Université d'Ottawa | University of Ottawa

ELG4177 - DIGITAL SIGNAL PROCESSING
Lab6

By:Hitham Jleed

http://www.site.uottawa.ca/~hjlee103/

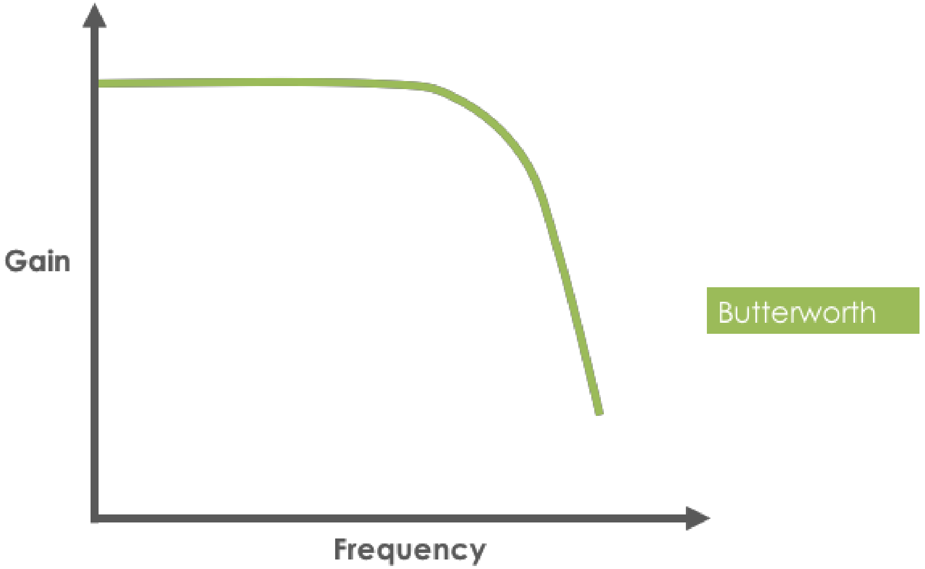
uOttawa

Assignment 06

IIR FILTER DESIGN

Butterworth Filter

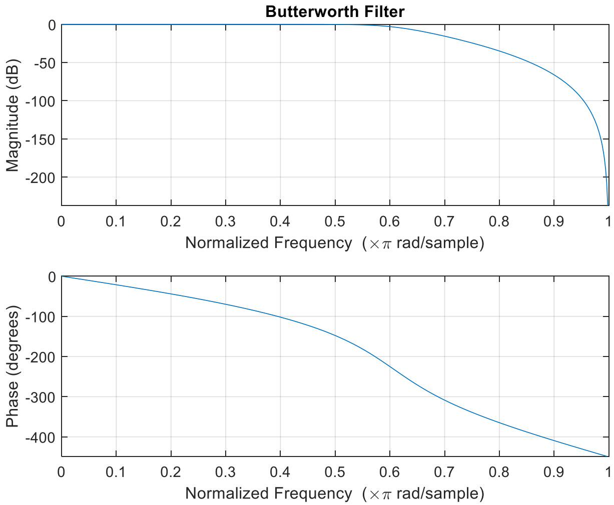
•**Named after:** British Physicist Stephen Butterworth
•**Butterworth filter response:**
the passband response offers the steepest roll-off without inducing a passband ripple. In addition to the flat passband response

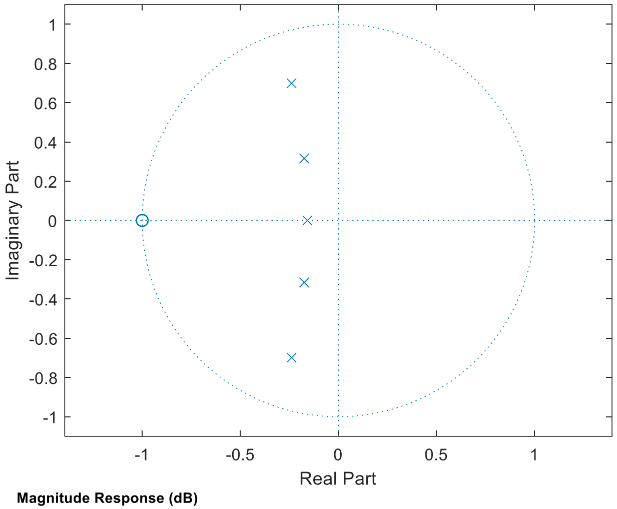


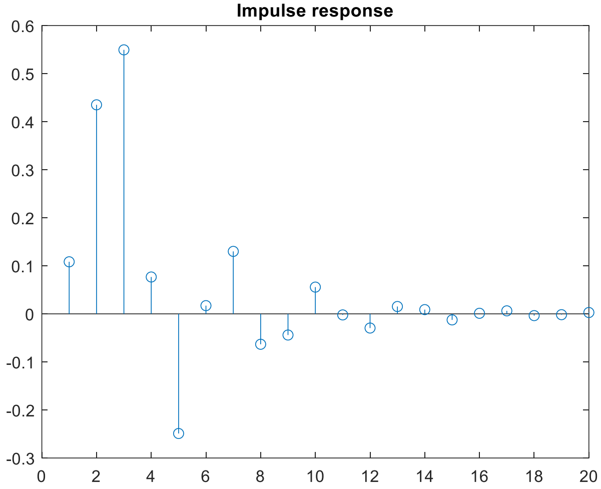
a) Design a fifth-order digital Butterworth low-pass filter with a sampling frequency of 200 Hz and a passband edge of 60 Hz. Use the *butter* function. Plot the phase and magnitude frequency response using *freqz*. Find the zeros and the poles, and plot them on the same **z-plane** { option : old way: (using *plot(roots(a), 'x')* and *plot(roots(b), 'o')*}. Plot the significant (energetic) part of the impulse response of the filter, using *impz*. .

