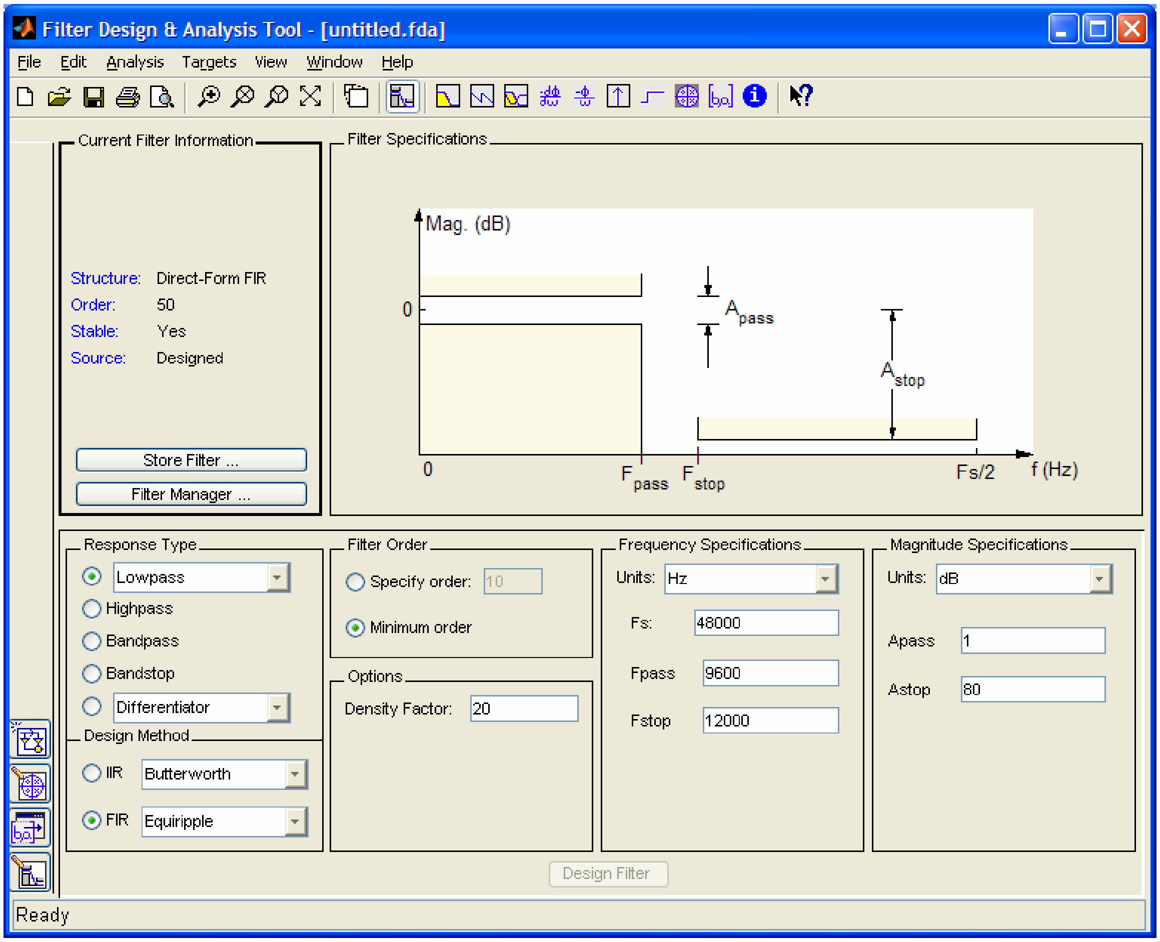
071014/ Thomas Munther
Halmstad University
School of Information Technology, Computer Science and Electrical Engineering

Exercises in Matlab/Simulink VI

In this exercise will we use another tool within Matlab called fdatool.
Simply write fdatool in the command window.

```
>> fdatool
```

This tool allows one to construct filters of different types:
Lowpass, HighPass, Bandpass and Bandstop together with different designs like Butterworth, ChebyshevI, Chebyshev II and Elliptic filters (IIR) or FIR filters using window methods among other techniques. We can of course specify very high orders or choose from 16 different windows. The frequency specification need sampling frequency and frequencies for passband and stopband. These need to be written either as frequency[Hz] or as normalized between 0 and 1. In the menu one can find other useful things like: impulse responses, step responses, pole/zero map, filter coefficients, magnitude and phase characteristics. This tool also provides possibilty to import filters or export a variable to the command window/ or as a block to Simulink or even generate a m-file.



