ECE1563: Lab Report
Lab 4: IIR and FIR Filter Design

Nate Carnovale and Seth So
Lab Group: 1
TA: Kechen Shu

August 15, 2020

Introduction

In digital processing, computers can only process information in discretized, finite quantities. Therefore analog inputs must be converted to digital before any operators are applied. Theoretically this can be accomplished using the Z-transform to convert the mathematical representation of signal from the time domain to the discretized frequency domain. These signals must be processed further before they are of any use, wchihc can be accomplished through filter design - the practice of imposing constraints on an input signal to control its behaviour.

The purpose of Laboratory 4 was to understand the filters mathematically as well as their practical application in discrete-time signals and systems. The practical example examined in the lab was creating and implementing an IIR smoothing filter for the output of the Arduino Due

Our approach to the lab was to practice the implementation of finite and infinite (FIR and IIR) iimpule response filters in MATLAB. Observations would be made after each controlled change of various parameters by looking at the frequency plots of the figures to see the effect of each parameter on the output. After sufficient understanding, practical realization of an IIR filter was explored on an Arduino Due.

Methodology

Experimental Design

The laboratory explored three main topics: IIR filters, FIR filters, and Microcontroller implementation. These topics could be broken up into 2 main sections, Matlab (IIR and FIR) and Practical (Microcontroller).

In MATLAB, IIR filters were tested by plotting the effects of 6th order Butterworth and Chebyshev filters (cutoff frequency 150Hz) on a specified signal xn. Differences between the two filters were observed by directly comparing their amplitude spectra and frequency response plots. FIR filters tested by observing the 4 different types of window filters built into MATLAB with certain specifications (Bartlett, Chbyshev, Gaussian, and Kaiser). The windows were plotted both independently and as applied to a new signal xn so that the filter could be shown filtering out undesired frequencies from the signal.

For the Microcontroller, an Arduino Due was used to generate the signal. After inputting the signal, code was written in order to produce an output filtered by an averaging IIR filter.

Protocol for Collecting Data

For the IIR and FIR filter sections, all results were plotted and recorded in MATLAB. Values were highlighted to see salient features like peak height and frequency.

For the Microcontroller, all outputs were observed and recorded on the oscilloscope. measurements were aken using both onboard measuring system and confirmed with the multimeter. The effect of the filter was observed by listing the the amplitude as frequency increased towards the folding rate.

Data Analysis

IIR and FIR filters were compared primarily by checking differences in zero-pole plots and seeing how the zero and pole placements affected the frequency responses (phase and magnitude) as well as the magnitude spectra. For example, a zero pole plot with zeros in the left side of the unit circle would yield a low pass filter in the magnitude response, and filtering a signal with a low pass filter should reduce its amplitude components in the higher frequency range.

While minimal effect of the filter could be seen from the oscilloscope image, the amplitude data was much more telling of the filters presence. The coefficients indicated that it was an averaging IIR filter, so expected response was a smoother curve as well as steadier amplitude changes.

Results

IIR Filters

A plot of the amplitude spectrum of the signal $x_a(t)$ prepared, where:

$$x_a(t) = 3cos(10\pi t + \pi/4) + 4sin(240\pi t) + cos(400\pi t + \pi/3)$$

And the signal is sampled at $f_s = 512$ Hz for 1 second.

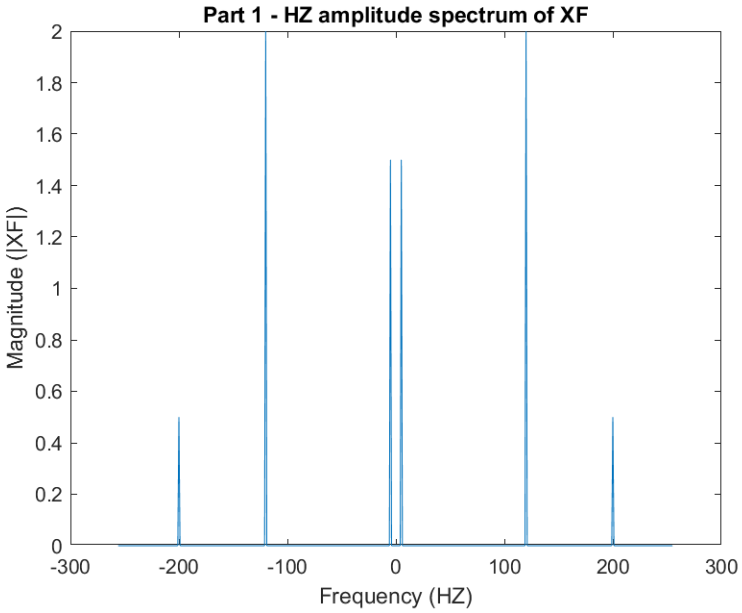

Figure 1: Amplitude Spectrum of $x_a(t)$

A sixth-order low-pass Butterworth filter with a cut-off frequency at 150 Hz was designed in MATLAB. The following pole and zero plot resulted:

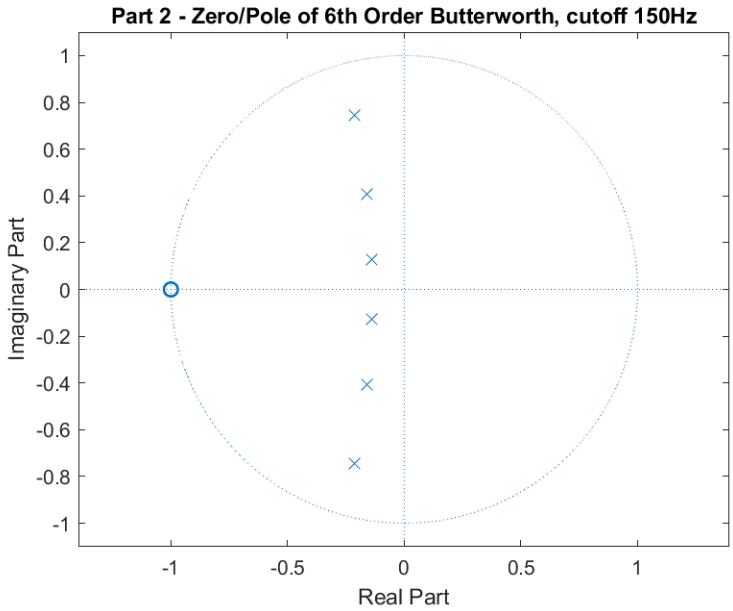

Figure 2: Poles and Zeros of Butterworth Filter with Cut-Off Frequency at 150 Hz

The frequency response of the Butterworth filter from 2 is shown in the figure below:

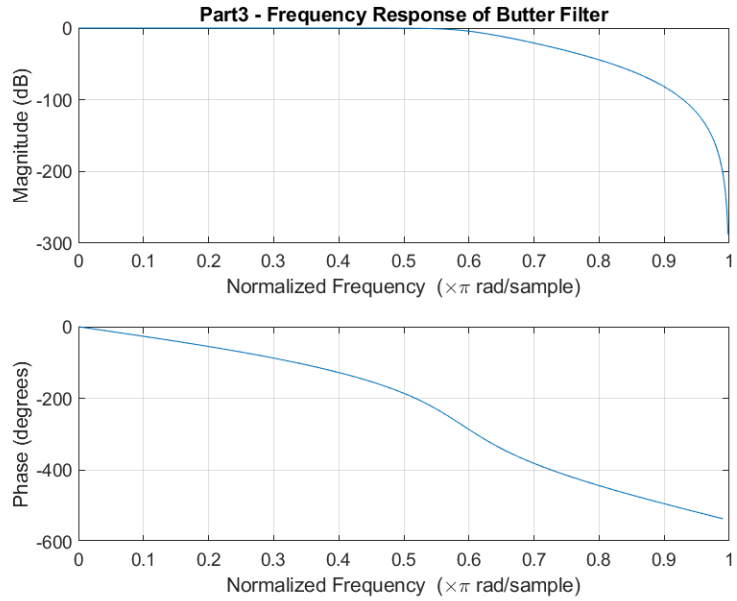

Figure 3: Frequency Response of Butterworth Filter with Cut-Off Frequency at 150 Hz