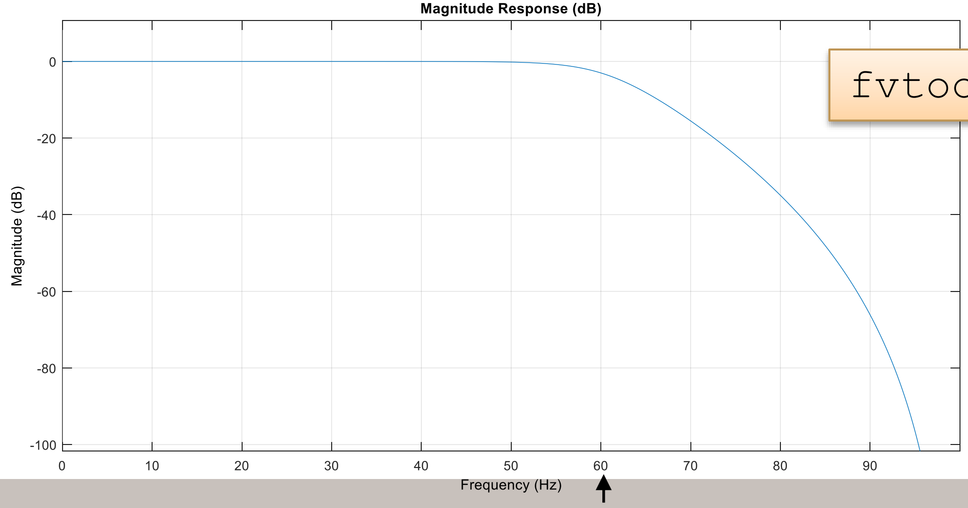
```
n = 5; fp = 60; fs = 200;
```

```
[b, a] = butter(n,fp/(fs/2));
```


Butterworth Filter




Impulse response

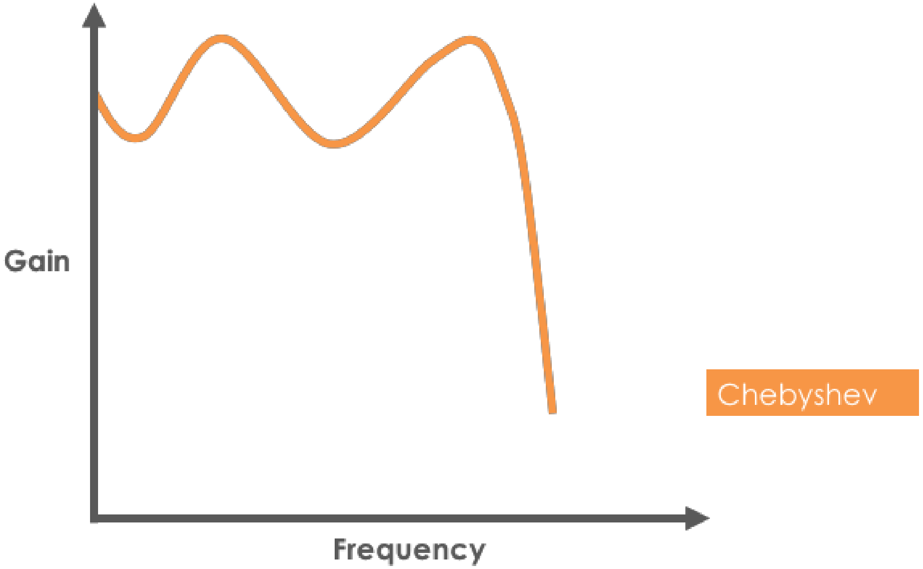

Magnitude Response (dB)

```
fvtool(b,a,'Fs',fs)
```

Chebyshev Filter

- **Named after**: Russian mathematician Pafnuty Chebyshev
- **Chebyshev filter response**:

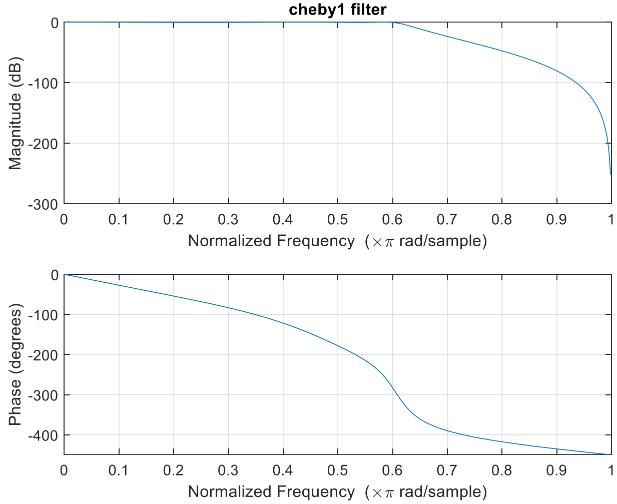
The Chebyshev filter is known for it's ripple response. This ripple response can be designed to be present in the passband (Chebyshev Type 1) or in the stopband (Chebyshev Type 2).