Current Filter Information

Structure: Direct-Form FIR
Order: 50
Stable: Yes
Source: Designed

Filter Specifications

Response Type
● Lowpass
○ Highpass
○ Bandpass
○ Bandstop
○ Differentiator

Design Method
○ IIR Butterworth
● FIR Equiripple

Filter Order
○ Specify order: 10
● Minimum order

Options
Density Factor: 20

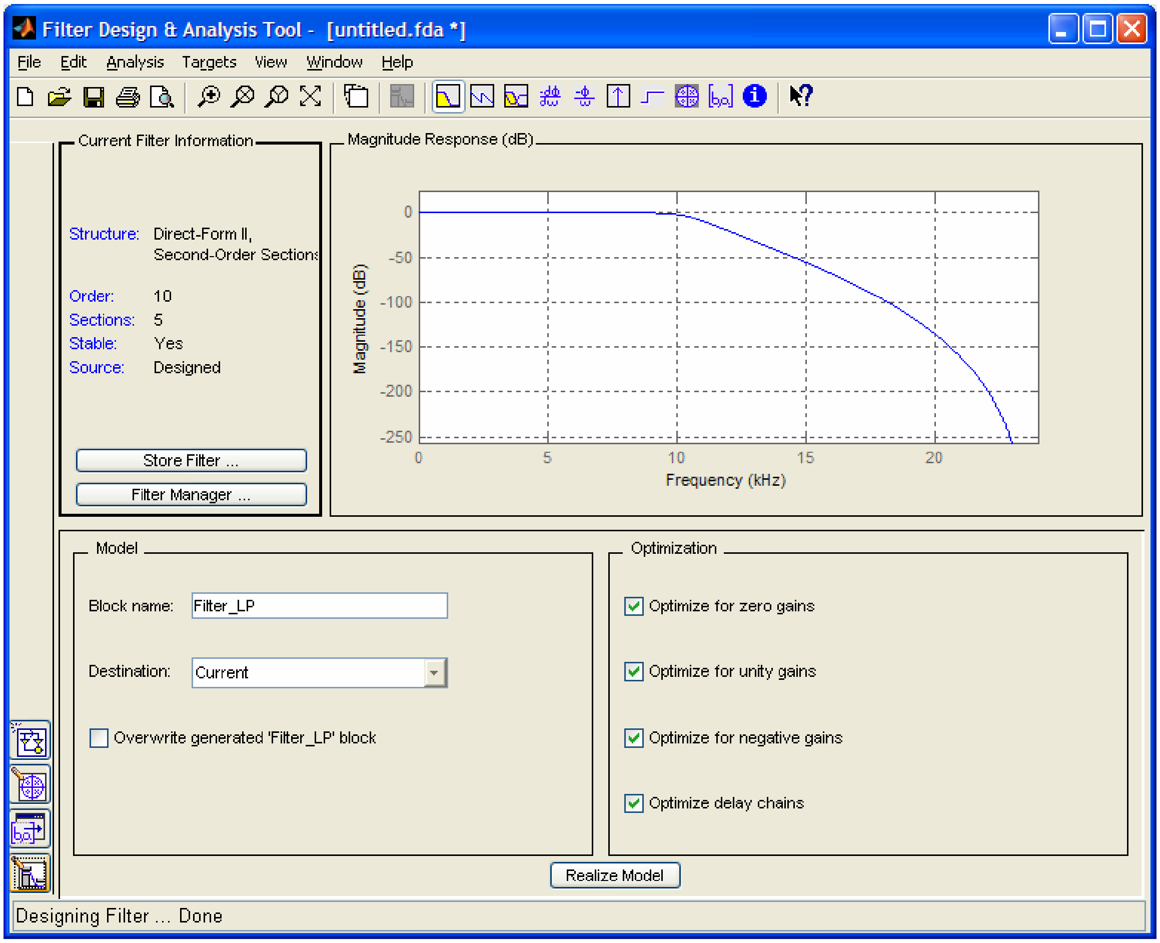
Frequency Specifications
Units: Hz
Fs: 48000
Fpass: 9600
Fstop: 12000

Magnitude Specifications
Units: dB
Apass: 1
Astop: 80

Figure 1

Design a filter of LowPass (Butterworth) since fdatool only designs discrete filters it is also a IIR filter as well. Choose sampling frequency 48000 Hz, cutoff frequency for passband is 9600 Hz and the stopband starts from 12000 Hz. Attenuation in the passband 1dB and attenuation in the stopband 20 dB. Also choose minimum order.
Now press the button ”Design filter” and the corresponding magnitude function is plotted. On the left side if the plot window are some information about the filter. For instance 10:th order and stable filter.

