ELEC 240
Lab 6 - Digital Signal Processing

Test performed: October 16, 2018

Report submitted: October 23, 2018

Andrew Pham, Aneel Damaraju

1 Objective

The first objective of this lab was to understand the key concepts behind sampling, through use of varying the sampling frequency and input frequency of a given signal. There was also some inspection into another important factor in digital signal processing, quantization, and seeing how effective a quantized signal recreates an input. The second objective was more skills based, focusing on learning how to upload vocal signals to MATLAB through LabView. This included understanding the specgram and sound functions in MATLAB, as well qualitatively understanding some key features of spectral plots. For the final objective, we focused on learning about various types of filters and their frequency and unit sample responses.

2 Materials

- Virtual Bench (Software, Oscilloscope, Function Generator, DC Power Supply)
- LabView
- BNC Male to Clips cord
- Oscilloscope Probe
- Breadboard (with setup from Lab 4
- 2 10 cm length wires (with 6 mm stripped on each end)
- Digital Multimeter
- 100 $k\Omega$ resistor
- 2 100 Ω resistors
- 1 $k\Omega$ resistor
- 2 LM 741 Op-Amps
- Telephone handset
- Dynamic microphone
- Smartphone (or some device to play audio from a speaker)
- MATLAB
- Lab PC

3 Test Description

In Experiment 6.1, Part A, we connected the function generator to a LabView spectrum analyzer, and varied the input function. We created a sine wave and varied the frequency of it as well as the sampling frequency and checked what properties of the sampled wave would change. We focused on the fundamental frequency and how the output looked around the Nyquist frequency. For Part B, we added a quantizer to our spectrum analyzer and compared the quantized signal to the original signal. From there, we changed the number of quantized bits and qualitatively viewed the output.

In Experiment 6.2, Part A, we created two Op-Amp circuits, one connecting the Lab PC sound card to the handset and one connecting the dynamic microphone to the DAQ cable. We then recorded two sounds, one being general human voice and the other being a single pitch whistle. For Part B, we simply loaded the signals into MATLAB and made sure that MATLAB could effectively recreate the now quantized sounds. For Part C, we took the signals and then plotted the magnitude of the signal as well as just a chunk of the signal.

In Experiment 6.3, Part A, we made changes to the signals amplitude and frequency and listened to the changes. We then added the two signals. For Part B, applied simple filters to the voice output, including cariable length boxcar filters and infinite impulse response filters. We found the unit sample, frequency and sound responses for each filter. For Part C, the same tests were done, but for the more complicated Butterworth and Finite Impulse responses.

3.1 Pre-Lab Calculations and Schematics

The Pre-Lab for this lab only included a general understanding of LabView and Virtual Bench, which was done in previous labs. The other information in this lab, is expected to be learned through examples found in the lab.

4 Results and Discussion

In Experiment 6.1, Part A, we explored sampling a signal in the time domain. We began by connecting the function generator to pin 46, which was the input to the LabView software and generated a 5 V_{pp}, 300 Hz sine wave. We then sampled the signal at a rate of 10000 samples per second with average set to RMS averaging to achieve the following waveform graph in Fig 1 below.

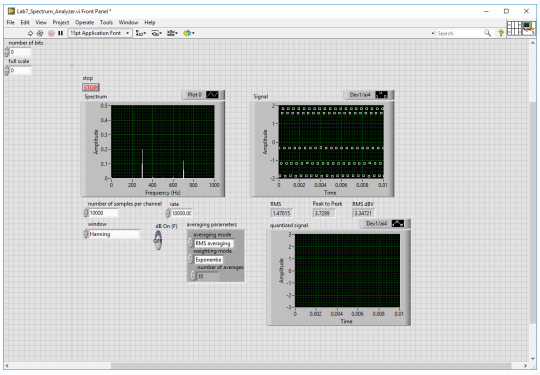

Figure 1: Sample of 5 V_{pp}, 300 Hz Sine Wave

We then began to increase the frequency of the sine wave outputted by the function generator. When the output frequency was half of the sampling frequency, we observed that the samples alternated between positive and negative values. We began to observe aliasing past 5kHz, and the input frequency stopped matching the displayed frequency in Labview. For example, in Fig 2 and Fig 3 below, we can observe aliasing when the input frequency is 11 kHz and 13 kHz, respectively.

