Lab 4 IIR Filters

Objectives:

- Use filter design and analysis tools to create IIR filters based on general filter specification.
- Understand the impact of the placement of poles and zeros on the frequency response of the filter.
- Understand the impact of different computational structures for filter implementations on filter performance.

1. Background

Infinite impulse response (IIR) filters have some advantages over finite impulse response (FIR) filters. An IIR implementation can meet filter specifications with less computations than an FIR implementation; however, IIR filters have nonlinear phase responses, are potentially unstable, and have increased sensitive to numerical errors.

Like FIR filters, IIR filters are linear time-invariant (LTI) systems that can realize a wide range of frequency responses. IIR implementations with specified stop-band, pass-band, and transition-band properties typically require fewer filter taps (coefficients) than an FIR filter meeting similar specifications. This leads to a significant reduction in computational complexity and signal/group delay through the filter. However, IIR filters have poles as a result of feedback, which cause instability when poles are not inside the unit circle. Feedback can increase the sensitivity to errors from finite arithmetic computations (especially for fixed-point processors). In addition, IIR filters result in nonlinear phase distortion (frequency components delayed by different amounts, especially near the transition bands).

Up to this point, filters used in the laboratory assignments have been implemented with the ***Direct Form***, which is suggested by converting the transfer function (TF) directly to a difference equation, using the TF in the following form:

$$\hat{H}(z) = \frac{\sum_{m=0}^{M} b_m z^{-m}}{1+\sum_{n=1}^{N} a_n z^{-n}} = \left(\frac{\sum_{m=0}^{M} b_m z^{-m}}{1}\right)\left(\frac{1}{1+\sum_{n=1}^{N} a_n z^{-n}}\right) \tag{1}$$

The rational polynomial in z for Eq. (1) is factored into an all-zero filter (FIR) followed by an all-pole filter. Recall the product of TFs corresponds to connecting the systems in series (cascading) in the time-domain, where the output of one filter becomes the input to the next. The factoring of Eq. (1) suggests the *direct form I* implementation shown in Fig. 1.

