HUMBER POLYTECHNIC

Faculty of Applied Science and Technology

Laboratory 7 Filter Design

Student Name: Michael McCorkell Student Number: N01500049 Date: April 1st 2025

Note 1: This is an individual lab; please complete this on your own laptop.
Note 2: Lab report needs to be converted into .pdf format and submitted to SLATE -> Assignments folder before the deadline.

1. Learning outcome:

1.1 Familiarize with the design process of simple analog and digital filters mathematically.
1.2 Differentiate and assess the performance of different prototype filters in the frequency domain.
1.3 Apply filter design tools to design analog and digital filters.

2. Background

2.1 Filter overview

In the previous labs, we have used several basic filters to solve some signal processing problems. They are implemented either using mathematically formulated transfer functions or in Simulink. In this lab, we are going to explore how to design analog and digital filters based on the design specifications.

Filters are systems that process signals in a frequency-dependent manner. To implement the filter, there are either analog or digital approaches.

- Analog filters: systems that process and transform analog signals. Since most of the physical quantities that an electrical system encounters are analog, such as voltage and current. In such cases, analog filters work with the analog input signals authentically.
- Digital filters: systems that process and transform digital signals. If the actual signals are analog, they need to be sampled and converted into digital signals by analog-to-digital converters (ADC) before the filtering process starts. In the process of designing digital filters, the sampling rate needs to be specified, assuming the inputs are analog.

Both types have their advantages and disadvantages. The main areas of comparison include speed, complexity, flexibility, and range. Regardless of the input and output signals being analog or digital, filters can also be classified based on their behavior at different frequencies:

- Low-pass filters (LPF): used to pass a band of preferred low frequencies and reject undesirable high frequencies.
- High-pass filters (HPF): used to pass a band of preferred high frequencies and reject undesirable low frequencies.
- Band-pass filters (BPF): used to pass a band of frequencies and reject low- and high-frequency bands.

An ideal filter must satisfy the following conditions: its amplitude response is unity (or at a fixed gain) for the frequencies of interest (defined as the passband of the filter) and zero everywhere else (defined as the stopband of the filter); its phase shift in the frequency bands of interest is either zero or a fixed constant.

The frequency at which the response changes from passband to stopband is defined as the cutoff frequency fc. Figure 1 shows the magnitude spectrum of three types of ideal filters.

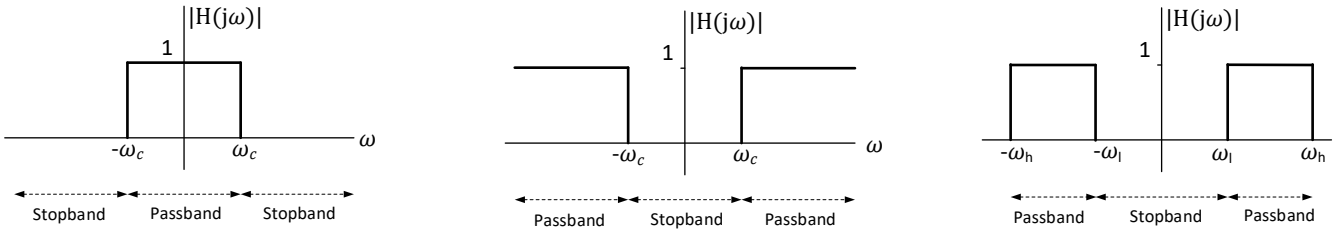


Figure 1. Three types of ideal filters (left: low-pass; middle: high-pass; left: band-pass)

Ideal filters are great for understanding the concept and some theoretical analysis. However, such filters are often impractical or impossible to implement in practice. The sharp transition from passband to stopband is not instantaneous. Also, the physical components that the filters are built from introduce imperfections and fluctuate in performance. The resultant non-ideal filters are often referred to as practical filters. In this lab, we are going to design analog and digital filter design with the help of MATLAB filter design tools. With appropriate tools, you can design filters that meet the design specifications of the problem working within the hardware constraints. You can further analyze filters performance and other properties.

2.2 Compromises in filter design

When designing digital filters, please keep in mind to design the practical filters in which compromises have to be made in passband ripple, stopband ripple or transition width, and the filter order in order to keep them practical.

Figure 2 below demonstrate such compromises and how they differ from an ideal one. Here cutoff frequency fc is defined as the frequency at which the filter's magnitude response drops to **-3 dB** (≈ **70.7%** of the passband amplitude). Passband ripple (Rp) and stopband attenuation (Sp) are important filter parameters that you need to use when designing certain types of prototype filters.