The sampled version of $x_a(t)$ was filtered to obtain $x_f(nT_s)$ using the Butterworth filter designed above. A plot of the filtered signal is shown in the figure below:

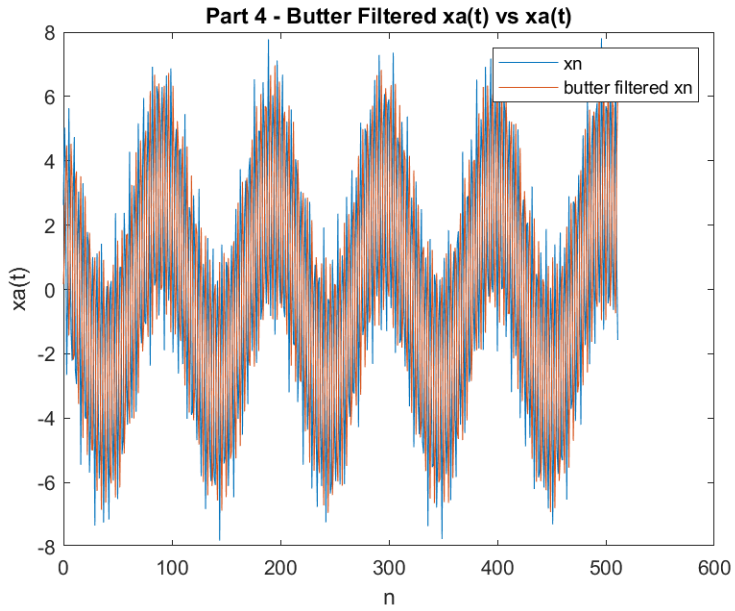

Figure 4: Filtered Signal Using Butterworth Filter

The frequency spectrum of the filtered signal was also realized using MATLAB. It can be seen in the figure below:

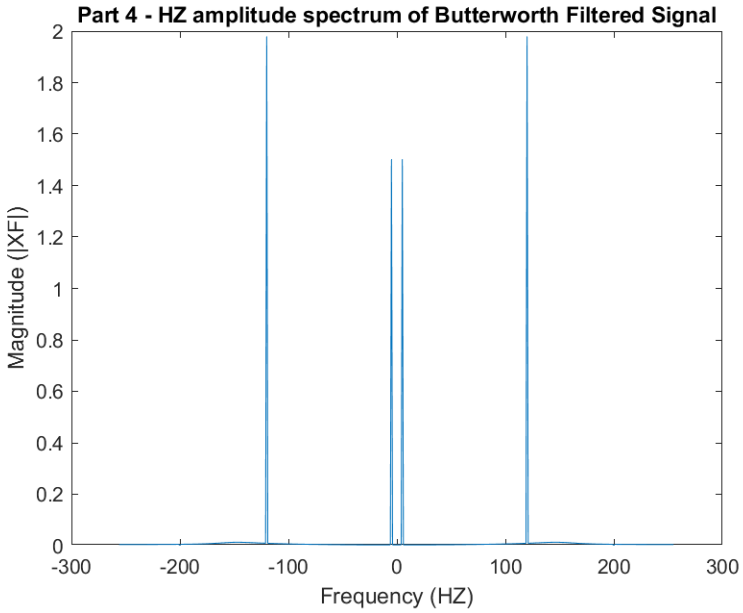

Figure 5: Amplitude Spectrum of Butterworth Filtered Signal

A sixth-order low-pass Chebyshev Type I filter with a cut-off frequency at 100 Hz was designed in MATLAB. The following pole and zero plot resulted:

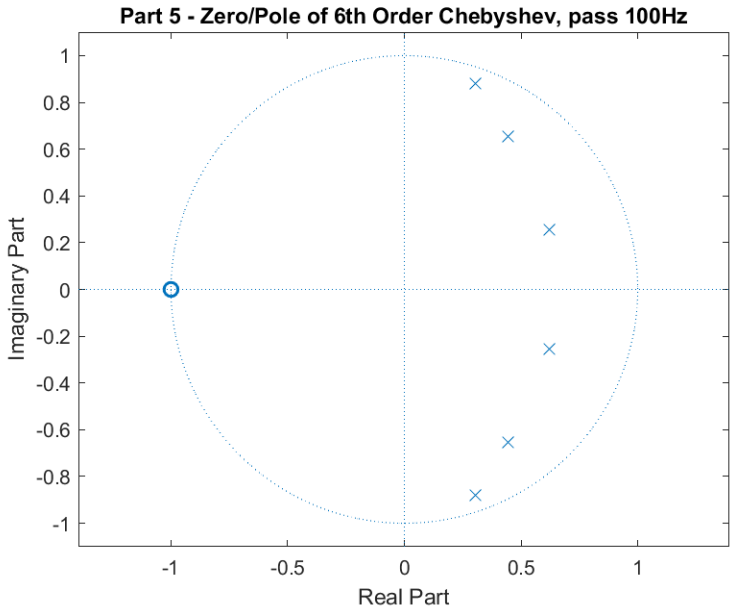

Figure 6: Pole-Zero Plot of Chebyshev Type I Filter with Cutoff Frequency at 100Hz

The frequency response of the Chebyshev Type I filter from 6 is shown in the figure below:

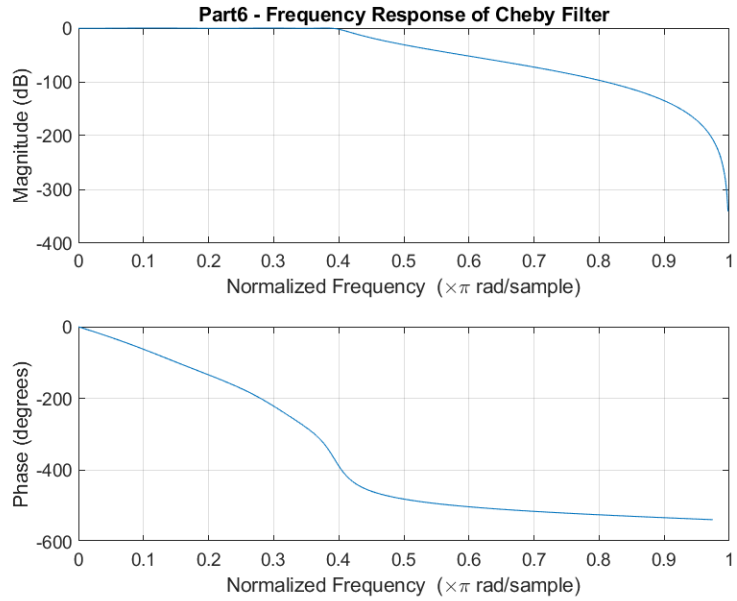

Figure 7: Frequency Response of Chebyshev Type I Filter with Cutoff Frequency of 100 Hz

The sampled version of $x_a(t)$ was filtered to obtain $x_f(nT_s)$ using the Chebyshev filter designed above. A plot of the filtered signal is shown in the figure below:

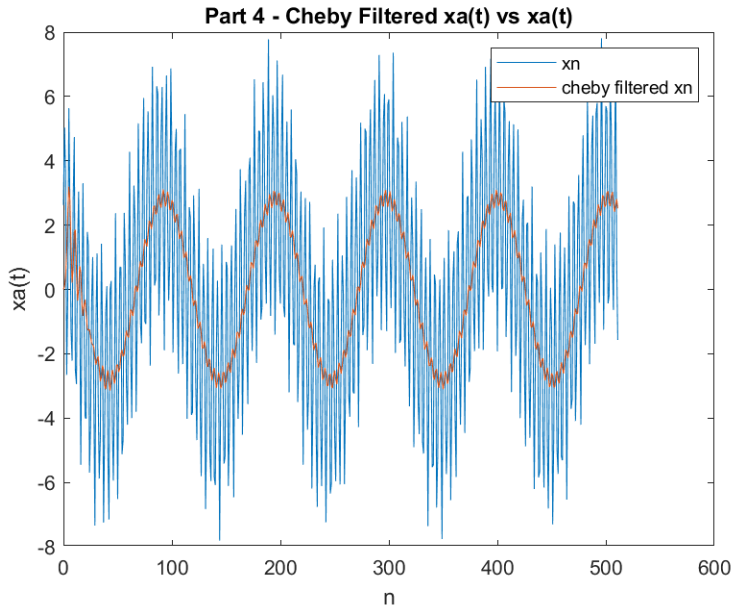

Figure 8: Signal Filtered with Chebyshev Type I Filter

The frequency spectrum of the filtered signal was also realized using MATLAB. It can be seen in the figure below:

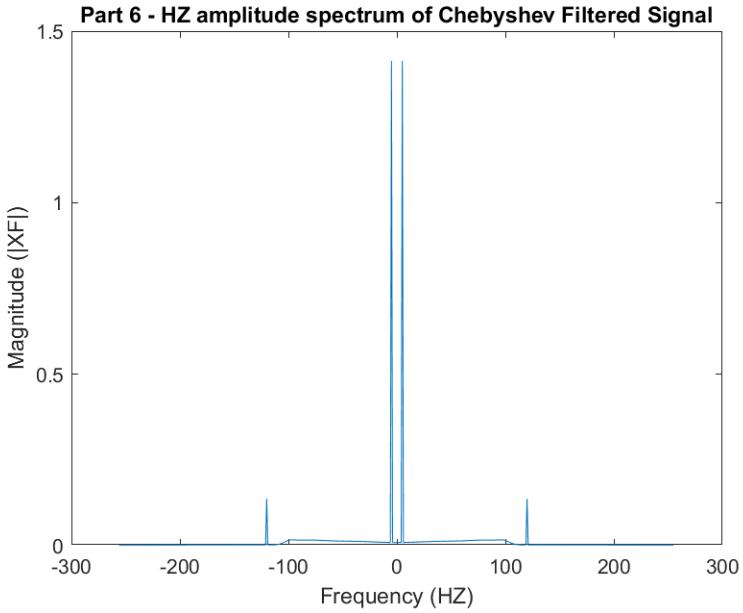

Figure 9: Amplitude Spectrum of Chebyshev Type I Filtered Signal

FIR Filters

FIR Filters were designed using various window functions utilizing the 'window' function in MATLAB.

A Bartlett window was designed with N = 31. The time domain plot and the amplitude spectrum can be seen in the same figure below:

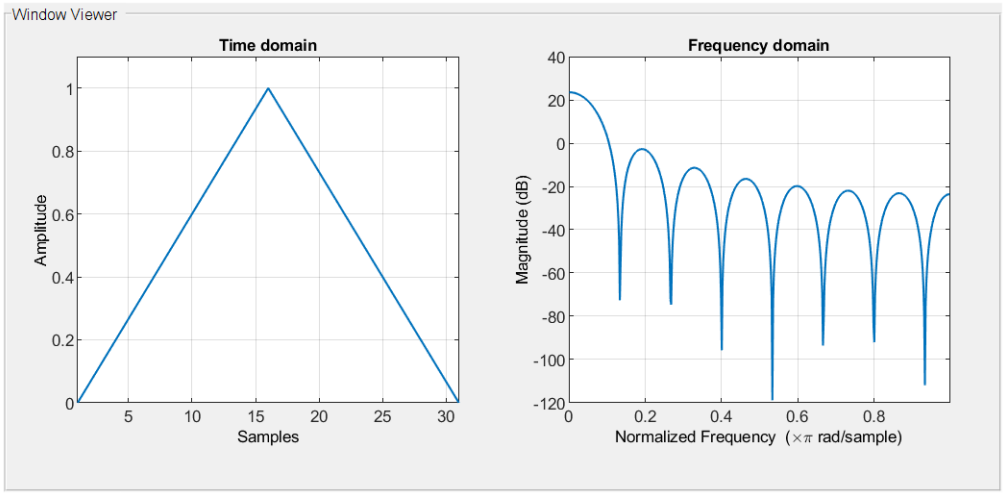

Figure 10: Time Domain Plot and Amplitude Spectrum of Bartlett Window

A Chebyshev window was designed with N = 31. Moreover, the sidelobe magnitude was 50 dB below the mainlobe magnitude for the design. The time domain plot and the amplitude spectrum can be seen in the same figure below:

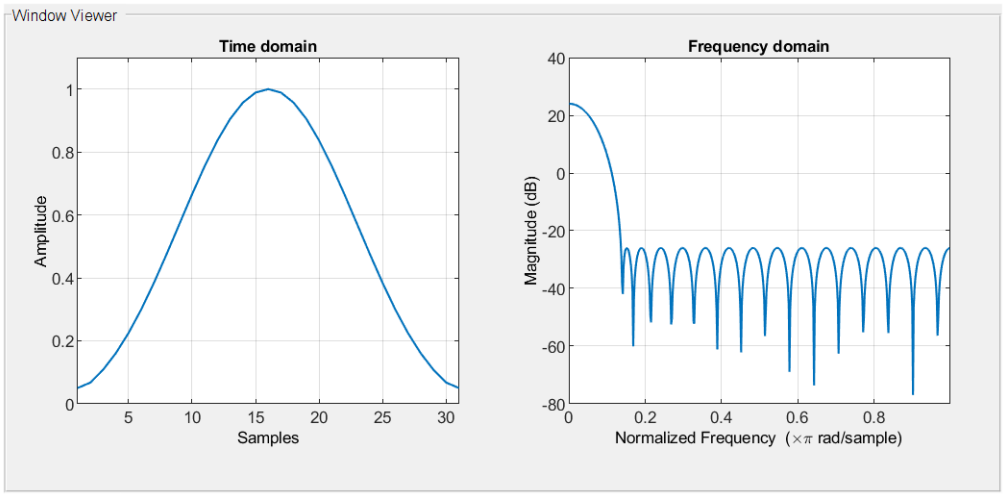

Figure 11: Time Domain Plot and Amplitude Spectrum of Chebyshev Window

A Gaussian window was designed with N = 31. Moreover, the standard deviation was equal to 3 for the design. The time domain plot and the amplitude spectrum can be seen in the same figure below:

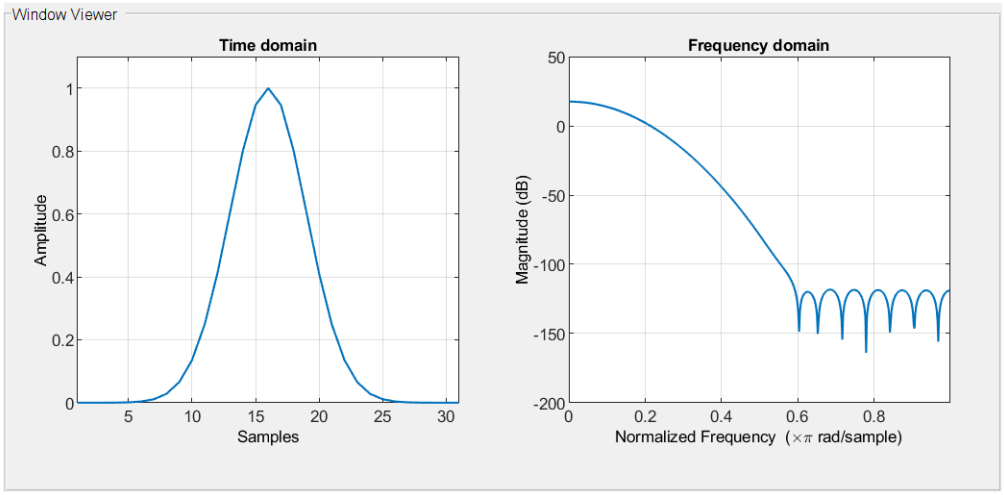

Figure 12: Time Domain Plot and Amplitude Spectrum of Gaussian Window

A Kaiser window was designed with N = 31. Moreover, β was set to 10 for the design. The time domain plot and the amplitude spectrum can be seen in the same figure below:

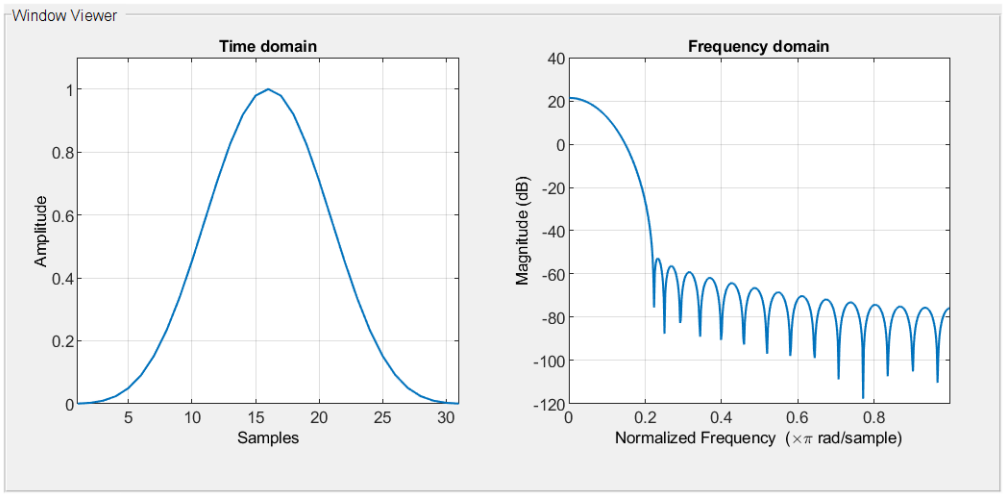

Figure 13: Time Domain Plot and Amplitude Spectrum of Kaiser Window

Four 30th-order low-pass filters in were designed in MATLAB with a cut-off frequency at 160 Hz using the window functions defined above. Poles and zeros plots of the filters can be found in the figures below:

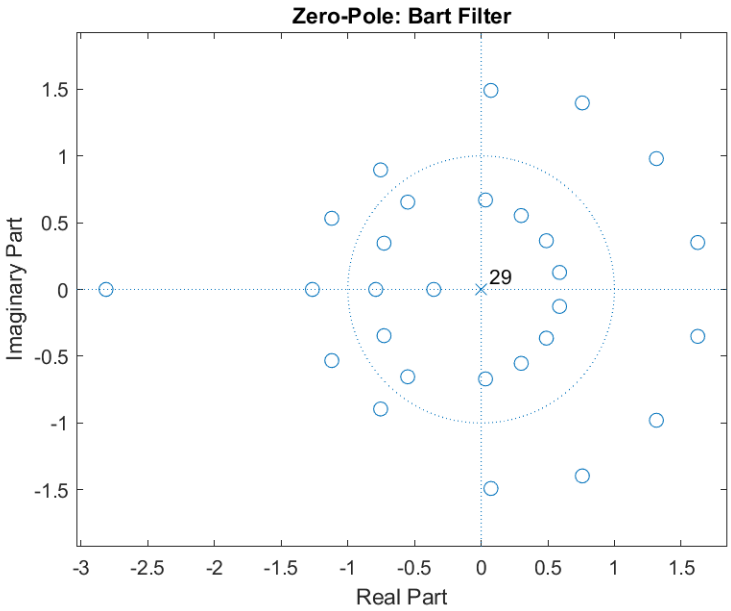

Figure 14: Zero-Poles Plot for Bartlett Filter

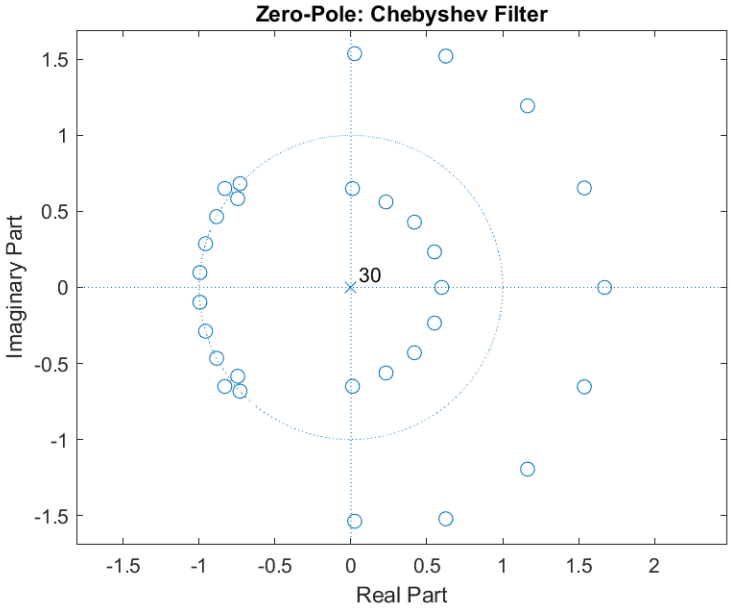

Figure 15: Zero-Poles Plot for Chebyshev Filter

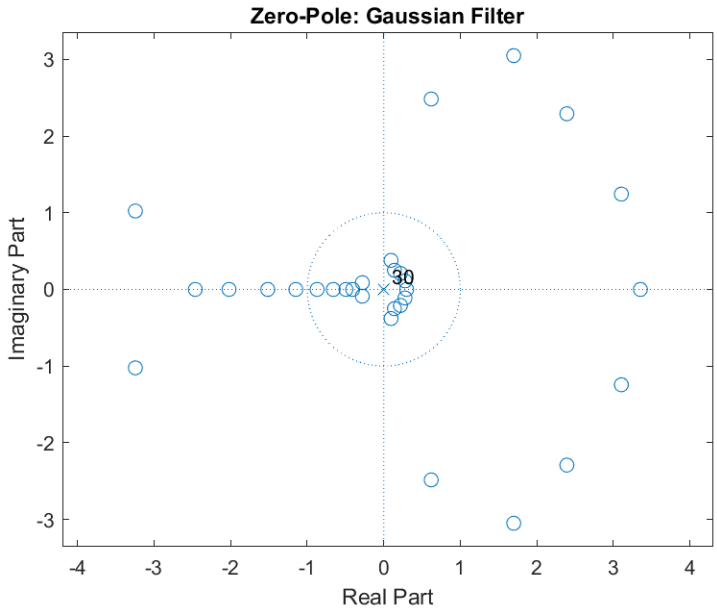

Figure 16: Zero-Poles Plot for Gaussian Filter

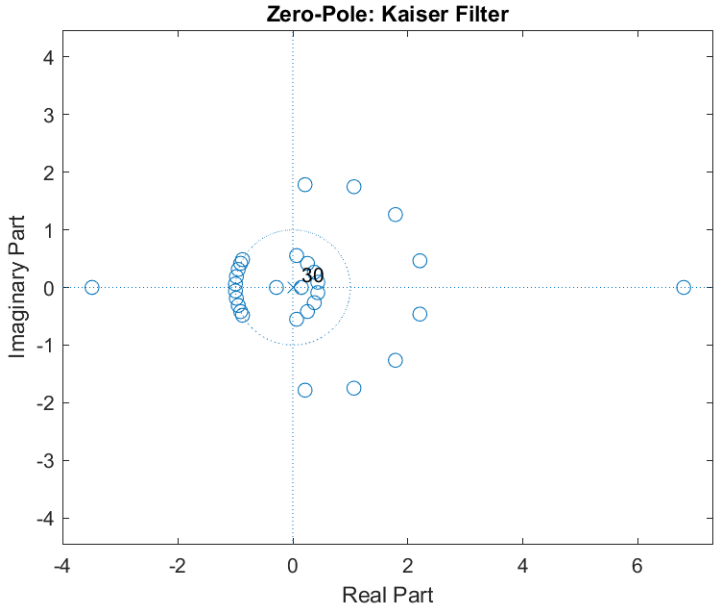

Figure 17: Zero-Poles Plot for Kaiser Filter

The frequency responses of the filters were then determined using MATLAB for each of the four 30th order low-pass filters. They can be found in the figures below:

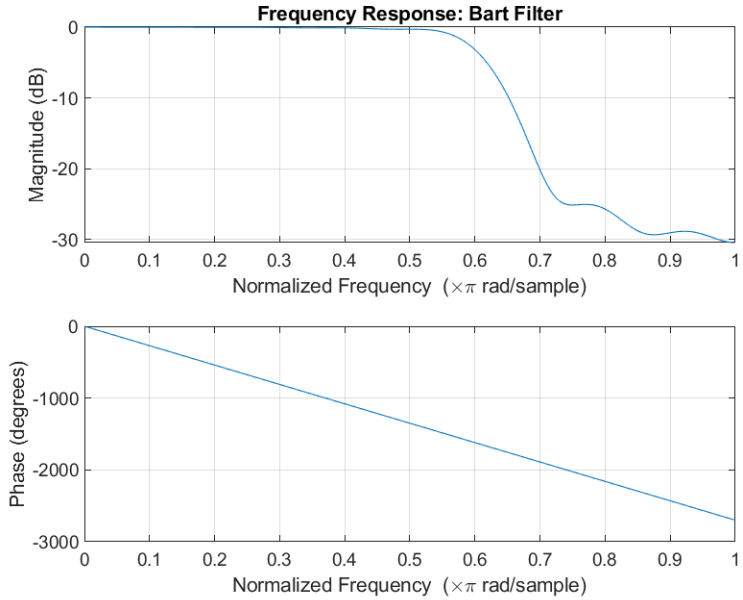

Figure 18: Frequency Response of Bartlett Filter

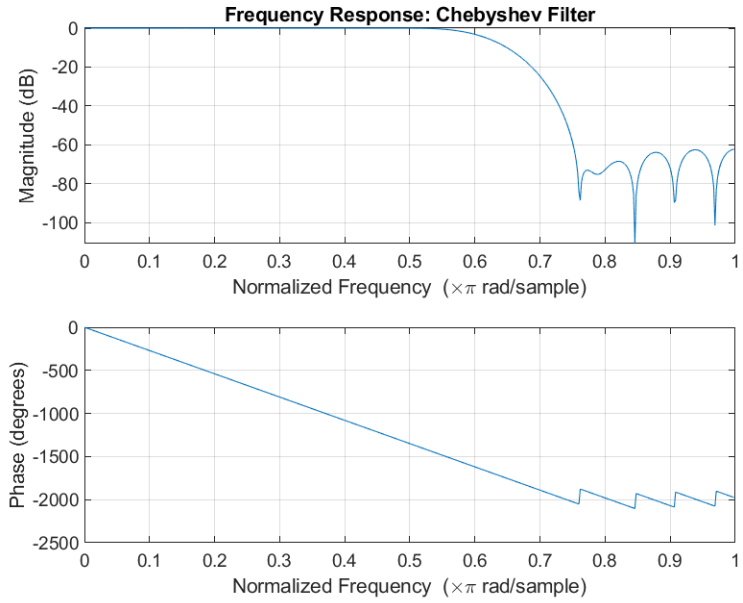

Figure 19: Frequency Response of Chebyshev Filter

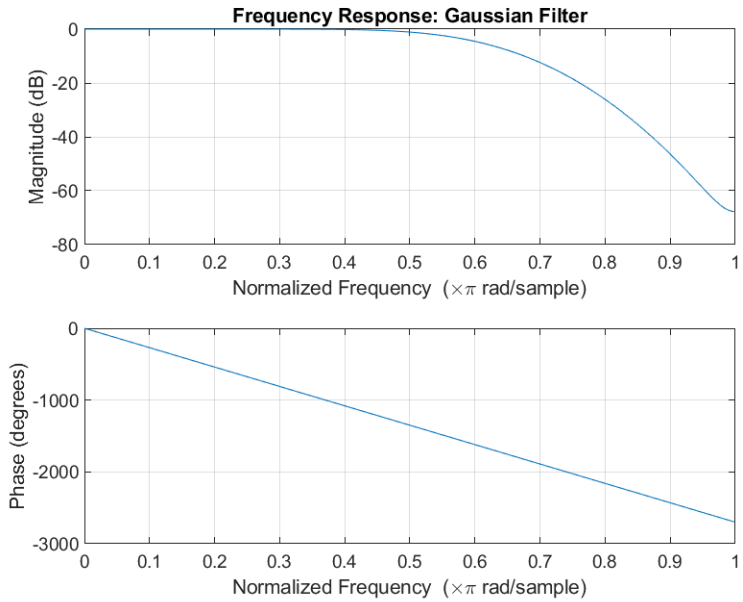

Figure 20: Frequency Response of Gaussian Filter

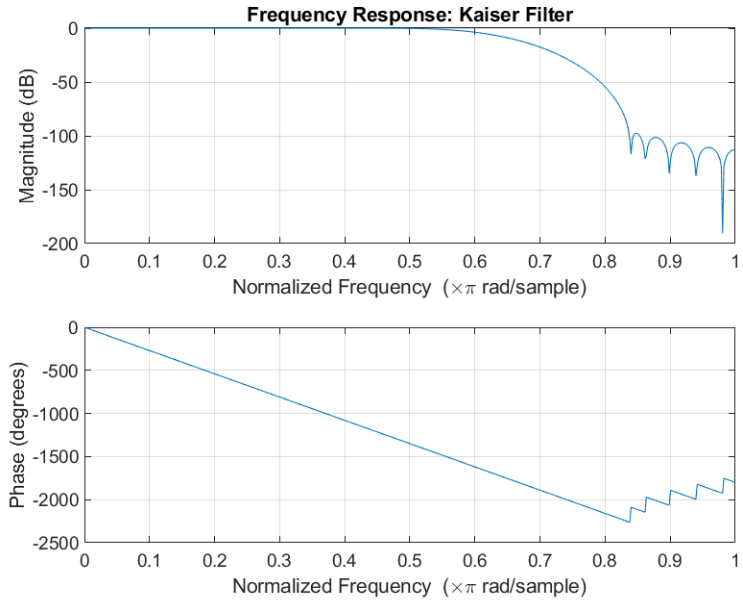

Figure 21: Frequency Response of Kaiser Filter

A plot of the amplitude spectrum of the signal $x_a(t)$ prepared, where:

$$x_a(t) = cos(20\pi t) + sin(240\pi t) + cos(360\pi t) + sin(420\pi t)$$

And the signal is sampled at $f_s = 512$ Hz for 1 second.

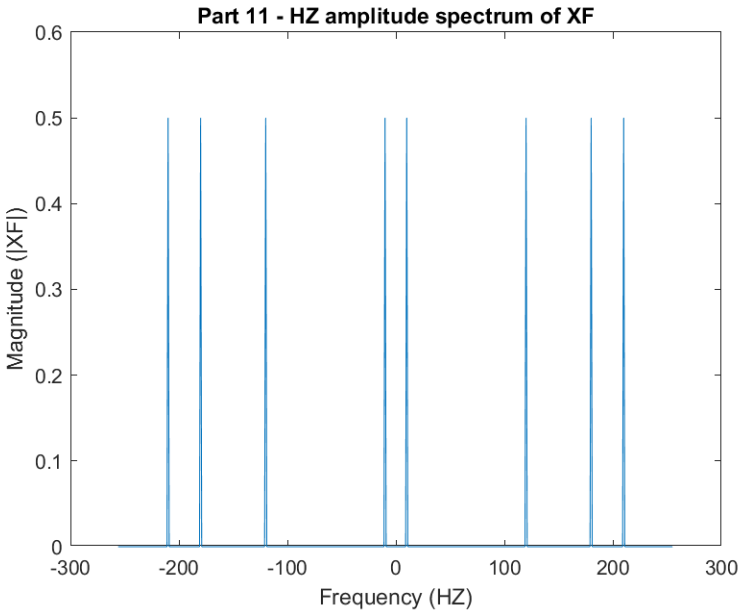

Figure 22: Amplitude Spectrum of the Signal $x_a(t)$

This signal was then filtered by each of the four window filters designed above. The filtered signal was captured along with the amplitude spectrum.