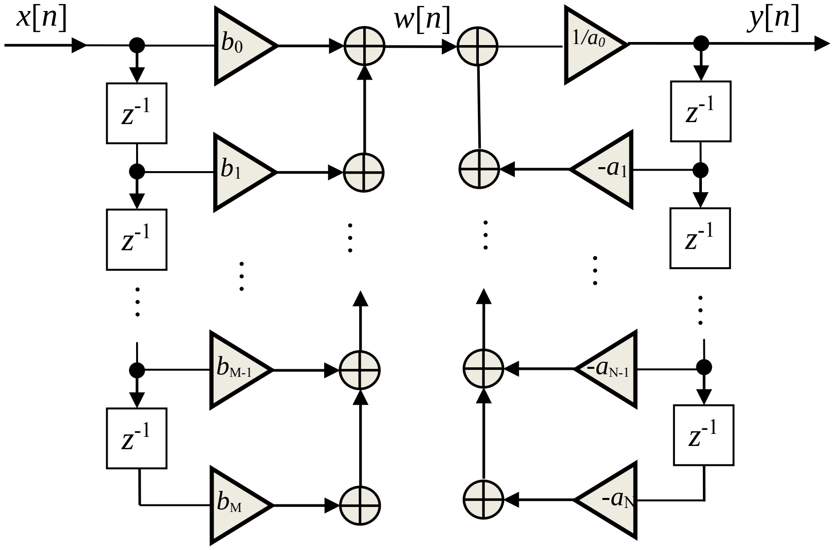


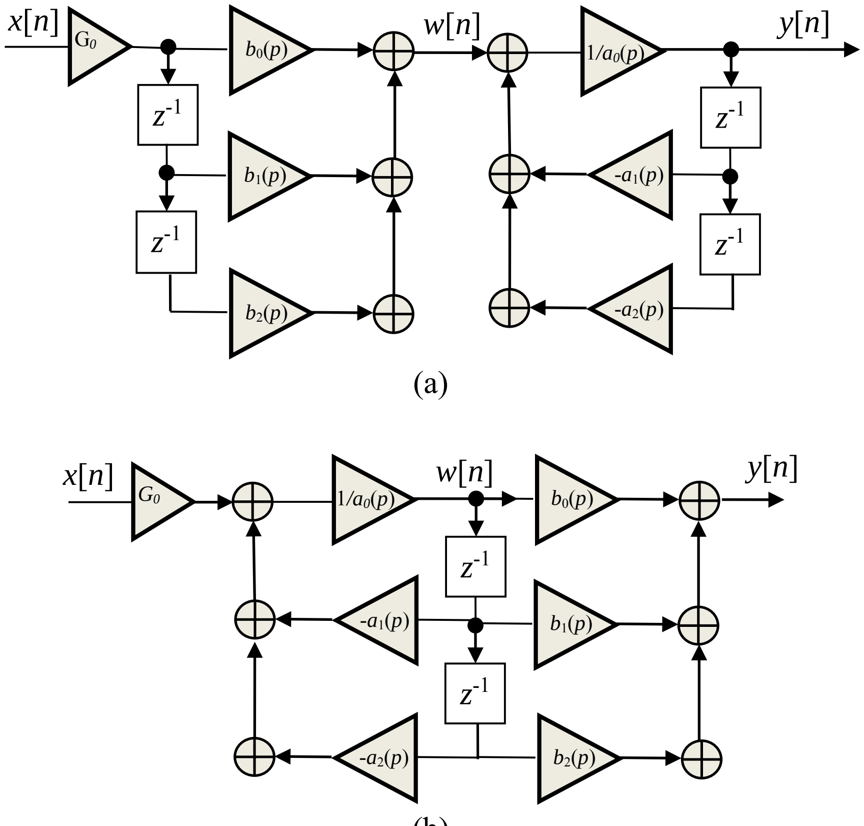
Figure 1. Computational illustration of direct form I implementation of an IIR filter. The square blocks represent unit delays, the triangles represent multiplies, and the circles represent accumulators. The variable $w[n]$ is an intermediate value being the output of the all-zero component and the input to the all-pole component of the filter, with a_0=1 for the IIR filter of Eq. (1).

For higher order filters, the direct-form implementation involves long delays (M samples for the input and N samples for the output, which are typically realized with a sequence of memory cells and a rotating pointer). This may lead to numerical instability from rounding of the filter coefficients and its ill-conditioned impact of pole location. For example, with higher-order polynomials the roots may shift widely for small changes (i.e. rounding errors) in the filter coefficients. Therefore, it is hard to predict how small coefficient errors will impact the pole positions of a high-order direct-form implementation. Different implementations (computational structures) can be realized by factoring and representing the polynomials of Eq. (1) in different ways.

Consider factoring numerator and denominator polynomials into second-order stages as follows:

$$\hat{H}(z) = \frac{\sum_{m=0}^{M} b_m z^{-m}}{1 + \sum_{n=1}^{N} a_n z^{-n}} = G_0 \prod_{p=1}^{P} \frac{b_0(p) + b_1(p)z^{-1} + b_2(p)z^{-2}}{1 + a_1(p)z^{-1} + a_2(p)z^{-2}} \tag{2}$$

where argument *p* denotes each filter stage. The product of second-order stages are derived from the original direct-form single stage filter coefficients by factoring. When factoring into second order stages and complex poles exist, pairs of complex conjugate poles are combined to ensure real coefficients. G_0 is the overall gain, which can be factored out of all the *b* coefficients. The form of Eq. (2) suggests cascading second-order IIR filters in series. This implementation is referred to as the ***Cascade*** form. One way to improve numerical stability over a higher-order direct form implementation is to implement the IIR filter as a cascade of second-order *direct form* sections. The data flow for a second-order, *direct-form* implementation is shown in Figure 2. Note that for the *direct form II* implementation, shown Fig. 2b, the delayed samples are neither input nor output samples, but are intermediate values *w*[*n*]. This implementation can be derived by first applying the all-pole component of the TF and then making that the input of the all-zero component (just the opposite multiplication order of the *direct form I* implementation shown in Eq. (1)).



(a)

(b)

Figure 2. (a) Single stage *direct form I cascade* implementation of an IIR filter. (b) Single stage *direct form II cascade* implementation of an IIR filter. G_0 is multiplier affecting the overall gain of the filter (does not change relative spectrum or waveform values). This is only applied at one stage. And a_0=1 in order to correspond to the IIR filter of Eq. (2).

There are several IIR filter design methods that primarily affect how ripple is distributed over the pass and stop bands. The Matlab functions for the main IIR filter designs are Elliptical (*ellip*), Butterworth (*butter*), and Chebyshev (*cheby1* and *cheby2*). The unique features of these approaches will be examined in the laboratory assignment.