In practice, designing any filter means trading off between the following key design specifications:

- Filter order: for analog filters, that means the number of components needed for filter construction and the complexity of the filter circuit; for digital filters, this means the number of coefficients (impacting memory usage) and the complexity of the computations.
- Transition bandwidth: the steepness of the roll-off.
- Passband ripple / stopband attenuation ratio: ideally to have minimal pass band ripple in both and preferably maximal stopband attenuation.

This trade-off triangle works for any filter design project that you are going to get involved in future. And understanding the inevitable compromise will make your design more practical and efficient. For example, you can design a highly complex high order filter, and see the changes if you relax one of the three specifications.

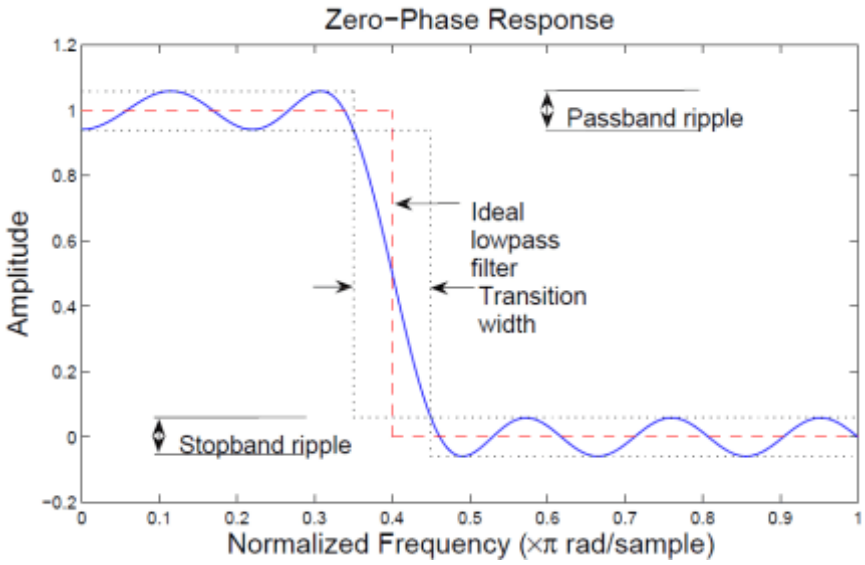


Figure 2. Practical Considerations in Digital Filter Design (Image from here).

3. Procedures

3.1 First-order single pole RC filter.

Let's start with an investigation on the frequency response of a RC network (see Figure 3). The goal is to determine the frequency response of this analog filter. Consider $x(t)$ to be the input voltage and $y(t)$ to be the output voltage. With Kirchhoff's law:

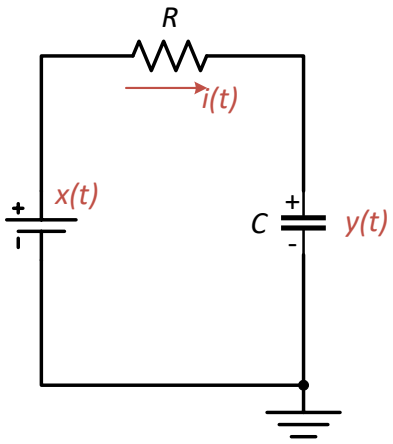


Figure 3. RC circuit

$$x(t) - y(t) = Ri(t) \qquad \text{Eq. 1}$$

$$i(t) = \frac{dQ_c}{dt} = \frac{dCy(t)}{dt} \qquad \text{Eq. 2}$$

Substitute eq. 2 into eq. 1, we have:

$$x(t) = y(t) + RC\frac{dy(t)}{dt} \qquad \text{Eq. 3}$$

Perform the Laplace transform on this equation, in the s-domain;

$$X(s) = Y(s) + sRCY(s)$$

Transfer function of this RC filter is:

$$H(s) = \frac{Y(s)}{X(s)} = \frac{1}{1 + sRC}$$

This transfer function represents a first-order low pass RC filter. Here first order refers to the number of poles in this transfer function.

Task 1 (5%). Based on this transfer function, decide the frequency response of this system and how it will serve as a low-pass filter. Show your work below. Hint: compute the frequency response of this system to help with your analysis.