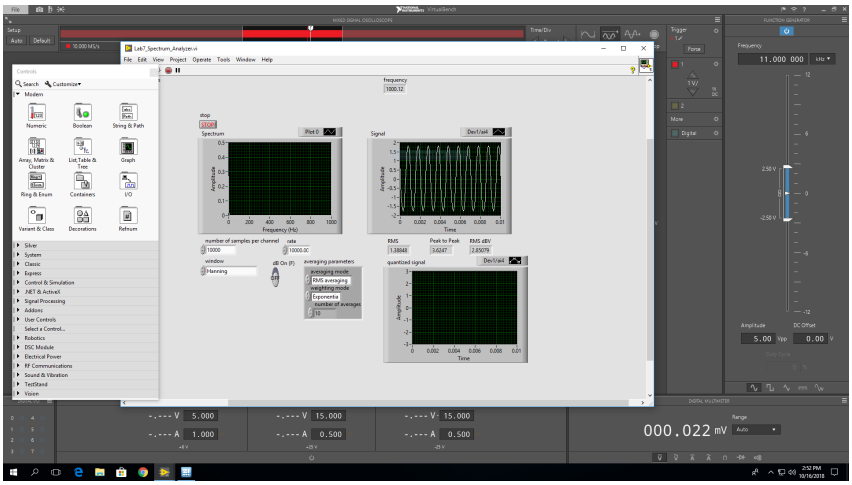

Figure 2: 10 kHz Sample of 11kHz Sine Wave

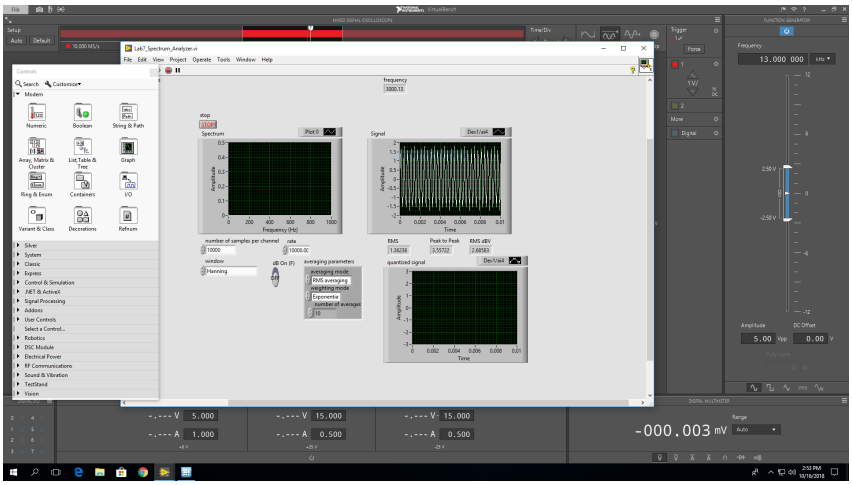

Figure 3: 10 kHz Sample of 13kHz Sine Wave

In the above figures, the output waveform is the alias of the input frequency, which occurs whenever twice the input frequency exceeds the sampling frequency. This concept is directly related to the Nyquist criterion, which states that no aliasing will occur as long as the sampling frequency is greater than or equal to twice the input frequency. Whenever the input frequency exceeds the Nyquist frequency, the input frequency is "folded over" into a lower. Thus, the 11 kHz signal was folded over the 10 kHz sampling frequency into a 1 kHz wave, and the 13 kHz signal was folded over the 10 kHz sampling frequency into a 3 kHz wave, as seen in the figures above.

We then switched the function generator to produce a square wave and a sine wave. On frequencies that were integer divisors of the sampling frequency, the wave appeared to be sampled correctly below 5 kHz. However, on other frequencies such as 1.3 kHz, both the triangle and square wave appeared distorted. We attribute this distortion to the fact that neither square waves nor triangle waves are bandlimited, which means that the Nyquist Sampling theorem does not apply to them.