Pole-zero impact on frequency response:

1. Pole-zero diagrams are useful for understanding the approximate behavior of a filter. Recall that points around the unit circle in the *z*-plane corresponds to a sinusoidal excitation with a frequency equal to the angle made with respect to the positive real axis. So starting at z=1+j0 corresponds to ω=0 and ending at z = -1+j0 corresponds to ω=π radians or f = .5 Hz (normalized). For non-normalized frequencies the upper limit is scaled up to the half the sampling frequency, (2π)f_s/2 radians (f_s is the sampling frequency) or f_s/2 Hz. Regions along this path over the unit circle that are close to *zeros* correspond to regions where signal attenuation occurs (TF numerator gets small). The closer the *zero* is to the unit circle the more dramatic the attenuation for frequencies near the zero. Regions along the unit circle that are close to *poles* correspond to frequency regions of amplification (TF denominator gets small). The closer the *pole* is to the unit circle the more dramatic the amplification around the frequency corresponding to the pole angle (resonance). Consider the following TF that shows the relationship between a pair of complex conjugate poles (expressed in polar form: $r\exp(\pm j\theta)$) and a second-order IIR filter:

$$\hat{H}(z) = \frac{1-r}{(1-r\exp(j\theta)z^{-1})(1-r\exp(-j\theta)z^{-1})} = \frac{1-r}{(1-2r\cos(\theta)z^{-1}+r^2z^{-2})} \qquad (4)$$

Figure 3 shows the locations of the poles of this filter in the z-plane. If the system has real coefficients, the poles must occur in complex conjugate pairs, i.e. $p_1 = re^{j\theta}$, $p_2 = re^{-j\theta}$ where r is the distance from the origin, and θ is the angle of p_1 relative to the positive real axis.

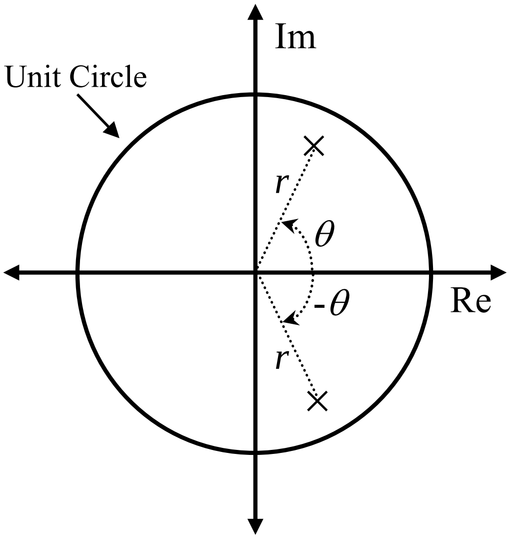


Figure 3. Pole location of an IIR Filter.

Consider the frequency response of a simple all pole IIR filter with sampling rate $f_s = 1$ Hz (normalized) and poles at r = 0.75 and $\theta = \pm 45^{\circ}$. The TF with these poles can be written as (ignore overall TF gain for now, since it does not affect the shape of the TF):

$$\hat{H}(z) = \frac{1}{(z - .75\exp(j\pi/4))(z - .75\exp(-j\pi/4))} =$$

$$\frac{1}{z^2 - 0.75\sqrt{2}z + (.75)^2} = \frac{z^{-2}}{1 - 1.061z^{-1} + 0.562z^{-2}} \qquad (5)$$

Use the *freqz*() command with b = [0, 0, 1] and a = [1, -1.061, 0.562], to obtain 1024 points with sampling frequency 1. Then plot magnitude of response:

```
>> [h,f] = freqz(b,a,1024,1);
>> plot(f,abs(h))
>> xlabel('Hz')
>> ylabel('TF Magnitude')
```

The resulting plot with annotations is shown in Fig. 4. The expected frequency peak is close to the pole angle θ which is converted to Hz by:

$$f = \frac{f_s}{2}\frac{\theta}{\pi} \quad \text{for } \theta \text{ in radians} \qquad f = \frac{f_s}{2}\frac{\theta}{180} \quad \text{for } \theta \text{ in degrees} \qquad (6)$$