$H(jw) = \frac{1}{1+jwRC}$

	Magnitude	Phase	Cutoff (-3db)
	$\lvert H(jw)\rvert = \left\lvert \frac{1}{\sqrt{1+(wRC)^2}} \right\rvert$	$/\_H(jw) = \tan^{-1}(-wRC)$	$\frac{1}{\sqrt{1+(w_cRC)^2}} = \frac{1}{\sqrt{2}}$ $1+(w_cRC)^2 = 2$ $w_c = \frac{1}{RC}$
Low Frequencies	$\lvert H(jw)\rvert \approx 1$	$/\_H(jw) \approx 0$	
High Frequencies	$\lvert H(jw)\rvert \approx \frac{1}{wRC}$	$/\_H(jw) \approx -90°$	

$f_c = \frac{1}{2\pi RC} \rightarrow$ is a low pass because it passes low frequencies and attenuates high frequencies

Task 2 (5%). How to modify this system so that it can be used as a high-pass filter? Write the corresponding transfer function of the high-pass filter using $H(s)$.

$H(jw) = \frac{1}{1+jwRC} \rightarrow H(s) = \frac{sRC}{1+sRC}$

Task 3 (10%). Based on your design of low-pass and high-pass filters, explore how you can combine the low-pass and a high-pass filter to design a band-pass filter. Write the transfer function of this filter using $H(s)$.

$$H(s) = \frac{sRC}{1+sRC} * \frac{1}{1+jwRC} \rightarrow \frac{sRC}{(1+sRC)^2}$$

3.2 Higher order analog filter design using transfer function

In this section, we are going to explore some common design equations. Below are three second-order transfer functions commonly used as filter templates. Here quality factor Q is a constant that controls the bandwidth and the peak of the filter:

$$H_1(s) = \frac{{\omega_c}^2}{s^2 + \frac{\omega_c}{Q}s + {\omega_c}^2}$$

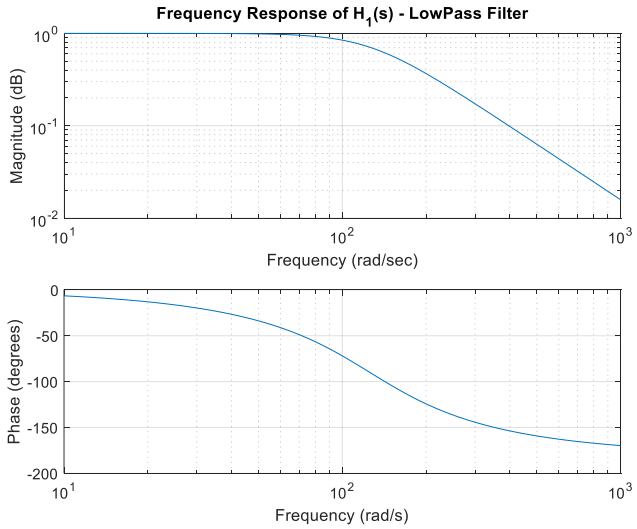
$$H_2(s) = \frac{\frac{\omega_c}{Q}s}{s^2 + \frac{\omega_c}{Q}s + {\omega_c}^2}$$

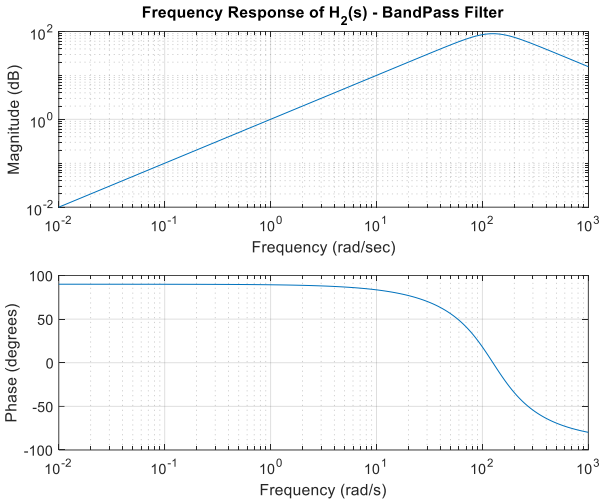
$$H_3(s) = \frac{s^2}{s^2 + \frac{\omega_c}{Q}s + {\omega_c}^2}$$