In Part B of Experiment 6.1, we explored amplitude quantization of discretely sampled signals. We began by producing a 1 kHz sine wave and feeding it into the amplitude quantizer that we constructed in Labview. As we increased the number of bits used by the quantizer, we noticed that the sampled sine wave became much more fluid and discerniblee. Likewise, as we decreased the number of bits, the sampled sine waveform became choppier and more step-like. We then switched the power spectrum graph to accept the discretely

quantized sine wave as input rather than the original analog sine wave. We noticed that the amplitude of the power spectrum graph increased by 20%.

In Experiment 6.2, Part A, we digitally recorded two signals. We began by constructing the two circuits below in Fig 4.

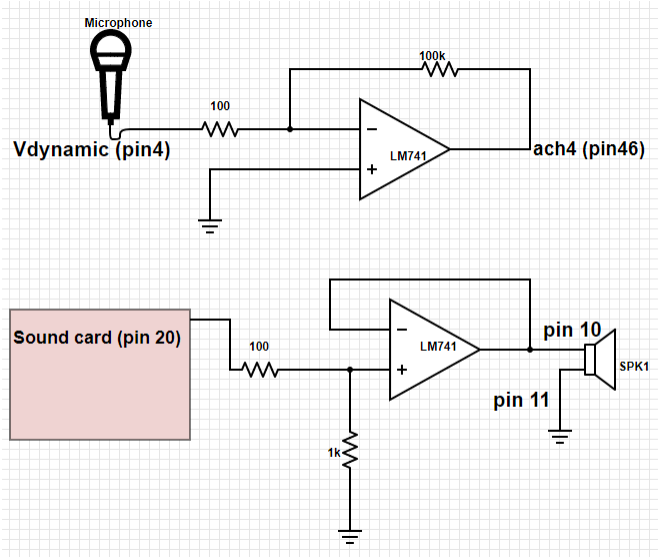

Figure 4: Recording Circuit (Top) and Playback Circuit (Bottom)

We then recorded one signal where I spoke a phrase and another signal where I whistled into the microphone. We saved the two signals into a .xlsx file. In Part B, we loaded these signals into MATLAB and performed digital signal processing on them. When we ran the sound command in MATLAB with $F_s = 5kHz$ rather than 10 kHz, we noticed that the pitch of the recording was one octave lower. This is because inputting a sampling frequency $1/2$ of the original effectively tells MATLAB that the signal was recorded at half the speed it actually was recorded it, which effectively halves all the frequencies (hence the lower overall pitch).

In Part C, we performed spectral analysis on both the sound signals. We used a length $n = 256$ discrete fourier transform. For a sampling rate of 10kHz, the time resolution was $n/f_s = 256/10kHz = 0.0256$ and the frequency resolution was $f_s/n = 10kHz/256 = 39.0625$. We then tried transforms of length 128 and 512. The length 128 transform halved the number of samples, which decreased the resolution. The time resolution of $n = 128$ was $n/f_2 = 128/10kHz = 0.0128$ and the frequency resolution was $f_s/n = 10kHz/128 = 78.125$. The difference in sound quality was noticeable. The length 512 transform doubled the number of samples and thus the resolution, though the difference in the length 256 and 512 transforms was not as discernible. The time resolution of $n = 512$ was $n/f_2 = 512/10kHz = 0.0512$ and the frequency resolution was $f_s/n = 10kHz/512 = 19.53$.

We then extracted a length 256 chunk from the speech signal at the 10,000 index of n. We then computed the fourier transform of this chunk using the fft function in MATLAB. We then truncated the signal and plotted it on a logarithmic scale on the y-axis to achieve the following Fig 5