Now let ut export our filter to Simulink.
Enter   **File-> Export to Simulink Model**
See figure 2!



Structure: Direct-Form II,
Second-Order Sections

Order: 10
Sections: 5
Stable: Yes
Source: Designed

Block name: Filter_LP

Destination: Current

☐ Overwrite generated 'Filter_LP' block

☑ Optimize for zero gains

☑ Optimize for unity gains

☑ Optimize for negative gains

☑ Optimize delay chains

Designing Filter ... Done

Figure 2

Give the Block used in the Simulink environment a name: Filter_LP
Press the button ”Realize Model ”at the bottom..
Simulink starts and a mdl-file is opened with our filter there, figure 3.

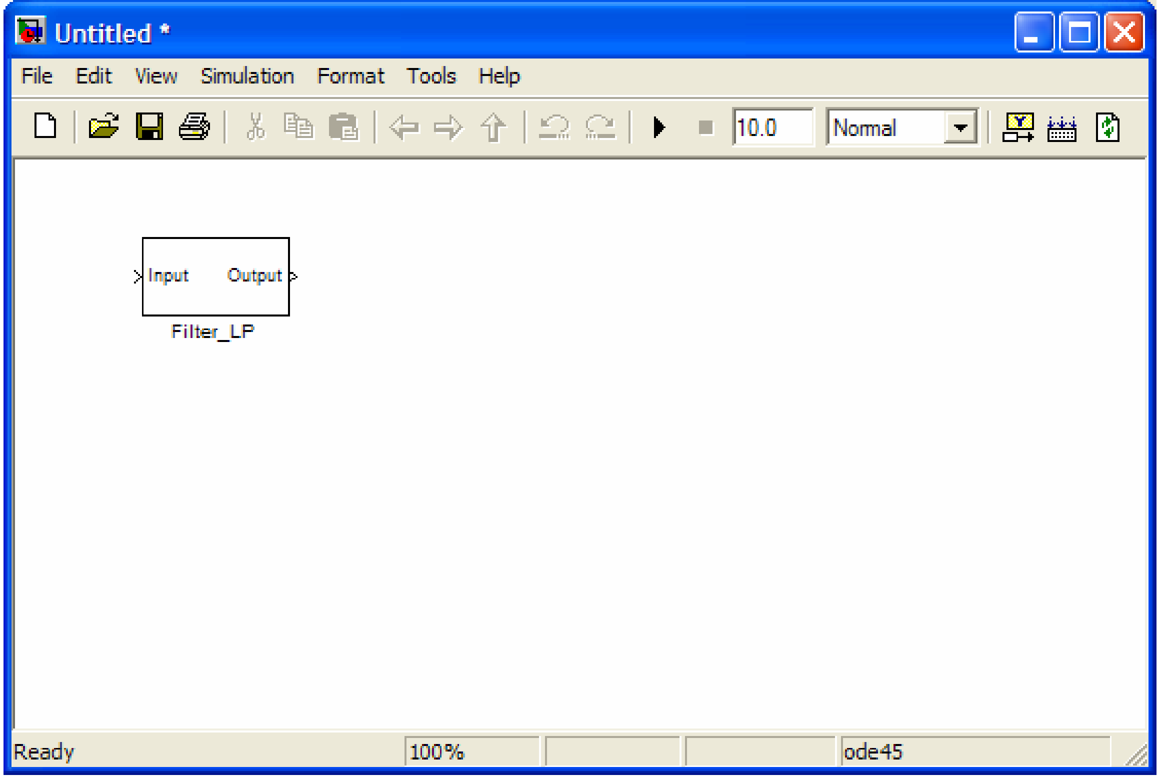


Figure 3

Let us now apply some signals to the filter to see how it performs.
If you are curious you can also double-click on the Filter icon to see how its Block diagram looks like, figure 4.