First, the Bartlett filter was used. The filtered signal plot and amplitude spectrum can be seen in the next two figures:

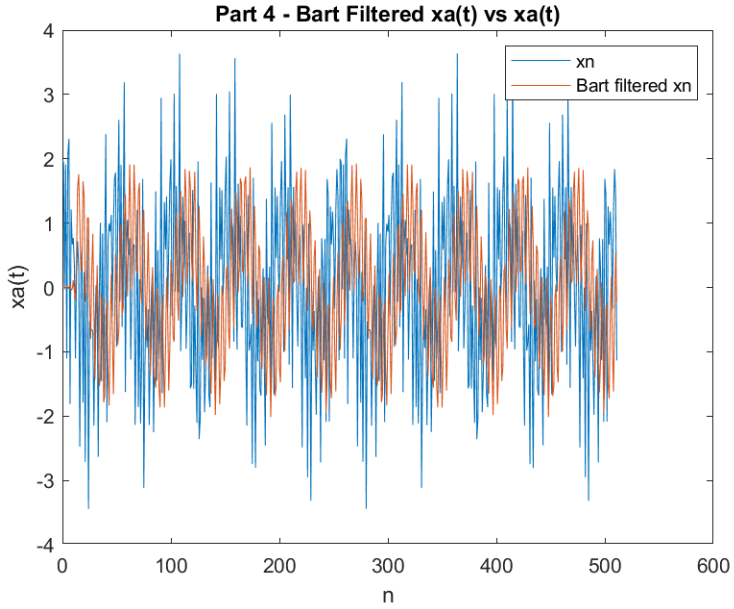

Figure 23: Signal Filtered with Bartlett Filter

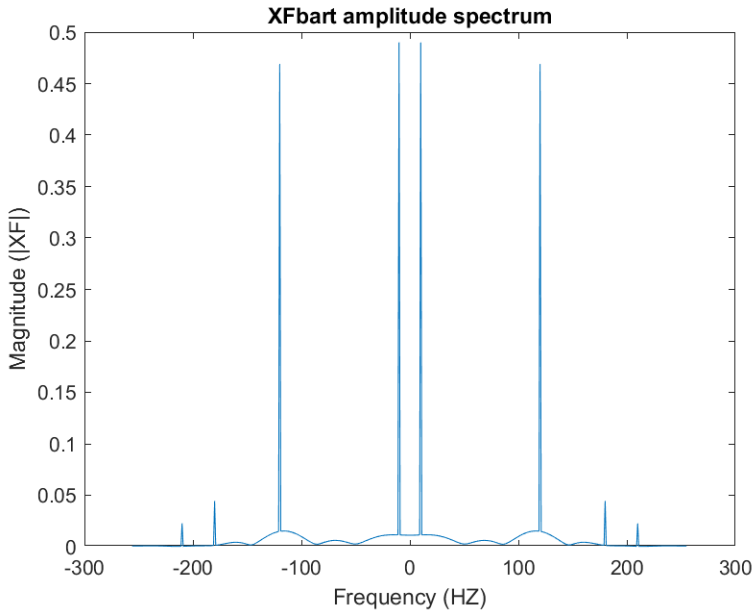

Figure 24: Amplitude Spectrum of Bartlett Filtered Signal

Second, the Chebyshev filter was used. The filtered signal plot and amplitude spectrum can be seen in the next two figures:

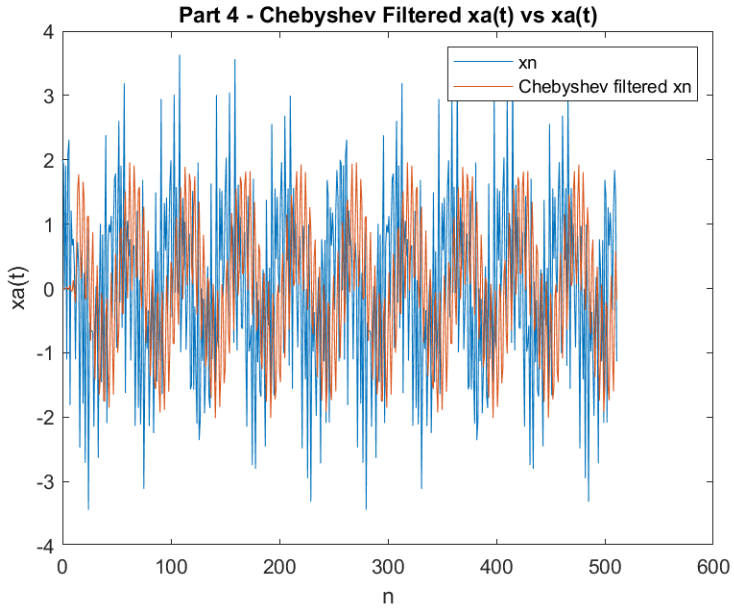

Figure 25: Signal Filtered with Chebyshev Filter

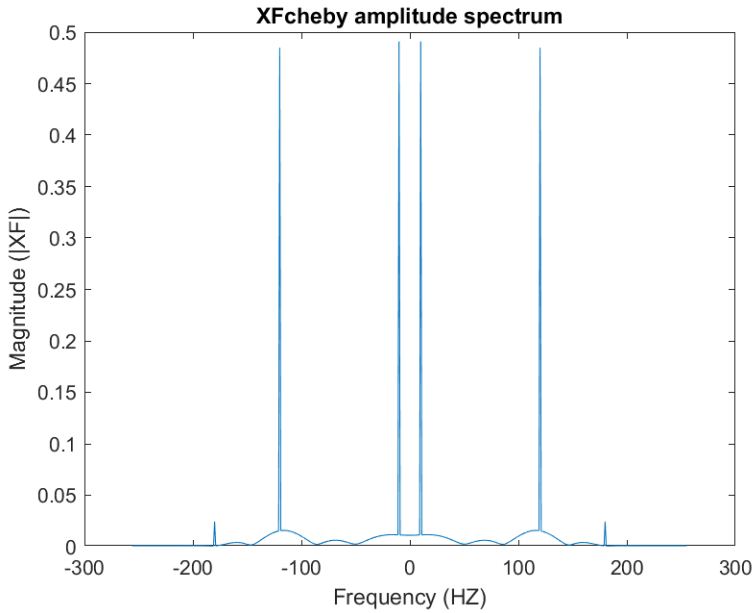

Figure 26: Amplitude Spectrum of Chebyshev Filtered Signal

Third, the Gaussian filter was used. The filtered signal plot and amplitude spectrum can be seen in the next two figures:

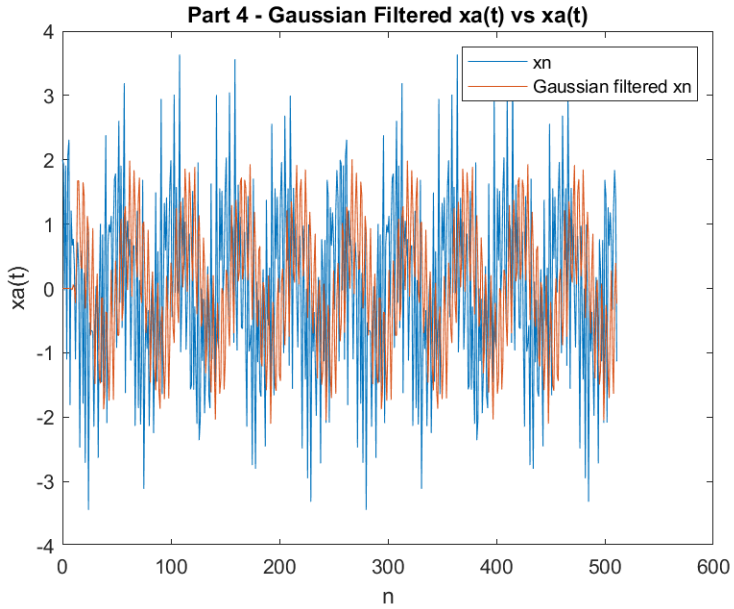

Figure 27: Signal Filtered with Gaussian Filter

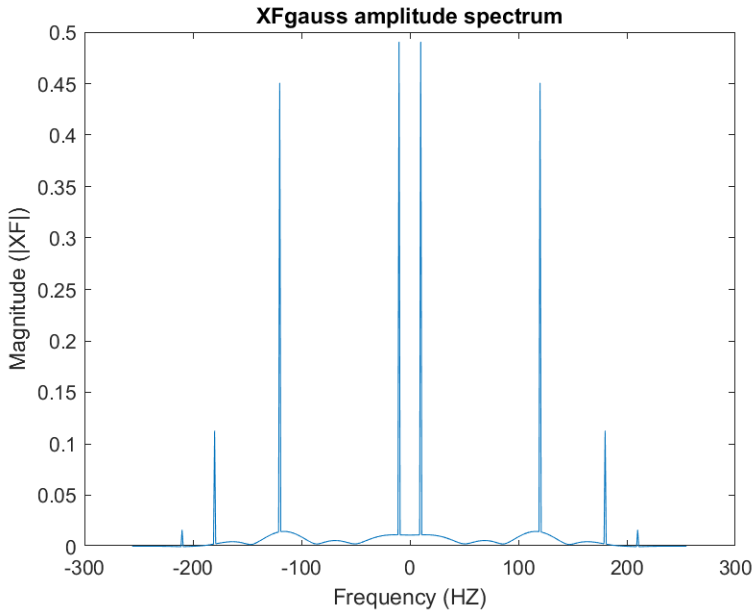

Figure 28: Amplitude Spectrum of Gaussian Filtered Signal

Finally, the Kaiser filter was used. The filtered signal plot and amplitude spectrum can be seen in the next two figures:

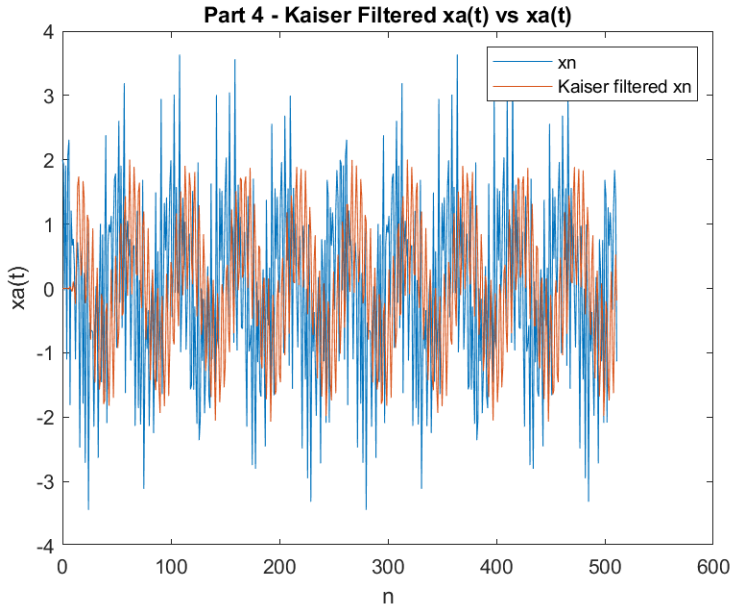

Figure 29: Signal Filtered with Kaiser Filter

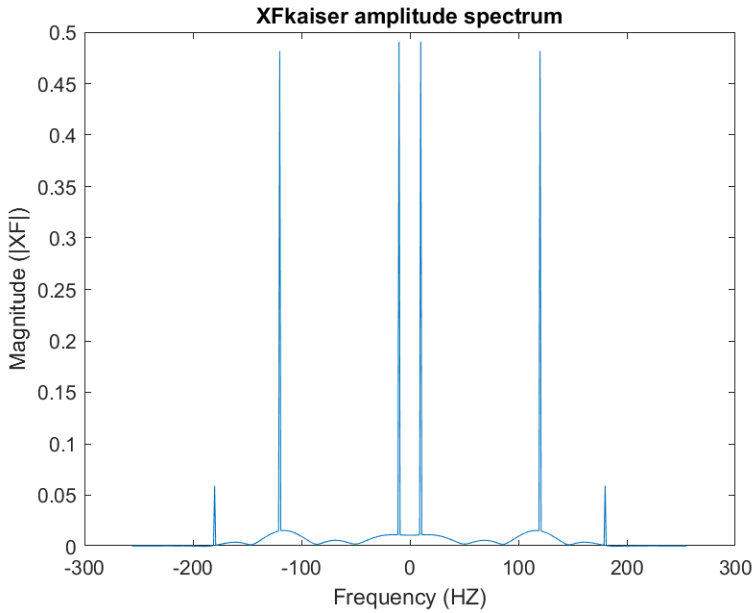

Figure 30: Amplitude Spectrum of Kaiser Filtered Signal

Microcontroller

Using a sinusoidal signal with peak to peak voltage of 2V and DC offset of 1.65V, sampled at 256Hz, the following filter was implemented in arduino:

$$y[n] = 0.122760378135474x(n) + 0.479525558184156x(n-1) + 0.479525558184156x(n-2) + 0.122760378135474x(n-3)$$

Varying the frequency of the input signal over the range $0 \leq f \leq Fs/2$, the amplitude of the output signal at each frequency was recorded and the magnitude frequency response was plotted below:

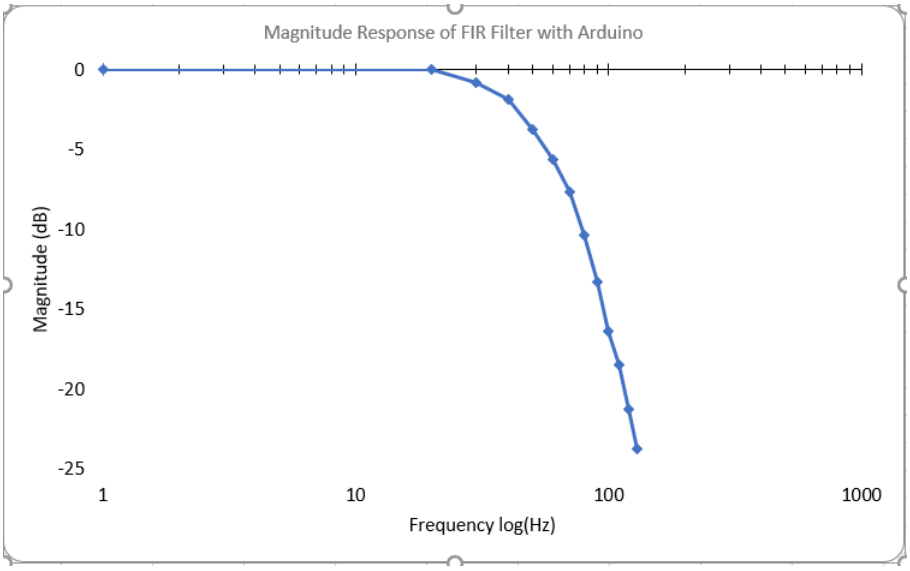

Figure 31: Plot of Measured Values Using Arduino Filter

The theoretical magnitude response was determined in MATLAB and the plot of this frequency can be seen in the figure below:

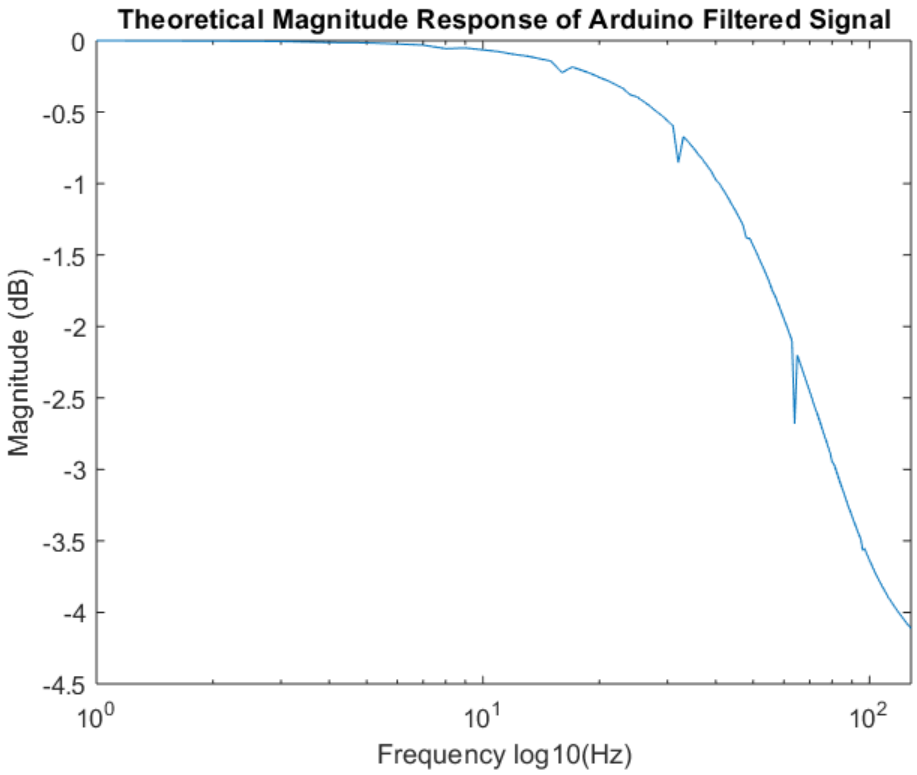

Figure 32: Plot of Theoretical Magnitude Response in MATLAB

Discussion

IIR Filters

As seen in Fig. 1, the peaks of the first signal are at the expected frequencies of 5 Hz, 120 Hz, and 200 Hz, which directly corresponds to the signal definition. This verified that the signal was generated correctly.

Fig. 2 shows the poles and zeros of the sixth-order Butterworth filter design. As seen in the plot, all of the zeros lie at π which makes sense because this is a low pass filter and therefore the zeros lie at high frequency.

In Fig. 3, the magnitude and phase response of the Butterworth filter is shown. From the amplitude response, it is easy to observe that this is a low-pass filter. The cut-off frequency is approximately at 0.55π rad/sample which corresponds to the 150 Hz designed. Also the phase of the filter is approximately linear in the pass region as expected.

The filtered signal in Fig. 4 and its corresponding amplitude spectrum in Fig. 5 show that the filter only filters out the highest frequency at 200 Hz. Since the cutoff frequency was designed to be at 150 Hz, the two lower frequencies remained. This is reflected in the quality of the filtered signal in Fig. 4 where the filtered signal appears to be very close to the original since only the highest frequency was filtered out.

Next, the Chebyshev sixth-order filter was designed and poles and zeros plot can be seen in Fig. 6. As seen in the plot the zeroes are again at high frequencies making this low pass. The poles are in the right half of the plane which reflects having a lower cutoff frequency than in the Butterworth case where the poles were in the left half of the plane.

The frequency response of the filter has an approximate cutoff frequency of 0.4π rad/sample which corresponds to the 100 Hz frequency in time of which it was designed. Again the phase of the response is approximately linear in the passband which is expected.

As seen in Figs. 8 and 9 this filter does a much better job of filtering out higher frequencies. A single sine wave is approximately visible in the filtered output. This is reflected in the amplitude spectrum as the

lowest frequency is most prominent while the highest frequency is completely filtered out and the middle frequency at 120 Hz has been substantially attenuated. Overall, the Chebyshev filter does a better job as a low pass filter than the Butterworth due to the lower cutoff frequency.

FIR Filters

Figs. 10 - 13 show the four window functions designed for the FIR section of the lab. Each plot contains the time domain plot and the amplitude spectrum in the same figure. As seen in the figures, all four have relatively similar time domain plots. The only differences are in the width of the band which is reflected in the amplitude spectra because a compression in time corresponds to ane expansion in frequency. This can be seen best in Fig. 12 which was designed with a standard deviation of 3.

Also important to notice are the degrees of maximum and minimum attenuation in the amplitude spectrum plots. All windows perform similarly in the passband, while the differences can be seen most in the stopband and slope between passband and stopband. These are expected to effect the filtered responses in the next part of this lab and will be discussed below.

Next, a 30th-order low-pass filter was designed with each window function in MATLAB with a cutoff frequency of 160 Hz. The poles and zeros of each system can be seen in Figs. 14 - 17. The placement of the poles and zeros for each filter reflect the amplitude spectra seen in Figs. 10 - 13 and explain why windows like the Gaussian filter have a shallower slope in the cutoff frequency range between passband and stopband for example.

Figs. 18 - 21 provide the frequency response of each filter showing both the magnitude and phase spectra for each filter. As expected, all filters have linear phase response in the pass region. The only differences in phase response between the filters is in the stop region where some remain linear while others are not. This is a reflection of the magnitude response in the stop band regions. The Gaussian and Bartlett filter remain mostly smooth throughout the stopband which is reflected in a mostly linear phase. However the Chebyshev and Kaiser have multiple lobes in the stopband which corresponds to the nonlinear phase seen in the plots.

With respect to magnitude, all filters have similar pass band behavior. The difference lies in the period from the cutoff frequency throughout the stopband as was seen in the window functions in Figs. 10 - 13.

Fig. 22 shows the amplitude spectrum of the sinusoidal signal defined as $x_a(t)$. As expected frequencies are at 10 Hz, 120 Hz, 180 Hz, and 210 Hz.

Next, each filter designed above was used to filter this signal in Fig. 22. Figs. 23 - 29 show the filtered signals. As seen in those plots, the filtered response looks fairly similar for each case. This is becase with a cutoff frequency of 160 Hz, all four filters do a good job of keeping the smallest two frequencies and cutting out the highest two which results in similar responses. The differences can really only be seen in the corresponding amplitude spectrum plots of Figs. 24 - 30. As seen in these plots, all filters keep close to full magnitude of the lowest two frequencies but are somewhat different with attenuating the highest two frequencies. The Chebyshev does the best job in attenuating the highest two frequencies followed by the Bartlett filter. Then both the Kaiser and Gauss do a good job of filtering the highest frequency but still maintain some of the second highest, in which Kaiser attenuates more than the Gauss. Overall, all four filters result in similar output signals which there are minute details in the filtering of the highest frequencies due to the nature of the cutoff region and stopband.

Microcontroller

Figs. 31 and 32 show the results of the microcontroller experiment and compare the measured frequency response with the theoretical response determined via MATLAB. As seen in the two plots, the passband of each is fairly similar where the cutoff frequency appears to be somewhere around 10 Hz in both the measured and theoretical cases which is expected. The difference lies in the amount of attenuation that results after the cutoff frequency. As seen in the plots, the measured case attenuates much more than in the theoretical case which is most likely due to the effect of measuring actual quantities rather than computer generated theoretical ones.

Conclusions

The lab was largely successful in helping us develop a better understanding of filters mathematically as well as their practical application in discrete-time signals and systems. Noteworthy topics of exploration included amplitude spectra analysis, Butterworth filters, Chebyshev filters, Gaussian filters, Kaiser filters, windowing functions, and the practical application of filtering through software using Arduino. All results were expected - amplitude spectra of filtered results reflected expected values and the Arduino measured frequency response properly reflected a low-pass filter in practice.

The major conclusion of this lab was that depending on the design, some types of filters work better than others in different regards, whether that be in the magnitude of that passband or stopband or in the phase spectrum, etc. It all depends on the requirements and specs of the design. In this lab, we were exposed to multiple common filter and window types which helps us build an intuition with respect to choosing effective filter designs. All in all, this lab provided an excellent topic to close out the semester.

References

[1] Dr. Abdelhakim, Class Professor

[2] Lecture Part IV: Filters

[3] Kechen Shu, Lab TA