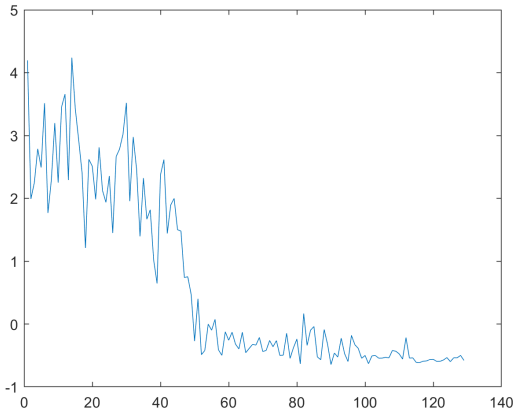

Figure 5: Discrete Fourier Transform of Truncated Speech Signal

We then created a spectrogram of the musical signal, depicted below in Fig 6

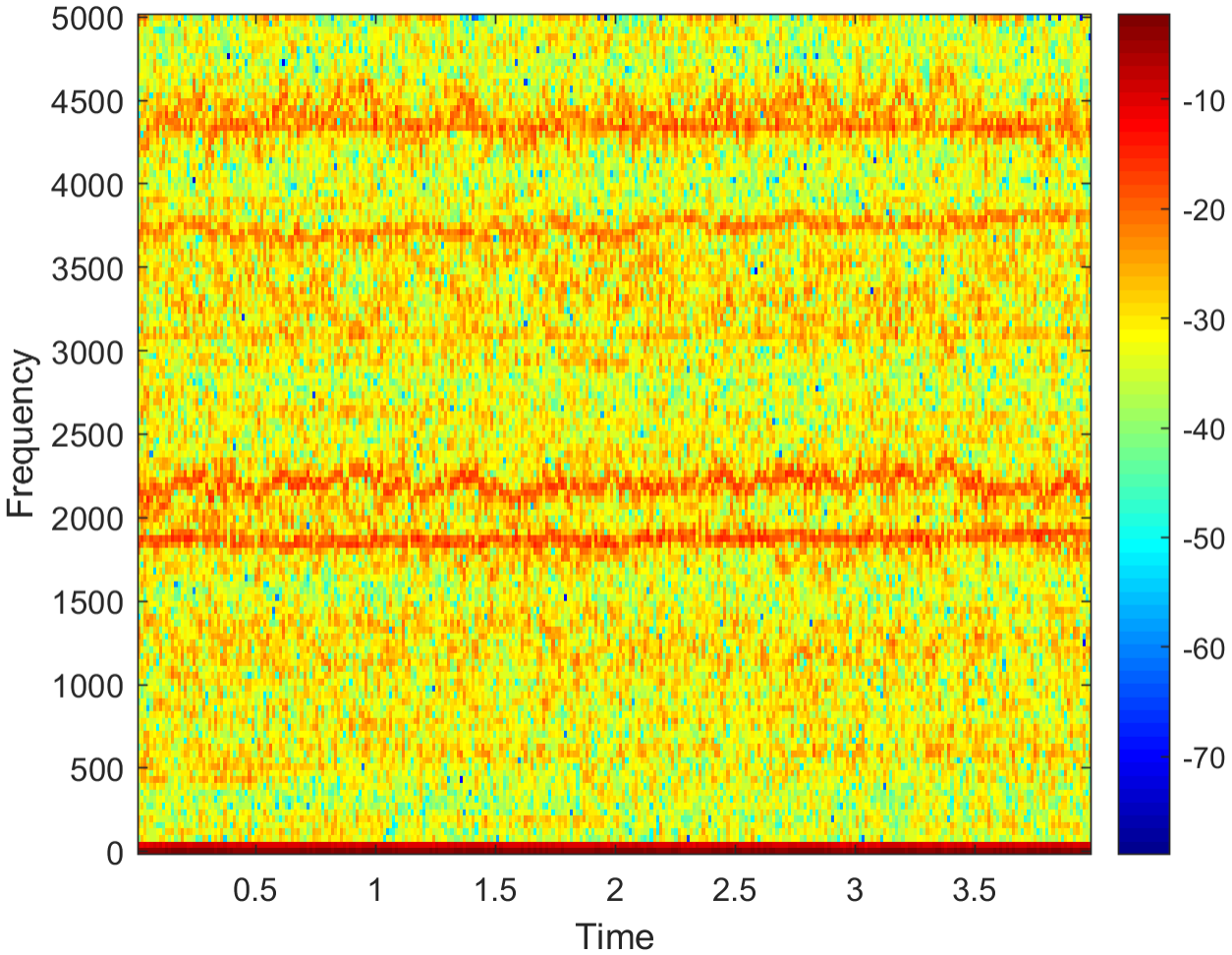

Figure 6: Spectrogram of Whistle

The four lines running horizontally at 1.8, 2.2, 3.6, and 4.4 kHz across the figure depict the four sine waves of these frequencies that constitute the whistle. Upon closer observation, we can see that the whistle is actually comprised of two notes at two different harmonics. The base frequencies are 1.8 and 2.2 kHz while the 3.6 and 4.4 kHz frequencies is exactly twice the base frequencies.