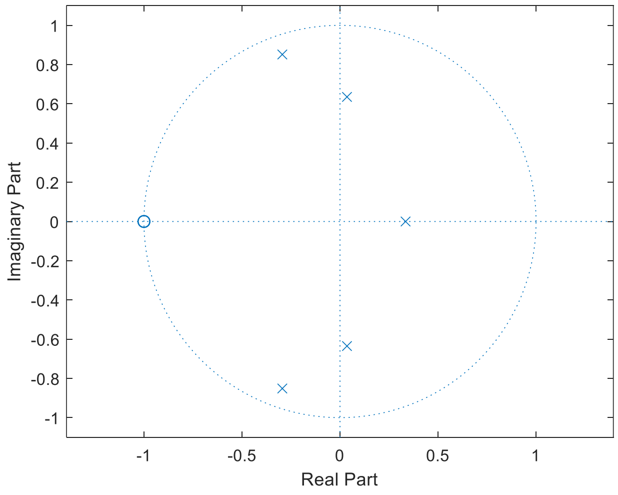


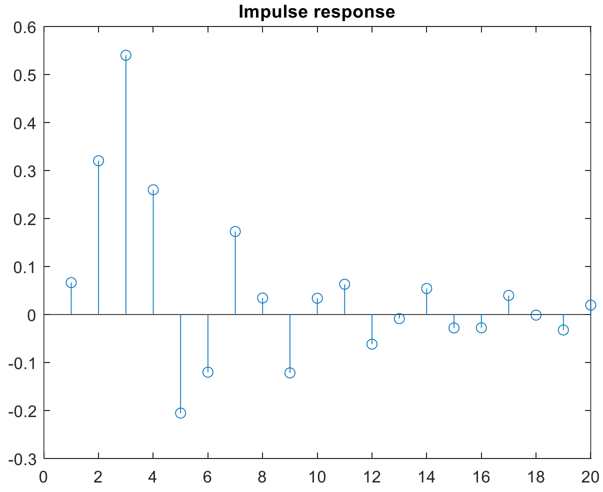
b) Repeat a) but use a Chebyshev type I filter with a passband ripple of 0.5 dB. Use the *cheby1* function. Compare with the plots found in a).

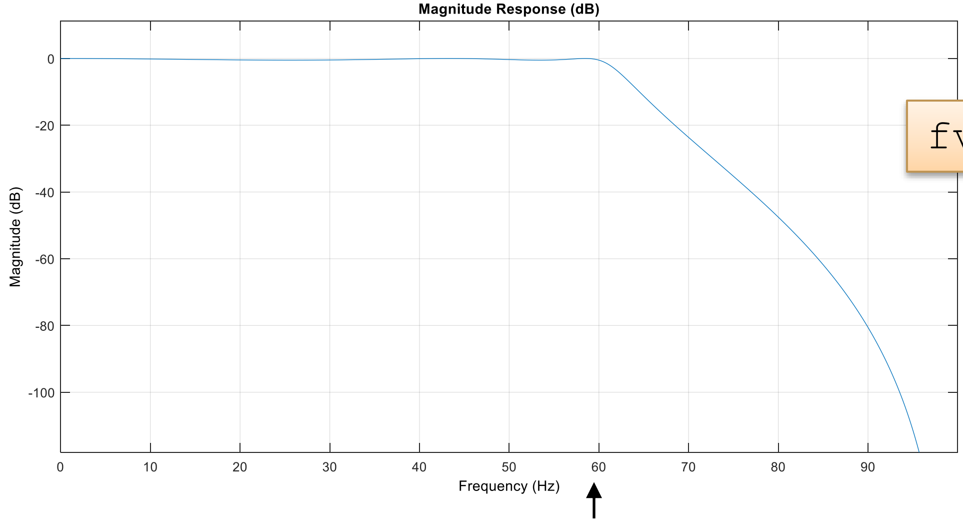
```
n = 5; fp = 60; fs = 200; Rp = 0.5;
```

```
[b,a]=cheby1(n,Rp,fp/(fs/2));
```