The reason the pole angle does not match up exactly with the TF peak is that the other complex conjugate pole is exerting an influence on the TF magnitude as well. This can be seen from computing the magnitude of the TF in Eq. (5) as (square root of real part squared plus imaginary part squared, applied to each zero or pole factor) :

$$\left|\hat{H}(z=\exp(j\omega))\right| =$$

$$\frac{1}{\sqrt{(\cos(\omega)-.75\cos(\pi/4))^2+(\sin(\omega)+.75\sin(\pi/4))^2}\sqrt{(\cos(\omega)-.75\cos(\pi/4))^2+(\sin(\omega)-.75\sin(\pi/4))^2}} \tag{7}$$

where each radical factor represents the distance of the pole to the point *z* on the unit circle. As *z* is moved over the unit circle, the distance to the poles are the factors the denominator of Eq. (7), and while one factor is minimized at the pole angle, the other factor from the complex conjugate is not, and therefore influences the actual resonant frequency. As the poles are closer to the unit circle the influence of other poles and zeros are lessened. The same pattern is true for zeros, except they will attenuate the TF when their factors become small.

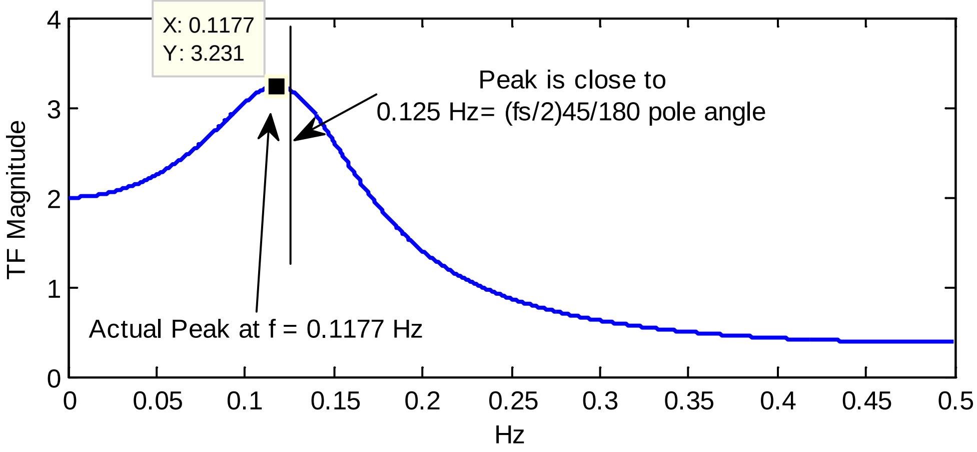


Figure 4. Frequency response for TF with poles at $z_{1,2}$=0.75exp(±jπ/4). Note actual peak position 0.1177 is slightly off from predicted peak at 0.125 Hz based on pole angle.

2. Pre-Laboratory Assignment

1. Derive the TF of filter from the block diagram shown in Fig. 1. (Hint: obtain TF between *x* and *w* and then between *w* and *y* and use

$$\hat{H}(z)=\frac{\hat{Y}(z)}{\hat{X}(z)}=\left(\frac{\hat{W}(z)}{\hat{X}(z)}\right)\left(\frac{\hat{Y}(z)}{\hat{W}(z)}\right) \tag{3}$$

to find the TF) .

2. Show that the block diagrams of Fig. 2 (a) and (b) are mathematically equivalent by showing that their TFs are equal.

3. Consider the design of a 4th order low-pass elliptical filter with a cutoff of 1000 Hz for a sampling rate of 8000 Hz. The pass-band should have ripple less than .5 dB and the stop-band should have ripple less than 15 dB. (In other words, the difference between the minimum and maximum value in the passband should not exceed .5 dB and the highest point in the stopband should not be within 15dB of the maximum passband value.) Determine where the stop band begins in this case and verify the ripple specifications are met.

   (**Hint:** use with the following command (see help *ellip*) to generate filter coefficients:

   ```
   >> [be4, ae4] = ellip(4, 0.5, 10, 1000/4000);
   ```

   Then check to see if this filter meets the specifications by plotting the magnitude response obtained with *freqz*() and verifying the ripple in pass-band (0 to 1000 Hz) does not exceed 0.5 dB, and the stop-band begins where the spectrum first drops below and stays below 15 dB. You can use the zoom and data curser functions on the figure menus, as well as the axis command or the axis properties tool bar on the figure to zoom in on the critical parts of the graph needed verify the pass-band properties and find the transition bandwidth. Include these figures (zoomed in versions) in the solution to the prelab problem.