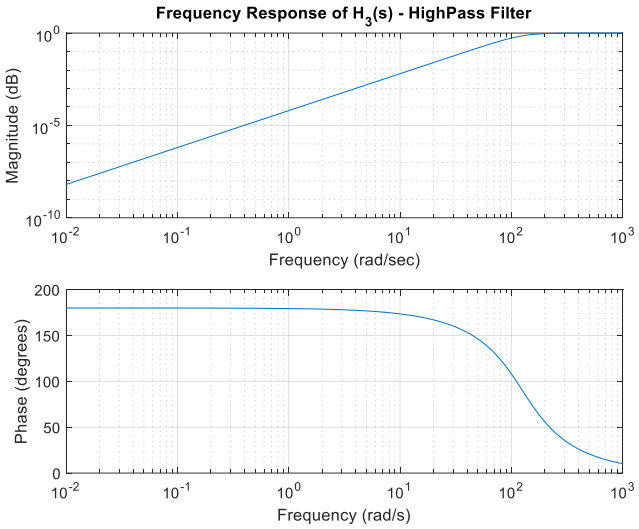
Now let's examine the use of MATLAB frequency response plotting tool to plot the magnitude of these transfer functions. To maintain consistency between these comparisons, $Q = 0.707$, cutoff frequency $f_c = 20Hz$. Below is some code snippet that you may find useful:

```
1. omega_c = 20*2*pi;                % cutoff frequency in radian
2. Q=0.707;                          % Quality factor Q
3.
4. NUM = [0, 0, omega_c^2];
5. DEN = [1,omega_c/Q,omega_c^2];
6. freqs(NUM,DEN,4096)               % Plot frequency response of analog filters
7.
```

Task 4 (10%): Please plot the frequency response of analog filters H_1, H_2, H_3 below. Clearly label the filter types in your plot.



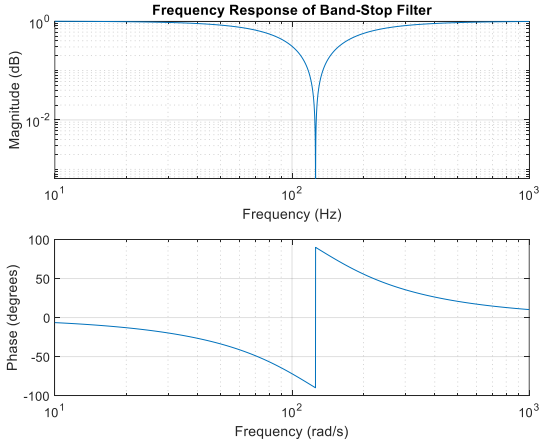




Task 5 (10%): Based on the activities and observations in Task 4, please create a band-stop filter template. Unlike the three types mentioned before, a band-stop filter will stop at a certain band and let other frequencies to pass through, i.e. a complement of a band-pass filter. What will the transfer function be in this case? Write your proposed transfer function and the corresponding frequency response MATLAB plots below.

$$H(s) = \frac{s^2 + w_c^2}{s^2 + \frac{w_c}{Q}s + w_c^2}$$

Low frequency and high frequency pass through unchanged but the filter removes frequencies at ≈ 20hz



3.3 Analog filter prototypes

In section 3.2, we have worked with some general filter types with some success. ***However, it is important to point out that the* efficiency and performance of these filters are very poor.** The trade-offs between different design factors need to be further explore to optimize filter performance. Extensive research and decades of industrial implementation contributed to the abundance of the design literature of analog filters. Among so many transfer functions that can meet the magnitude/phase requirements of your design specifications. Which one to choose really lies in what type of constraints you have and what types of compromises you are ready to take. Some of the commonly used filter prototypes are:

- Butterworth: flat in the passband and the stopband, however, with a bigger transition band between the pass- and the stopband.
- Chebyshev I: reduces the transition band (a steeper roll-off) at the expense of ripples in the passband.
- Chebyshev II: also known as the inverse Chebyshev filters, reduces the transition band at the expense of ripples in the stopband.
- Elliptic: with equalized ripple (equiripple) in both the passband and the stopband.

Table 1. Common Prototype Filter Specifications

Filter Type	fc Role	Rp (Passband Ripple)	Rs (Stopband Attenuation)	Roll-off Steepness
Butterworth	-3 dB point	Not used	Not used	Least steep
Chebyshev I	Passband edge	Required (e.g., 3 dB)	Not used	Steeper
Chebyshev II	Stopband edge	Not used	Required (e.g., 40 dB)	Steeper
Elliptic	Transition region	Required (e.g., 3 dB)	Required (e.g., 40 dB)	Sharpest

Task 6 (10%). In the figure below, the magnitude spectra are shown to represent four different filters with similar design specifications, please identify each one based on the characteristic of each filter (please directly label your answer in the plot).