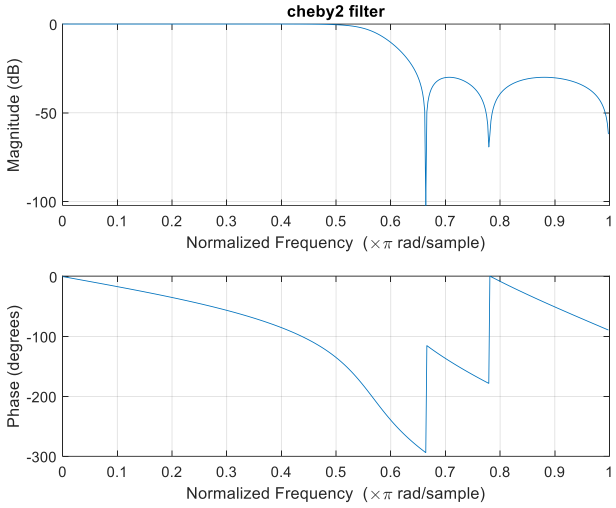


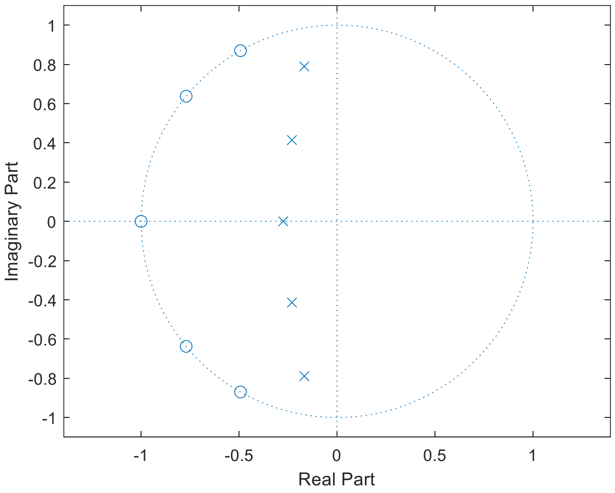
```
fvtool(b,a,'Fs',fs)
```

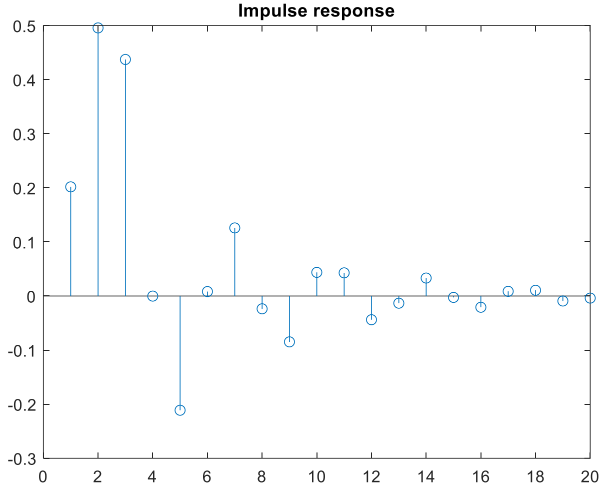
c) Repeat a) but use a Chebyshev type 2 filter with stopband edge of 65 Hz and a stopband ripple 30 dB less than the passband. Use the *cheby2* function. Compare with the plots found in a).

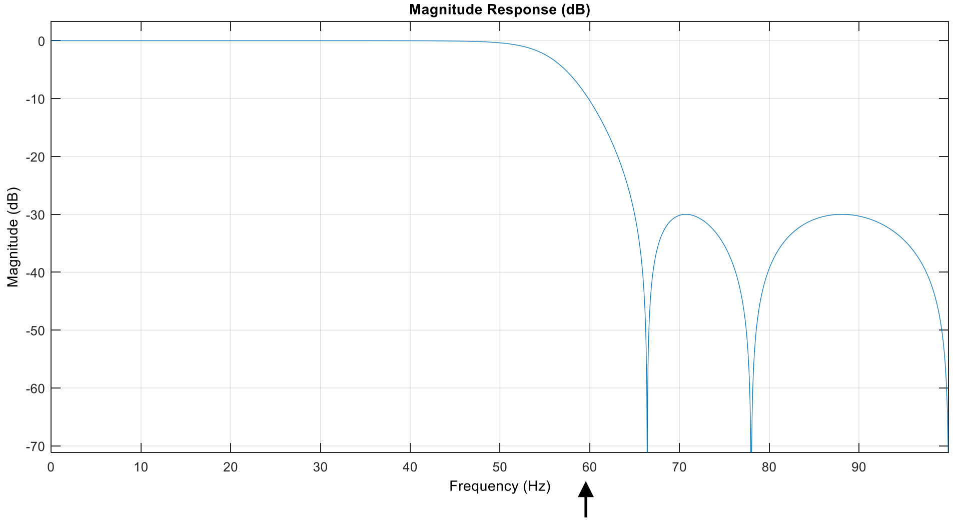
```
n = 5; fp = 65; fs = 200; Rs = 30;
```

```
[b,a]=cheby2(n,Rs,fp/(fs/2));
```