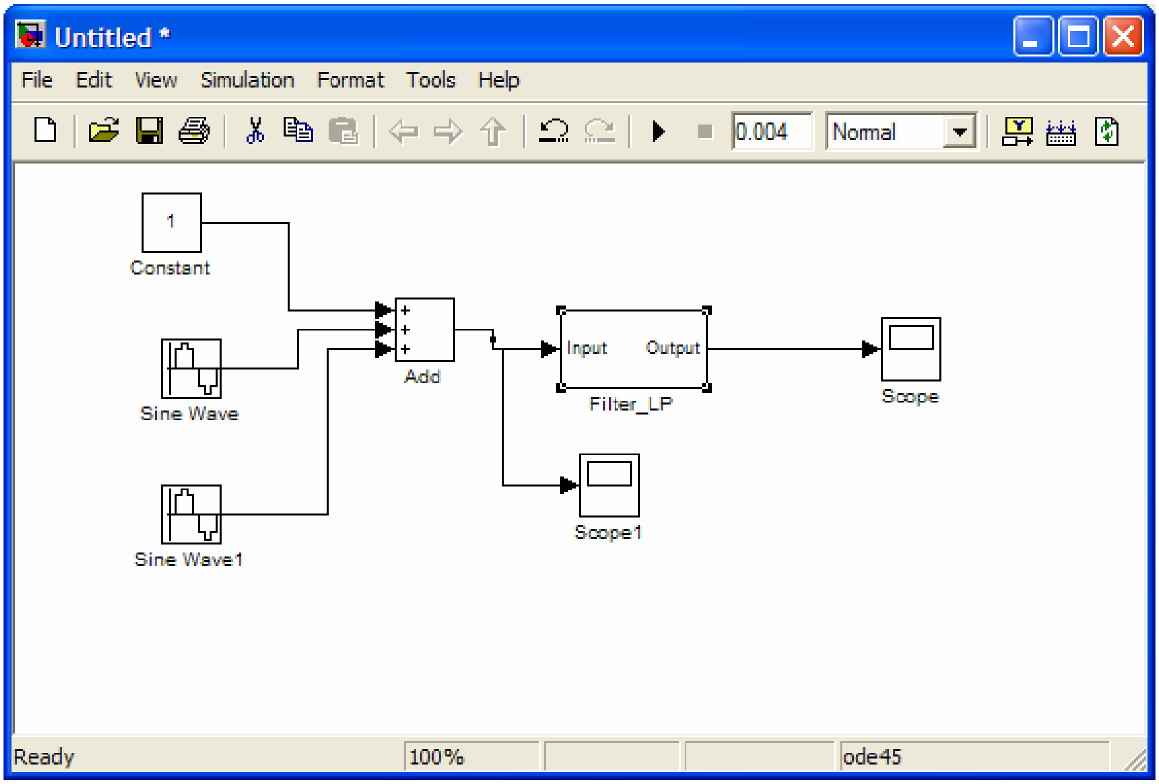


Figure 4

Sampling time is 1/48000 sec. Choose the angular frequencies to be: 1000*2π and 12000*2π in Sine wave blocks and add these together with a constant 1. Let then this signal enter the filter and see the filter action.