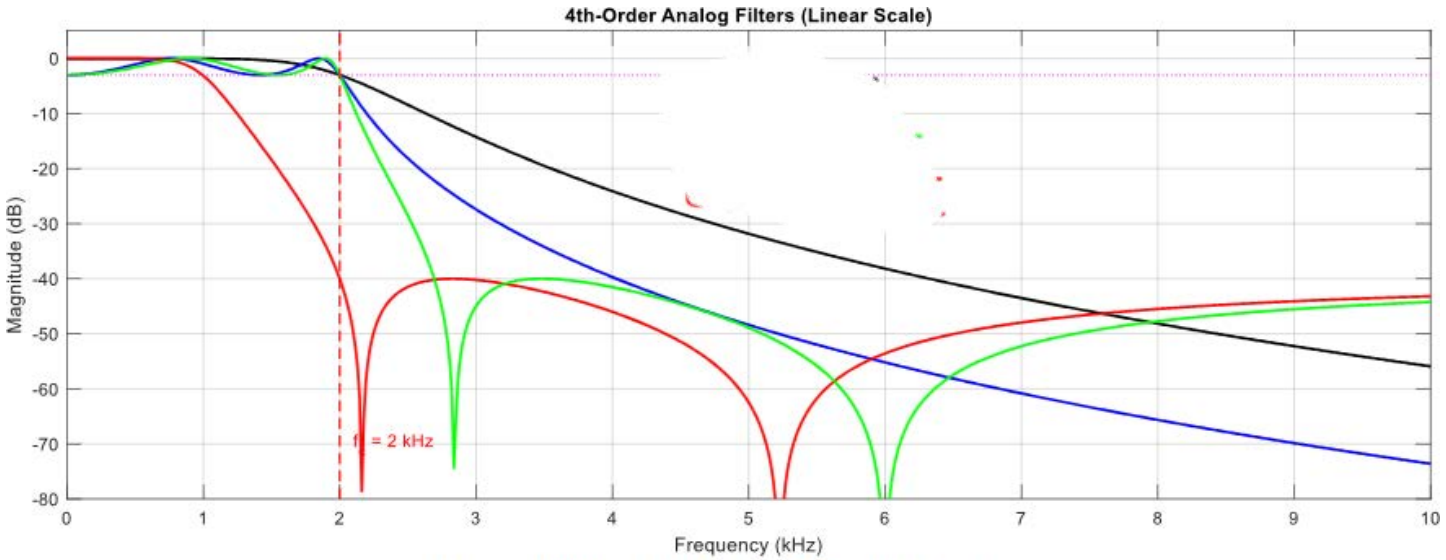


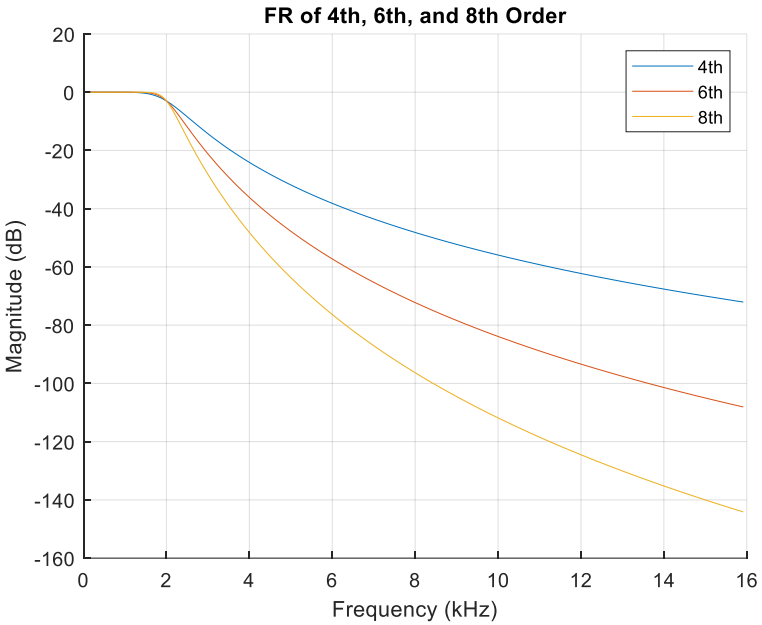
Figure 4. Different Analog Filter Prototypes