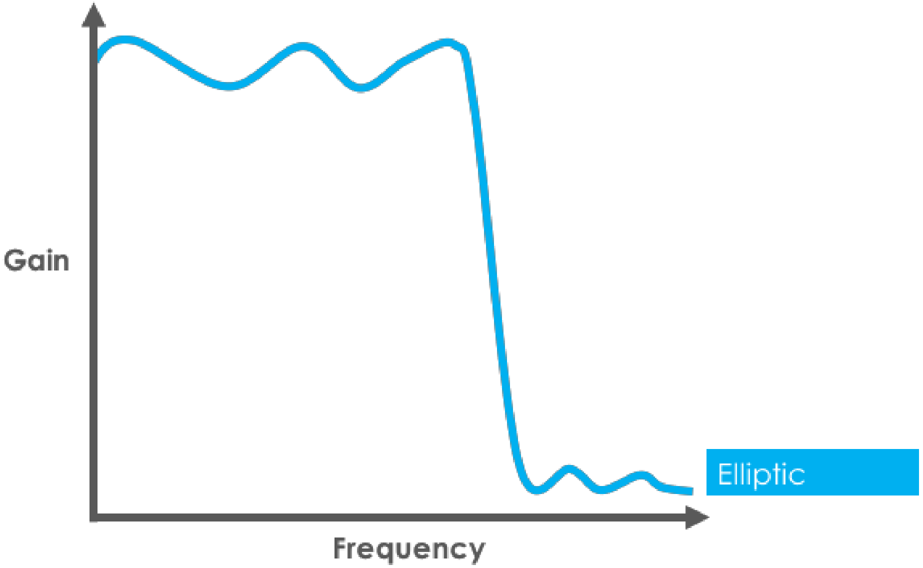


```
fvtool(b,a,'Fs',fs)
```

Elliptic (Cauer) Filter

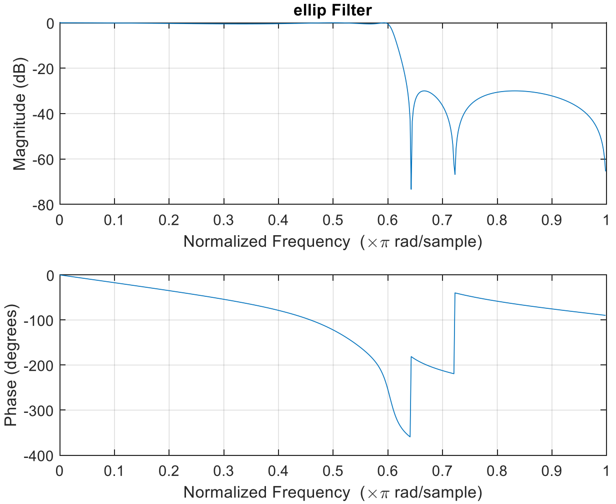
- **Named after:** German Mathematician Wilhelm Cauer
- **Elliptic filter response:**

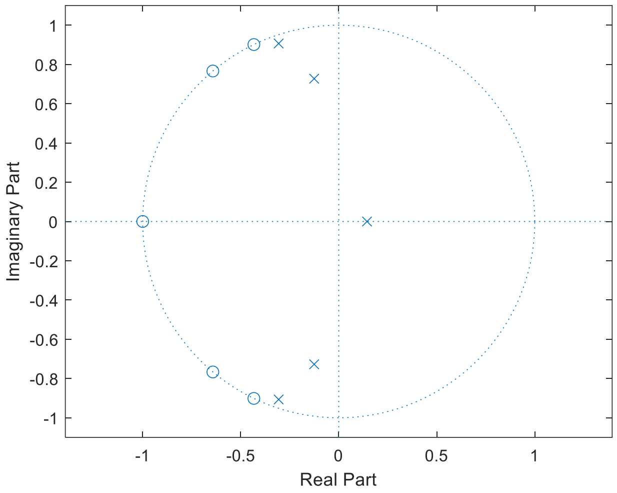
The elliptic filters is characterized by ripple that exists in both the passband, as well as the stopband. The passband ripple of the elliptic filter is similar to the Chebyshev filter, however the selectivity is greatly improved.

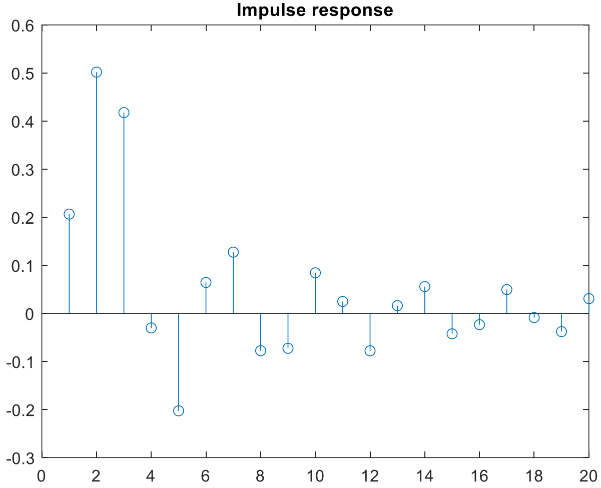


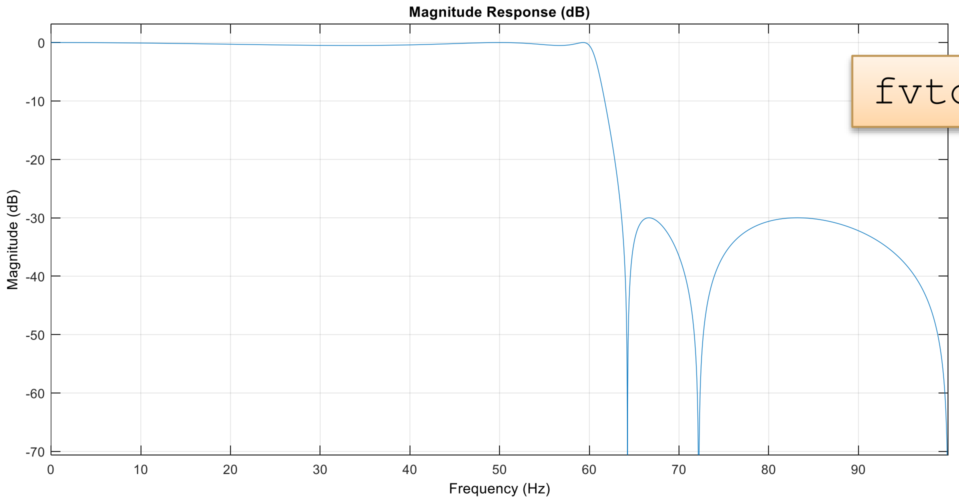
d) Repeat a) but use an elliptic (Cauer) filter with a passband ripple of 0.5 dB and a stopband ripple 30 dB less than the passband. Use the *ellip* function. Compare with the plots found in a).

```
[b,a] = ellip(n,Rp,Rs,fc/(fs/2))
```









```
fvtool(b,a,'Fs',fs)
```

e) Matlab provides functions to estimate the order required for different types of IIR filters in order to meet some specifications. For the following specifications :