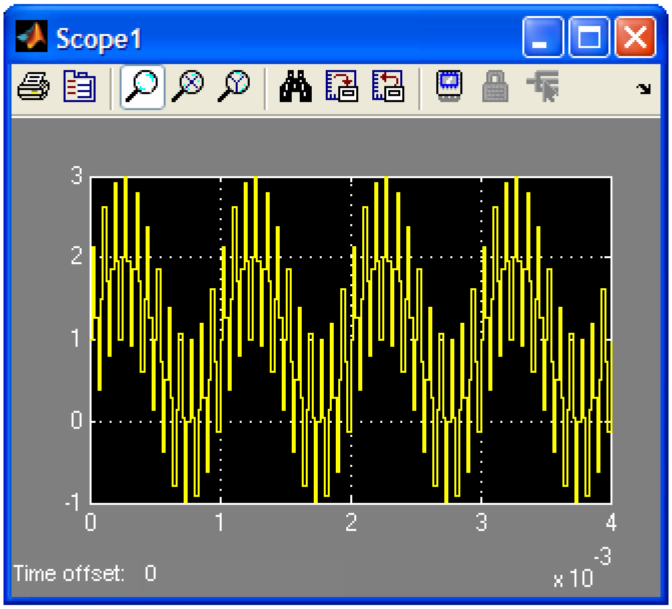


Figure 5

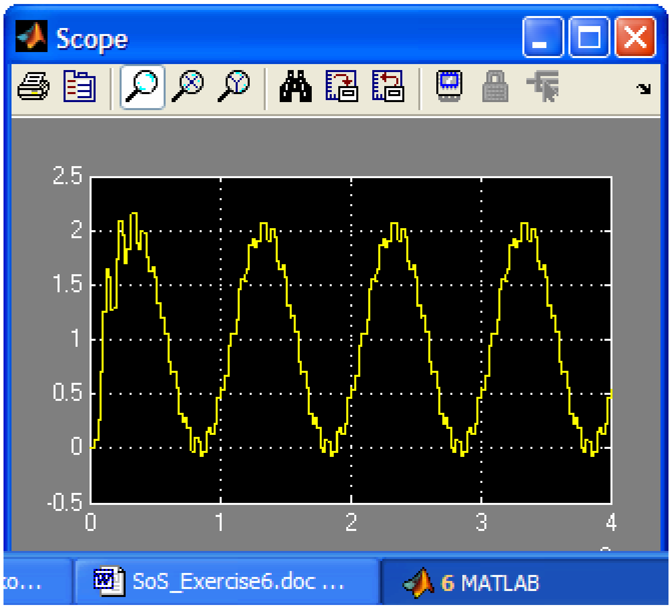


Figure 6

Filter using pole-zero map

We will try to design a filter using a technique where we pick poles and zeros that reinforces or attenuates different frequencies.
This can be done by finding the corresponding poles for the signals suppose we have the same signal as in figure 4.

```
>> t =0: 1/48000:1;     % sampling frequency 48000 Hz.
>> x = 1 .+ sin(2*pi.*1000*t) .+ sin(2*pi*12000.*t);
>> plot(t,x)               % The unfiltered input signal, figure 7
```

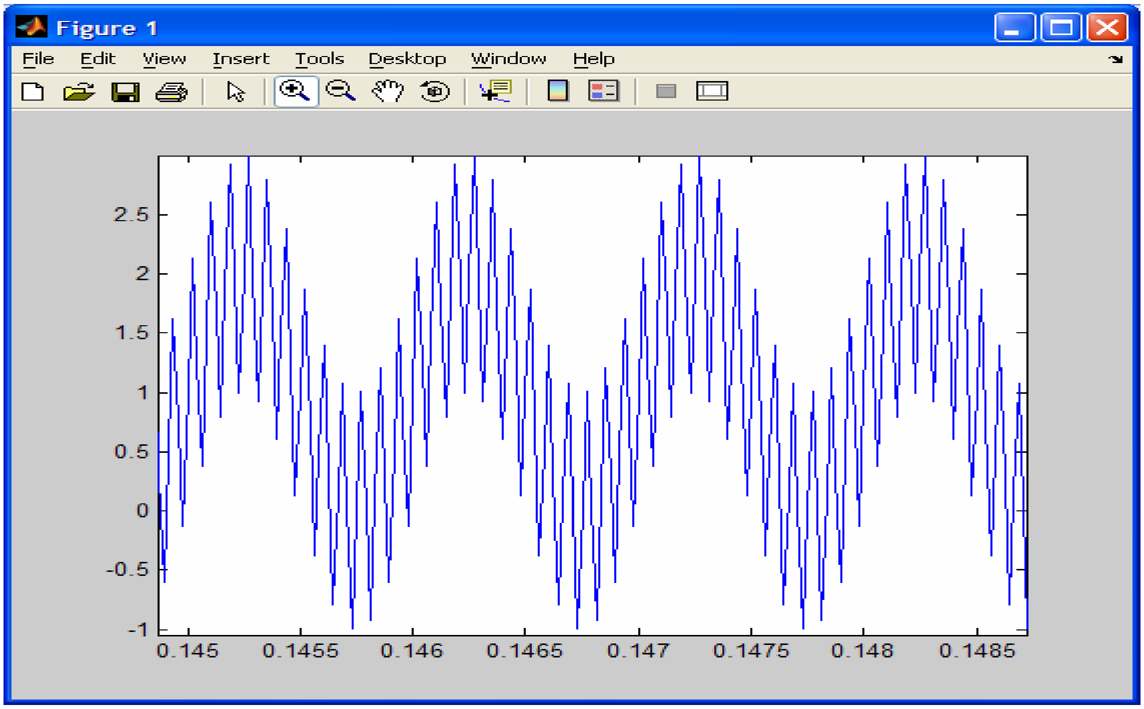


Figure 7

Eliminate the high frequency component by choosing a transfer function containing zeros at locations +i and –i .

```
        0.5 (z^2  + 1)
H[z]= -------------- , to make the filter add two poles at origin.
             z^2
```