3.4 Analog filter design

There are various ways to design analog filters in MATLAB. Below is an example of using built-in function butter() to compute an n-th order Butterworth filter with cutoff frequency fc. Here cutoff frequency fc is defined as the frequency at which the filter's magnitude response drops to **-3 dB** (≈ **70.7%** of the passband amplitude).

```
1. [zb, pb, kb] = butter(n, 2*pi*fc, 's');     % Analog Butterworth LPF
2. [bb, ab] = zp2tf(zb, pb, kb);               % Transfer function
3. [hb, wb] = freqs(bb, ab, 4096);              % Frequency response
4. plot(wb/(2*pi*1e3), mag2db(abs(hb)))     % Correct: x-axis in kHz
```

Task 7 (10%). Based on your given frequency, plot on the same figure, the frequency response (magnitude only) of the 4th, 6th, and 8th order Butterworth filters, with the cutoff frequency fc being 2kHz. Record the MATLAB plot and analyze what is the impact of increasing the filter order.



The Higher the order the steeper the roll off near the cutoff frequency will be.

3.5 Digital filter design.

3.5.1 Digital filter design using MATLAB functions

Once we know how to design analog prototype filters, we can move on to designing the digital equivalent of these filters. You can use the digital filter designer using MATLAB. Similar to the analog filter design, you can modify the parameters of the butter() function. An example code below shows how to use butter() to design a **digital** low pass filter and then to plot the frequency response:

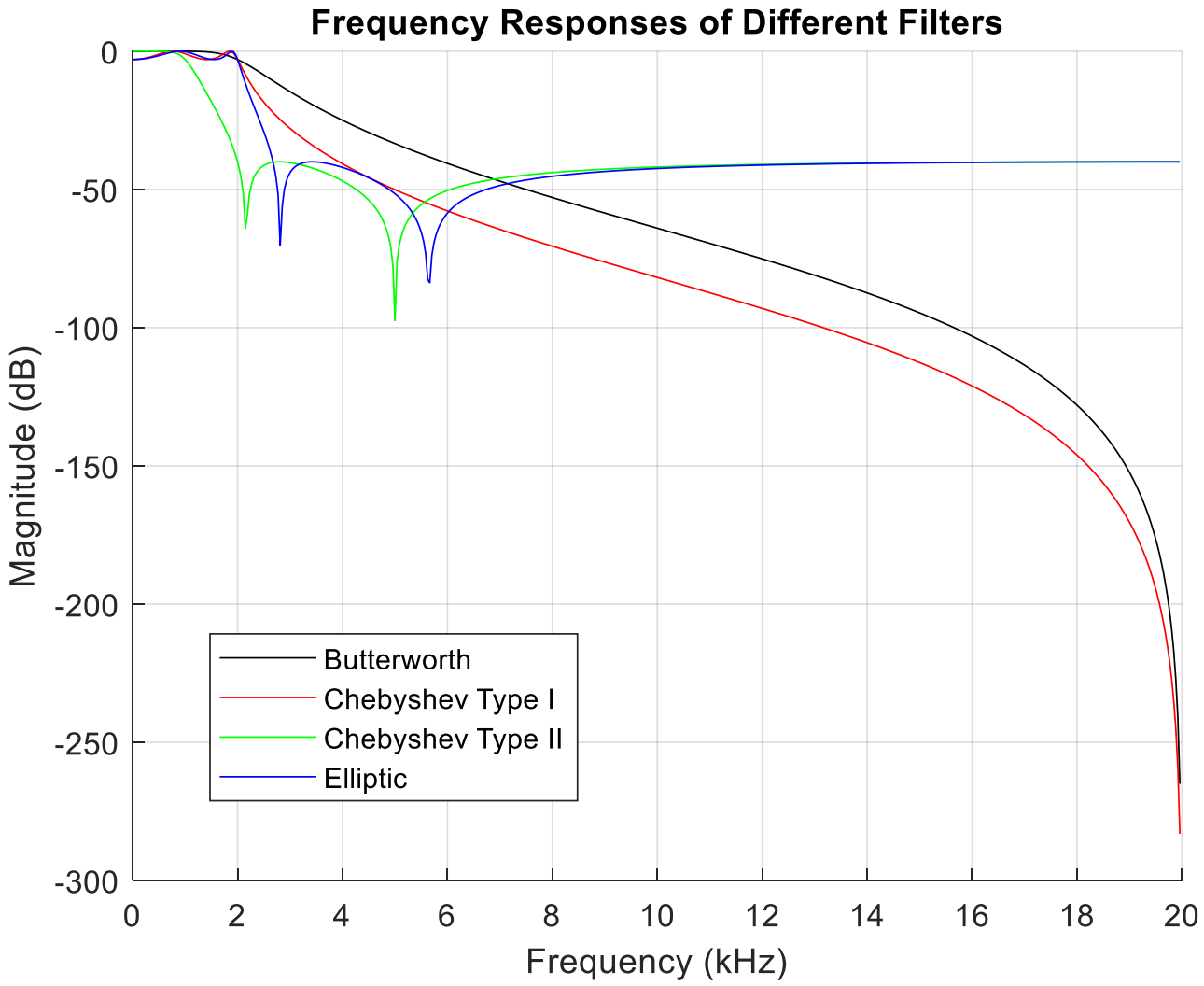
```
fc_norm = fc/(fs/2);                        % Normalized frequency for digital filter
[bb, ab] = butter(n, fc_norm);              % Digital Butterworth LPF
f = linspace(0, 15000, 1000);               % Frequency vector (0 Hz to 15 kHz)
w = 2*pi*f/fs;                               % Normalized frequency [0, π]
[hb, wb] = freqz(bb, ab, w);
plot(f/1000, 20*log10(abs(hb)), 'k', 'LineWidth', 1.5);
figure;
plot(wb, 20*log10(abs(hb)), 'k', 'LineWidth', 1.5); grid on;
xlabel('Frequency (kHz)');
ylabel('Magnitude (dB)');
title('4th-Order Digital Filters (f_c = 2 kHz, fs = 40 kHz)');
```