- sampling rate 200 Hz
- passband ripple 0.1 dB
- passband edge 28 Hz
- stopband edge 32 Hz
- stopband ripple below 30 dB

```
fs = 200;
Rp = 0.1;
fp = 28;      Wp = 2*fp/fs;
fc = 32;       Ws = 2*fc/fs;
Rs = 30;
```

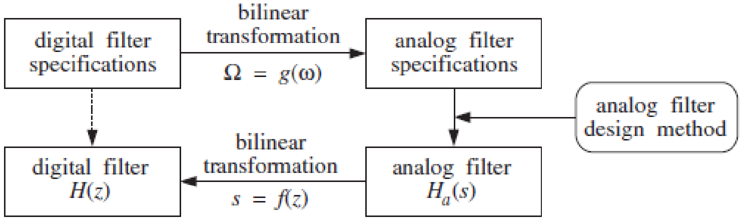
```
>>help buttord
>>help cheb1ord
>>help cheb2ord
>>help ellipord
```

f) For the specifications in e), design a FIR Chebyshev/Parks-McClellan/Remez low-pass filter, using *remez* and *remezord* functions. Verify the frequency response of the designed filter. Compare the required order for this FIR filter with the orders of the IIR filters found in e).Discuss.

```
% The desired amplitude of the output filter for each band.
dev = [(10^(Rp/20)-1)/(10^(Rp/20)+1)  10^(-Rs/20)];
f = [fp fc];    a = [1 0];
% specifications f, a, dev and fs
[n,fo,ao,w] = firpmord(f, a, dev, fs);

% Use firpm with the resulting order n, frequency vector fo, amplitude
% response vector ao, and weights w to design the filter b which
% approximately meets the specifications given by firpmord input
% parameters f, a, and dev.
b = firpm(n,fo,ao,w);
freqz(b,[1]);
```

Bilinear transform



$$s = \frac{2}{T_d}\left(\frac{1-e^{-j\omega}}{1+e^{-j\omega}}\right), \qquad (7.24)$$

$$\Omega = \frac{2}{T_d}\tan(\omega/2), \qquad (7.26)$$

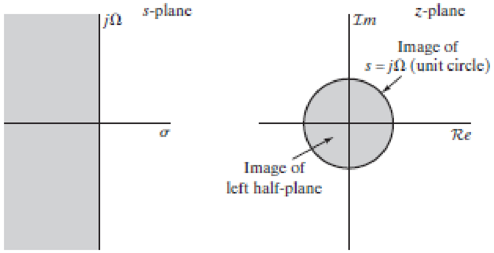


Figure 7.6 Mapping of the *s*-plane onto the *z*-plane using the bilinear transformation.

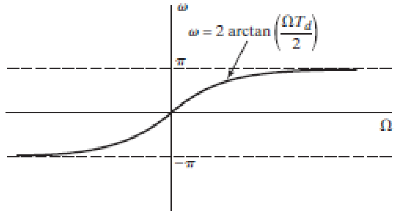


Figure 7.7 Mapping of the continuous-time frequency axis onto the discrete-time frequency axis by bilinear transformation.

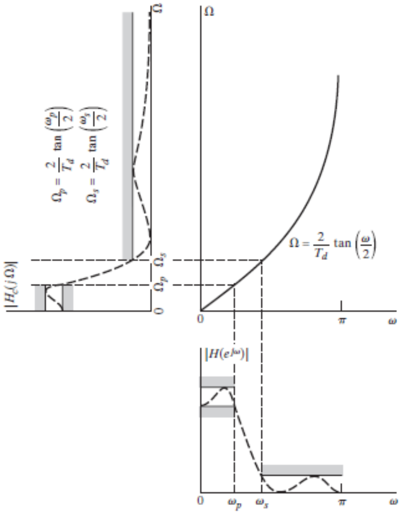
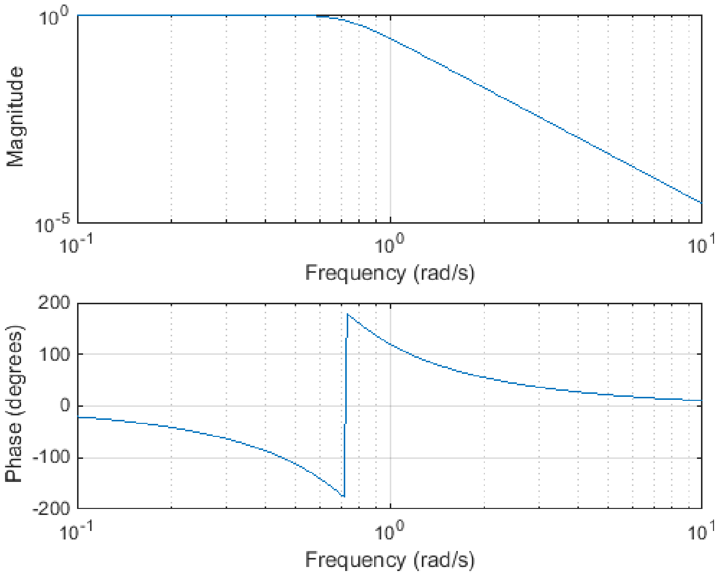
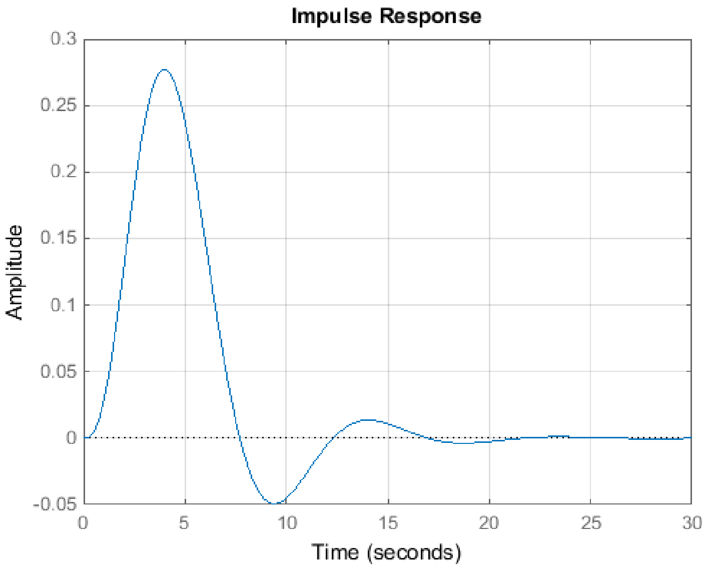