There is also a constant 0.5 assuring that the DC-gain goes to 1.

```
>> H=zpk([ i –i],[ 0 0 ],0.5, 1);
>> y=filter([1 0 1],[1 0 0],x )  % numerator and denominator polynomials of H.
                                 % are the two first arguments and x the signal
                                 % needed to be filtered.

>> plot(t,y)     % See the filtered output, figure 8 !
```

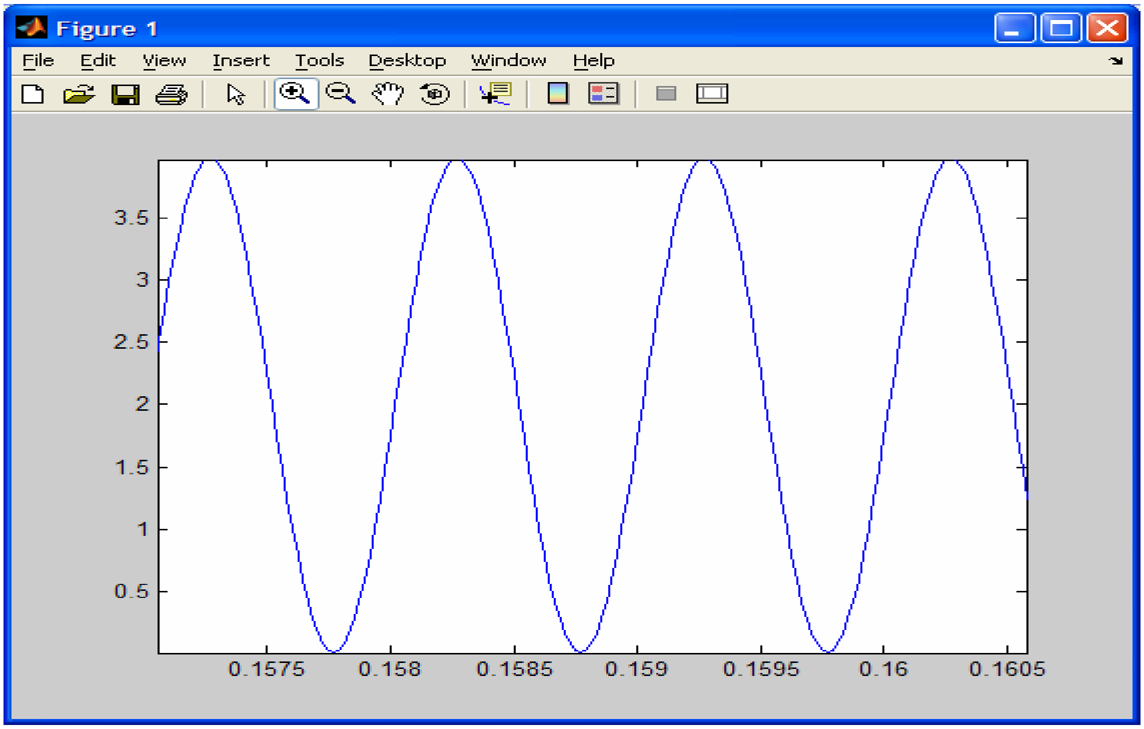


Figure 8

IIR Filters using bilinear method

To design a digital filter by the Butterworth and Chebyshev analog prototypes in matlab. First step is to decide the analog filter that meets the desired criteria and then map the analog filter to a discrete filter by a bilinear transformation. Start from a LowPass N-pole filter with a cutoff frequency 1 rad/sec.

Then transform the filter via frequency transformations like : **lp2lp, lp2bp, lp2hp** or **lp2bs** to the final analog filter.

From there we map the transfer function to a digital transfer function by the command **bilinear**.