Task 8 (10%). Similar to butter(), you can use MATLAB functions cheby1(), cheby2(), ellip() to design other types of digital filters. Based on this information, please design plot all four types of prototype filters (Butterworth, Chebyshev Type I, Chebyshev Type II, and elliptic filter) in the same plot with the same cutoff frequency fc =2kHz, and same filter order. Assuming the sampling frequency is 40kHz, passband ripple = 3 dB and stopband attenuation = 40 dB. Record your code and the plot below.

Hint: information on how to design the other common filters can be found in the example here.

```
fc = 2000;                    % Cutoff frequency (Hz)
Fs = 40000;                   % Sampling frequency (Hz)
ripplePB = 3;                 % Passband ripple (dB)
attenuationSB = 40;       % Stopband attenuation (dB)
filterorder = 4;              % Filter order
cutf = 2*fc/Fs;               % Normalized cutoff frequency for digital filter

```

```
[Bbutter,Abutter]=butter(filterorder, cutf);
[Bcheby1,Acheby1] = cheby1(filterorder, ripplePB, cutf);
[Bcheby2,Acheby2] = cheby2(filterorder, attenuationSB, cutf);
[Bellip,Aellip]= ellip(filterorder, ripplePB, attenuationSB, cutf);

[H_butter, f]    = freqz(Bbutter, Abutter,[],Fs);
[H_cheby1,~]  = freqz(Bcheby1,Acheby1,[],Fs);
[H_cheby2,~]  = freqz(Bcheby2,Acheby2,[],Fs);
[H_ellip,~]      = freqz(Bellip,Aellip,[],Fs);

figure;
hold on;
plot(f/1000, 20*log10(abs(H_butter)), 'k'); %Butter
plot(f/1000, 20*log10(abs(H_cheby1)), 'r'); %Chev1
plot(f/1000, 20*log10(abs(H_cheby2)), 'g'); %chev2
plot(f/1000, 20*log10(abs(H_ellip)), 'b'); %ellip

xlabel("Frequency (kHz)");ylabel("Magnitude (dB)");
title('Frequency Responses of Different Filters');
legend('Butterworth', 'Chebyshev Type I', 'Chebyshev Type II', 'Elliptic');
grid on;
hold off;
```



3.5.2 Digital filter design using filterDesigner tool

In MATLAB, there is a dedicated digital filter design tool called "filterDesigner". The information can be found here. Type filterDesigner in MATLAB to invoke the digital filter design GUI (graphic user interface).