Figure 7.8 Frequency warping inherent in the bilinear transformation of a continuous-time lowpass filter into a discrete-time lowpass filter. To achieve the desired discrete-time cutoff frequencies, the continuous-time cutoff frequencies must be prewarped as indicated.

Analog Butterworth

h) Design the low-pass analog 4th order Butterworth filter using the passband edge found in g) and the function butter(N,Wn,'s'). Verify the frequency response and stability (via the impulse response), using the freqs and impulse(b,a,Tf) functions for analog systems.





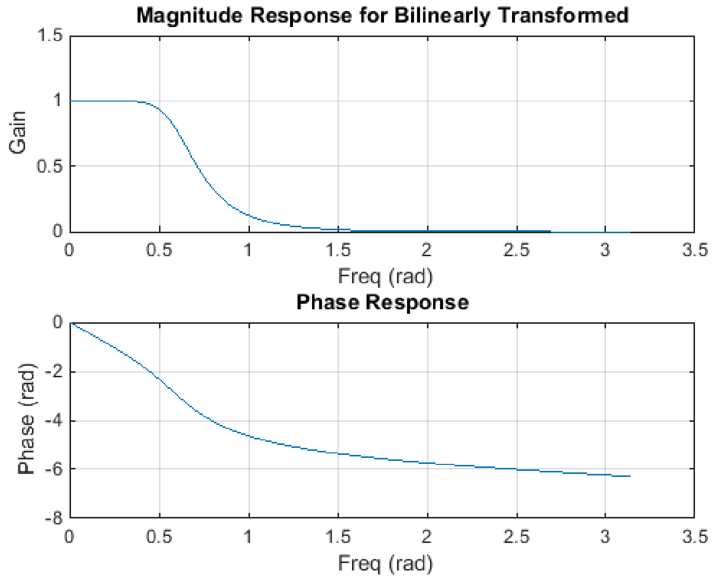
Bilinear: Analog -> Digital

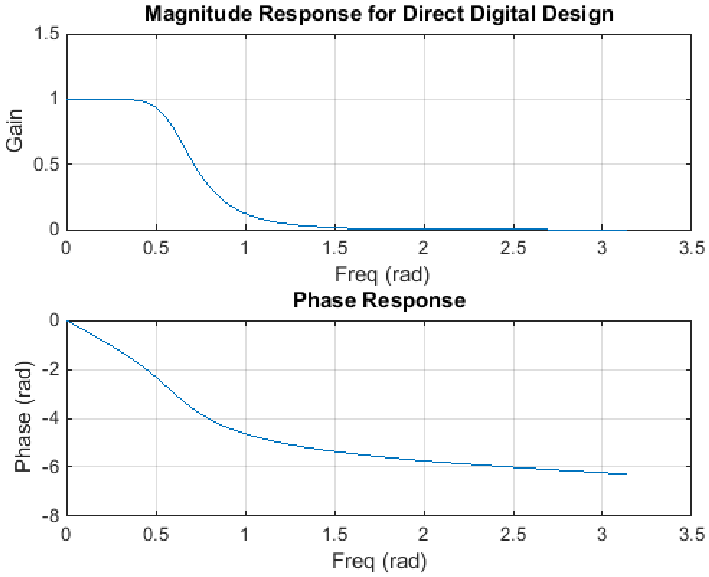
i) Using the *[b2,a2] = bilinear(b,a,Fs)* command, transform the continuous time filter to a discrete time filter. Check the frequency response of the digital filter. Does it meet the specifications ? Design an equivalent filter directly in the discrete time domain using *butter*, and compare the results.

```
n = 4; fs = 40000; fp = 8000;wp = 2*pi*fp/fs; Om_p = 2*fs*tan(wp/2); % or 2*tan(wp/2); but use below [b2, a2] = bilinear(b, a, 1);
```

```
[b,a] = butter(n, Om_p, 's');
[b1, a1] = bilinear(b, a, fs);
[H1, w1] = freqz(b1, a1);
```