4. For the filter of the previous problem, write the filter TF as a ratio of 4th order polynomials. Then use the *roots* and *poly* commands to write this filter as a product of two 2nd order polynomial ratios (cascade form). Scale the rational polynomials so the gain at each stage for DC (0 Hz) is 1 (**hint:** Substituting $z = \exp(j0) = 1$ in the $H(z)$ corresponds to the response for a DC excitation. If the TF is then scaled by reciprocal of that gain, the resulting gain is 1 for that stage). Also find the overall gain at DC, G_0, of the original system. Then scale your cascaded system so its overall DC gain is the same as the original. So each individual 2nd order cascaded system has a gain of 1, but there is a final gain stage that return of the overall gain to its original value. This intermediate scaling is often done to improve numerical stability. Draw a block diagram (*direct form I*) for the implementation using the 4th order polynomials (delays of up to 4 units) and another block diagram (cascade with direct form II on each stage) for its implementation using two 2nd order systems. Don't forget to label the multipliers with the actual values of the coefficients you found. You can use the MATLAB command *tf2sos* to check your work, but you should write your own script that converts the 4th order system to 2 second order systems must be written using *roots* and *poly*. Hand in the script you wrote to do this conversion in addition to the TFs and block diagrams.

5. Read the help files in Matlab for *butter*, *ellip*, *cheby1*, and *cheby2*. Read them to understand how each function operates and the unique features for each filter. Also read the documentation on the *fdatool* command, which launches a GUI for convenient design and analysis of filters. There is nothing to hand in for this particular problem, just read the help files and become aware of the important features of each.

3. Laboratory Assignment

These lab exercises focus on the effects of pole placement, robustness of filter implementations, and the nature of specifications for the different IIR filters.

1. Given a system with complex conjugate poles of the form: $z_p = r\exp(\pm j\theta)$, plot the magnitude responses $\left|H_1(\exp(j\omega))\right|$ on $0 \le \omega < \pi$ for pole angle $\theta = \frac{\pi}{4}$ for each of the following r values:
   a) $r = 1.1$
   b) $r = 0.9$
   c) $r = 0.8$

   Put all three plots on the same figure for a comparison. Likewise plot the pole-zero diagrams for each filter on the same plot for verification. **Describe the impact of pole placement to the TA using the resulting plot. Also comment on whether stability can be inferred from the magnitude response. (Hint:** Find the coefficients of the TF in Eq. (4) by substituting in the pole values, and use *freqz*() to obtain the magnitude response for plotting).

For Exercises 2-5, include the magnitude response, pole-zero diagram, and impulse response in the results section of your report.

2. Use the MATLAB command *butter*() to design a 5th-order low-pass IIR filter with sampling frequency of 2 kHz and a pass-band edge (or cut-off) frequency of 500 Hz. Plot the magnitude and phase responses using *freqz*(). Plot the pole and zero locations diagram using *zplane*(). Plot the significant part of the impulse response using *filter*() (i.e. stop plotting after the values get too small to see).

3. Use the MATLAB command *cheby1*() to design a 5th-order low-pass IIR filter with a sampling frequency of 2 kHz, a pass-band edge frequency of 500 Hz, and a pass-band ripple of 0.5 dB. Plot the magnitude and the phase responses. Plot the pole and zero location diagram.

