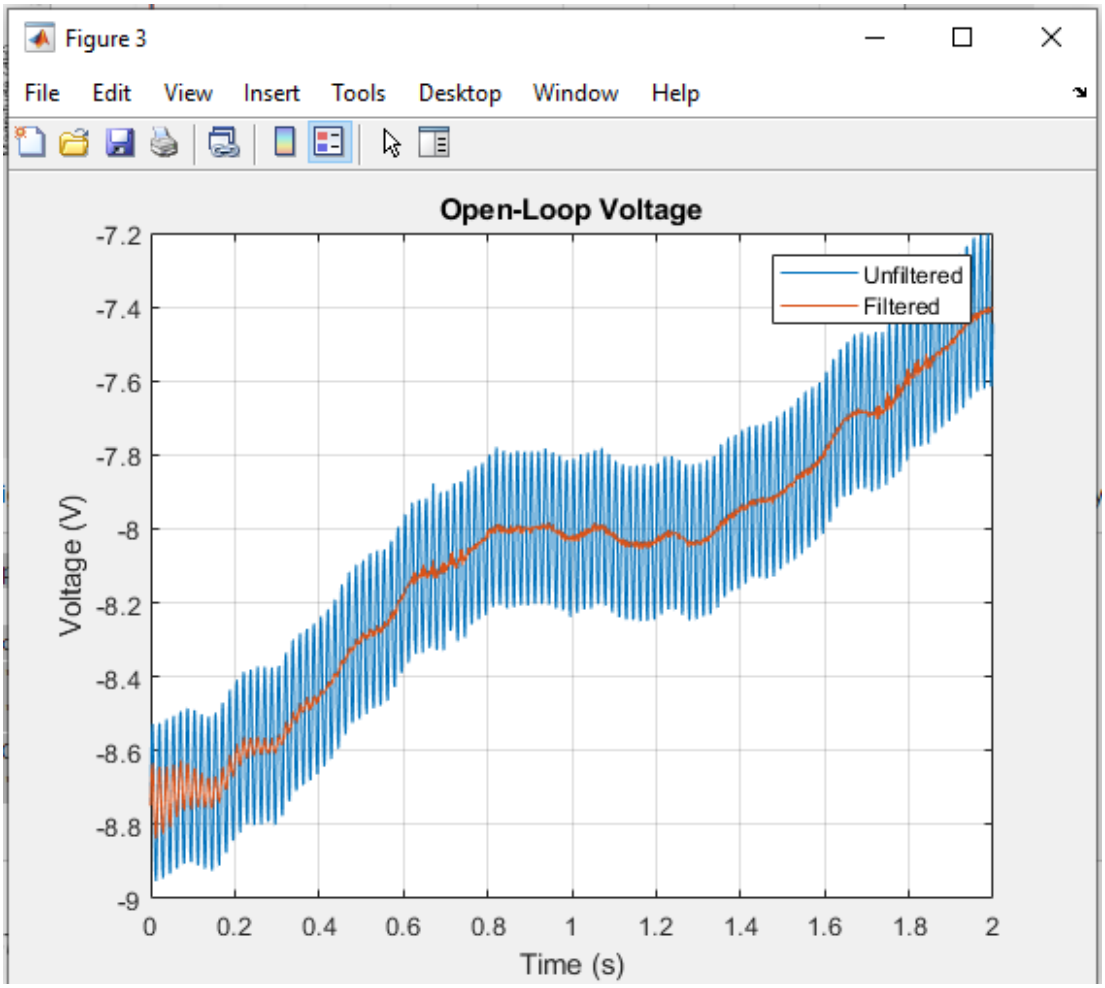
Frequency response of butterworth notch filter

STEP 3:

Filter the signal with filtfilt to compensate for filter delay. Note how the oscillations decrease significantly.