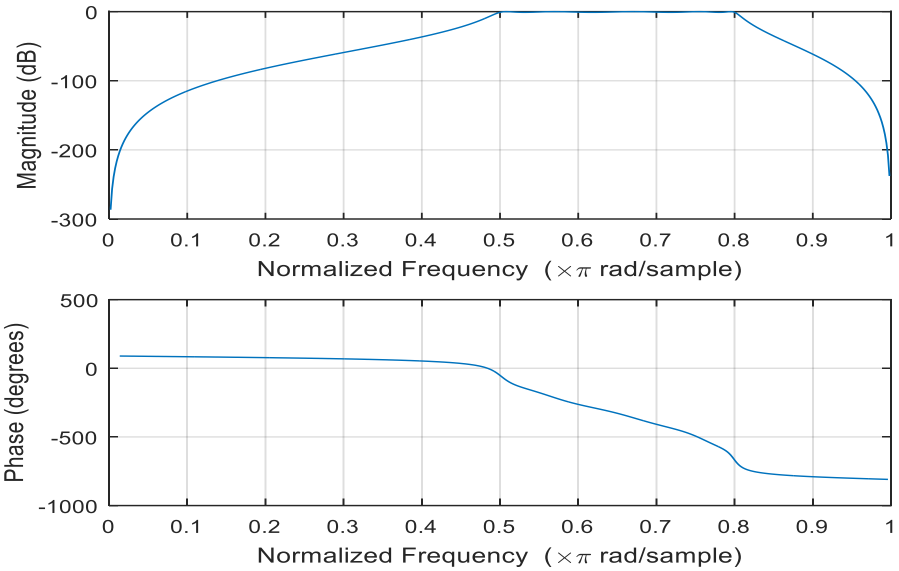
```
[b2,a2] = butter(n, wp);
[H2, w2] = freqz(b2, a2);
```

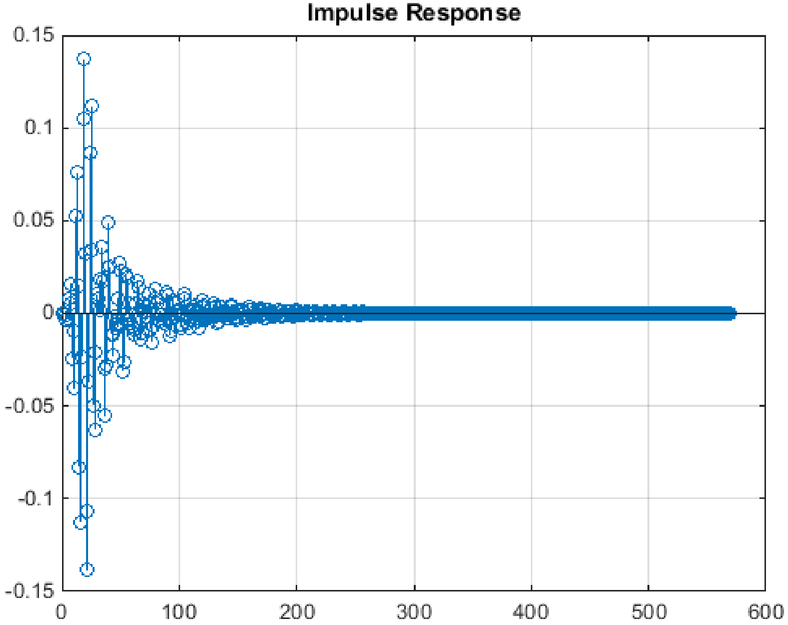




Design of filters by frequency transformation

j) It is possible to directly design a digital IIR highpass, bandpass or band reject filter using the *butter*, *cheby1*, *cheby2* and *ellip* functions. Design a 5th order Chebyshev type I bandpass filter with a sampling frequency of 20 Hz, a lower passband edge of 5 Hz, an upper passband edge of 8 Hz, and a passband ripple of 1 dB. Plot the frequency response and the impulse response.





```
% Normalised frequency needed
wp = 2*fp/fs;  wc = 2*fc/fs;
[b, a] = cheby1(n, Rp, [wp wc]);
```

k) The different steps performed in the design of the digital bandpass filter in j) will now be illustrated. First, design an **analog** 5th order Chebyshev type I **lowpass** filter with a passband ripple of 1 dB and a unity passband edge frequency. Plot the frequency response of the filter. Then convert it to an analog bandpass filter using the $s \rightarrow \frac{s^2+\Omega_1\Omega_2}{s(\Omega_2-\Omega_1)} = \frac{s^2+\Omega_0^2}{s \times BW}$ transformation (*lp2bp* function). Ω_1, Ω_2 are the passband edge frequencies of the analog bandpass filter, and they are computed from the edge frequencies of the desired bandpass digital filter using the bilinear transformation. Plot the frequency response of the filter. After the analog bandpass filter is designed, convert it to a digital filter using the *bilinear* function. Plot the frequency response of the digital filter, and compare it with the response found in j).

```
% peak ripple in the passband. It
returns the zeros and poles in length
% n column vectors z and p and the
gain in the scalar k.
[b, a] = cheby1(n, Rp, wp, 's');

fp = 5; fc = 8; fs = 20;

wp = 2*pi*fp/fs;
wc = 2*pi*fc/fs;

omegap = 2*fs*tan(wp/2);
omegac = 2*fs*tan(wc/2);

Wo = sqrt(omegap*omegac);
Bw = omegac - omegap;
```

```
% [bt,at] = lp2bp(b,a,Wo,Bw) transforms an
analog lowpass filter
% prototype given by polynomial coefficients
into a bandpass filter with
% center frequency Wo and bandwidth Bw. Row
vectors b and a specify the
% coefficients of the numerator and denominator
of the prototype in
% descending powers of s.
[bt at] = lp2bp(b,a,Wo,Bw);

[b2, a2] = bilinear(bt, at, fs);
freqz(b2, a2)
```

Now Follow these steps to complete all questions