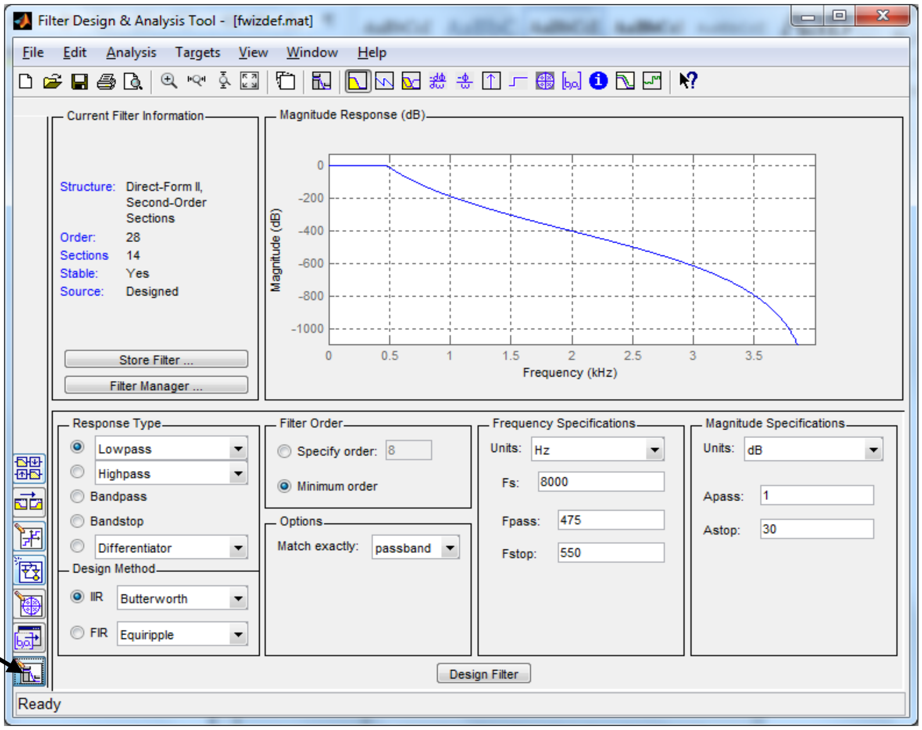
4. Use the MATLAB command *cheby2*() to design a 5th-order low-pass IIR filter with a sampling frequency of 2 kHz, a band edge of 500 Hz, and a maximum stop-band ripple 30 dB below the pass-band response. Plot the magnitude and phase responses. Plot the pole and zero location diagram.

5. Use the MATLAB command *ellip*() to design a 5th order low-pass IIR filter with a sampling frequency of 2 kHz, a band edge of 500 Hz, a pass-band ripple of 0.5 dB, and a maximum stop-band ripple 30 dB below the pass-band response. Plot the magnitude and phase responses. Plot the pole and zero location diagram.

Describe to the TA the difference between the responses of the filters in Problems 3-5. Also comment on the differences in the pole zero-diagrams, and how that influences the magnitude response. Focus on those relative differences that make each of the filters unique and highlight the advantages of each filter.

6. Generate a frequency sweep signal with a 3.5 second duration, with unit amplitude, and sweeps linearly between 0 and 3500 Hz. Filter this signal with each of the filters developed in Exercises 1 through 5 and plot the output. Is this the filtered output as you expected? If not, discuss with the TA.

7. Use the *fdatool* to design a low-pass Butterworth filter with the minimum order required to achieve a pass-band ripple of 1 dB and stop-band ripple of 30 dB. Set the pass-band edge frequency to 475 Hz and the stop-band edge to 550 Hz. See Fig. 6 for an example of an interface for designing and analyzing the filter with the *fdatool* (note this is just an example, the numbers in the text boxes are not consistent with the filter you are asked to design in this exercise). Once the parameters are set to generate the coefficients for the requested filter, click the *Design Filter* button. The default implementation is the cascade SOS Direct Form II. Information regarding the filter just designed is provided in the *Current Filter Information* box on the GUI. From this box note the filter order, sections, and stability for this design. Also generate and copy the impulse response and the pole-zero diagram for this filter copy these to a document for later reference.



Current Filter Information

Structure:	Direct-Form II, Second-Order Sections
Order:	28
Sections	14
Stable:	Yes
Source:	Designed

Response Type

- ● Lowpass
- ○ Highpass
- ○ Bandpass
- ○ Bandstop
- ○ Differentiator

Design Method

- ● IIR: Butterworth
- ○ FIR: Equiripple

Filter Order

- ○ Specify order: 8
- ● Minimum order

Options

Match exactly: passband

Frequency Specifications

Units: Hz
Fs: 8000
Fpass: 475
Fstop: 550

Magnitude Specifications

Units: dB
Apass: 1
Astop: 30

Select Filter Design

Figure 6. Filter Design and Analysis Tool (*fdatool*) dialog boxes for design and analysis GUI.

8. Now go back to the *fdatool* and return to the filter design interface on the GUI and under the edit menu, select *Convert to Single Section*. Then repeat part 7 for generating information and figures for comparison to the SOS sections and comment on the difference to the TA.

9. Through trial and error, find the lowest stop-band frequency such that the direct-form single-stage system becomes stable (i.e. you do not need to find this number beyond a 10 Hz precision). Note that that *Current Filter Information* box provides stability information. You can record this information and note the stop-band frequency that corresponds to when stability is indicated by this box; however, actual stability should be verified from the impulse response. When the single-stage direct form becomes stable, compare the pole-zero diagrams and impulse response for the single stage with those for the cascade implementations. Show the TA the result, **comment on differences, and provide reasons for them.**