Let's explore filterDesigner. Record your specifications as the following:

- Response type: Lowpass
- Filter order: 4
- Design method: IIR -> Butterworth

Once you click the "Design Filter" button, the magnitude response view will appear. For additional information of the characteristics of this filter, you can click Analysis -> from the menu bar to see other analytical tools.

Task 9 (10%). Filter information can be exported for additional analysis. Simply go to Analysis - > Filter Coefficients to review the actual transfer function coefficients. Remember that this is the corresponding digital filter you have designed in the previous task. Please conduct your own research to find at least two ways to export the designed filter for your signal processing application. Describe how to implement method to filter signals below.

One method is to generate MATLAB code from the design filter. Another common method (apparently but seems like 1st method with extra steps) is to export to workspace as a dfilt object.

1st method to implement it into signals is the same way as task 8

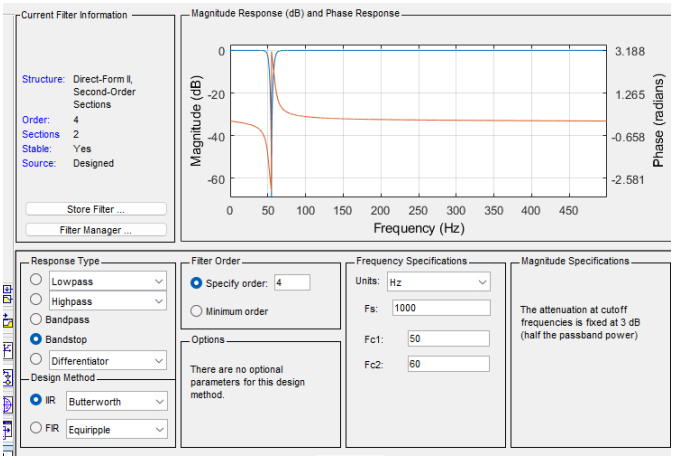
2nd method to implement it into signals is having the object with a sub declare i.e. (.Numerator, .Denominator)

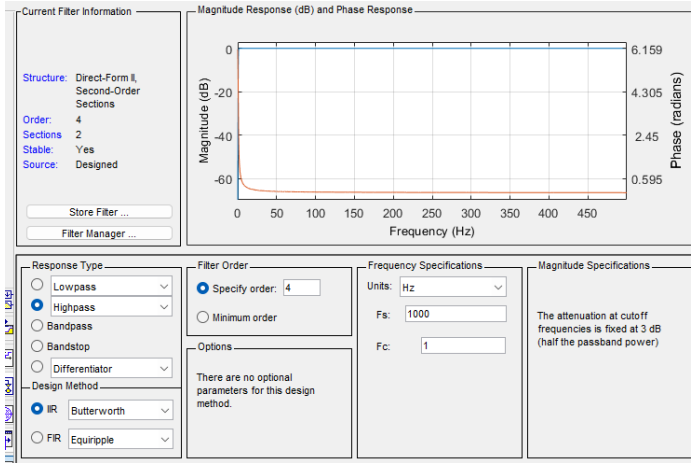
3.6 Practice

Task 10 (20%): Now we need to design a filter to denoise a real-life digital signal (sampled at 1kHz). There are two major sources of noise:

- Due to the powerline noise appears between 50-60Hz, the signal quality is significantly compromised.
- There is an ultra-low frequency drift (0-1Hz) that is due the limitations of the data acquisition.

Please use what we have learned in this lab to design a proper filtering signal to remove the aforementioned noise. Record your code/or settings in filterDesigner. Also, include the magnitude and phase spectrum of the final filter.






```
1. Fs = 1000;
2. [B_bsfd,A_bsfd]=butter(4, [50 60]/(Fs/2), 'stop');
3. [B_Hpfd,A_Hpfd]=butter(4,1/(Fs/2),'high');
4.
5. btot=conv(B_bsfd,B_Hpfd);
6. atot=conv(A_bsfd,A_Hpfd);
7.
8. [H,F]=freqz(btot,atot,[],Fs);
9.
```

```
10. figure;
11. subplot(2,1,1);
12. plot(F,abs(H));
13. title('Magnitude Spectrum of Final Filter');
14. xlabel('Frequency (Hz)'); ylabel('Magnitude'); grid on;
15.
16. subplot(2,1,2);
17. plot(F, angle(H));
18. title('Phase Spectrum of Final Filter');
19. xlabel('Frequency (Hz)'); ylabel('Phase (Radians)'); grid on;
```