MATLAB Code:

```
buttLoop = filtfilt(d,openLoop);
plot(t,openLoop,t,buttLoop)
ylabel('Voltage (V)')
xlabel('Time (s)')
title('Open-Loop Voltage')
legend('Unfiltered','Filtered')
grid
```



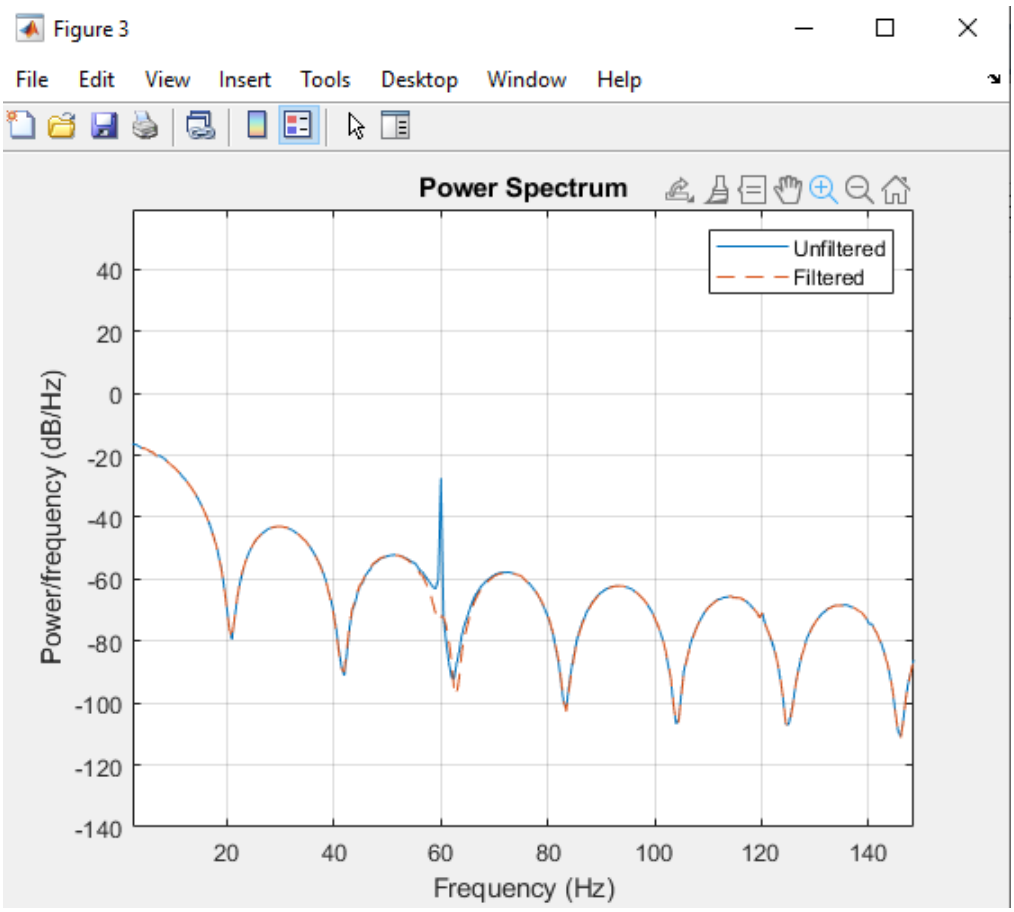
Filtered open loop voltage

STEP 4:

Use the periodogram to see that the "spike" at 60 Hz has been eliminated.

MATLAB Code:

```
[popen,fopen] = periodogram(openLoop,[],[],Fs);
[pbutt,fbutt] = periodogram(buttLoop,[],[],Fs);
plot(fopen,20*log10(abs(popen)),fbutt,20*log10(abs
(pbutt)),'--')
ylabel('Power/frequency (dB/Hz)')
xlabel('Frequency (Hz)')
title('Power Spectrum')
legend('Unfiltered','Filtered')
grid
```



60 Hz spike eliminated after filtering