Example: Investigate a two-pole Butterworth filter, cutoff frequency 2 rad/sec and sampling period,T=0.2.

```
[z,p,k]=buttap(2);   % creates a 2-pole Butterworth filter, wc=1rad/sec
 [num,den] = zp2tf(z,p,k)        % gives the transfer function polynomials
 wc=2              % desired cutoff frequency
[num,den] = lp2lp(num,den,wc)          % frequency transformation
T=0.2;
[numd,dend] = bilinear(num,den,1/T)

H=tf(numd,dend,1)     % corresponding digital transfer function
```

Now let us see how the Bode plots for the analog filter and digital filter look like.

```
>> bode(H), hold, bode(num,den)
```

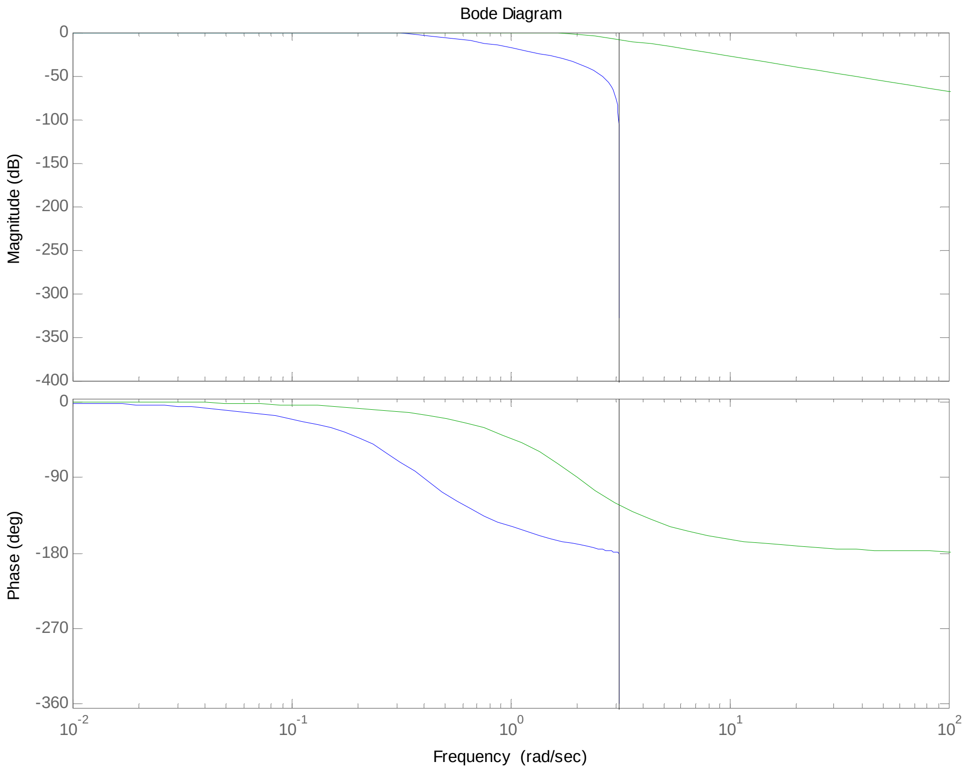


Figure 9

For low frequencies the magnitude functions seem to overlap, but at higher frequencies the gap between them becomes wider.

Exercises to be solved

The exercises should be handed in as m-files or mdl-files and sent to my emailadress. These exercises can be handed in individually or as a group, but notice that the group must not be larger than 2. Can give bonus points on the written exam.

1. Improve the filter in figure 4. see if you can make the 1000 Hz signal ”cleaner” in the output. Also combine this LP-filter with some other IIR-filter to eliminate the DC-voltage in the output. The filters should be cascaded !

2. Use a different approach to improve the filter in figure 4. Design a filter that attenuates DC-frequency and frequency 12000 Hz and possibly magnifies the frequency 1000 Hz. By using a method with poles and zeros, where we choose a certain pole close to the unit circle in the transfer function to reinforce a frequency and zeros (close to the unit circle)to attenuate frequencies.
   Hint: use zpk and at least filter order 3.

3. Use the fdatool to filter x(t) in problem 3, so that 12000 Hz sine wave is eliminated and the bias voltage disappears. Use a FIR filter with method windows. Try both Rectangular and Gaussian windows use the same cutoff frequencies and same order.
   Combine two cascaded FIR-filters to solve the assignment.

4. Construct a causal filter that eliminates 50 Hz sine wave signals while the alternating sequence $(-1)^n u[n]$ is reinforced with a factor 10. The frequency of the sampling is 200 Hz.
   Show the corresponding Bode plot of the filter and give the corresponding transfer function.
   Hint: third order filter is enough !