In Experiment 6.3, Part A, we scaled the signals by constant values, which simply changed the volume of the signal. We also added the two signals together and observed them in both the audible and visual domains. When the signal sum was played back, it simply sounded like the two signals were played simultaneously. When we viewed the signal sum on the spectrogram, it appears like the magnitudes in the individual spectrograms were added together so that the spectrogram of the signal sums are just overlayed, as seen below in Fig 7

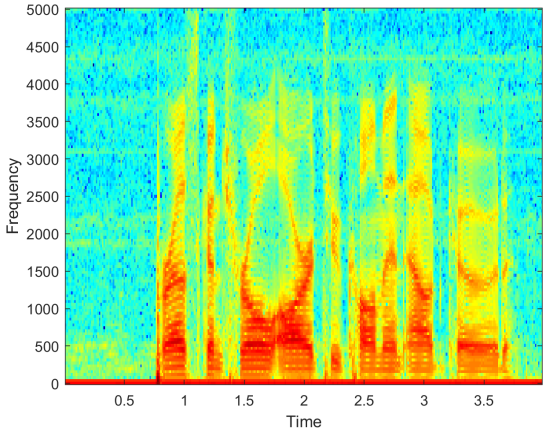

Figure 7: Spectrogram of Sum of Signal 1 and 2

We then took the dot product of the two signals. When we observed the product signal when played back, we could still discern the individual signals; however, the signal had significantly more noise than the individual signals' sum. In the spectrogram of the signals' dot product, we could also make out the individual signals but with significantly more difficulty. The individual horizontal lines from the whistle remained prominent. However, the spikes from the spoken signal seemed to be more distorted and the red portions appeared to spread out into the higher frequencies, as seen below in Fig

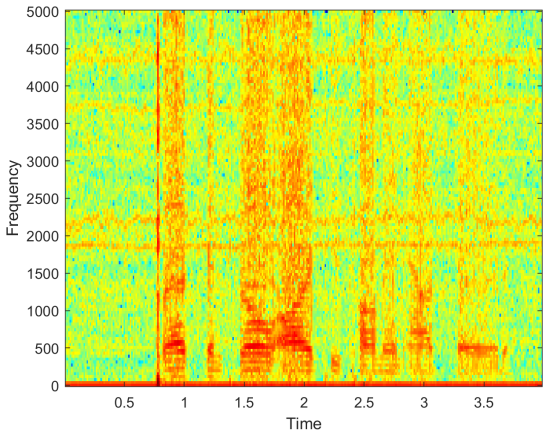

Figure 8: Spectrogram of Dot Product of Signal 1 and 2

In Part B, we first passed the signals through a Box Car filter of length 5. We then computed the

frequency response of the filter in MATLAB using the freqz function to yield the following response graph depicted below in Fig 9

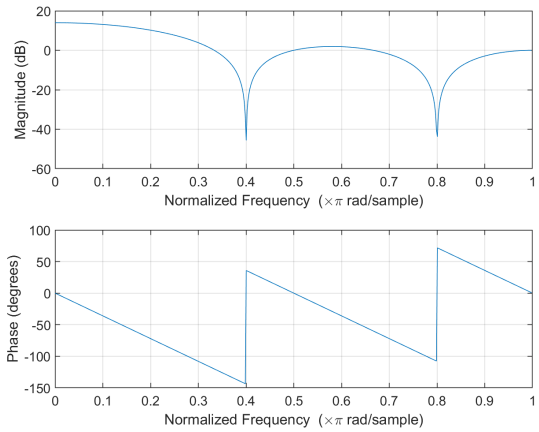

Figure 9: Frequency Response of Length 5 Boxcar Filter

We then applied this filter to our first speech signal and observed the following spectrogram in Fig 10

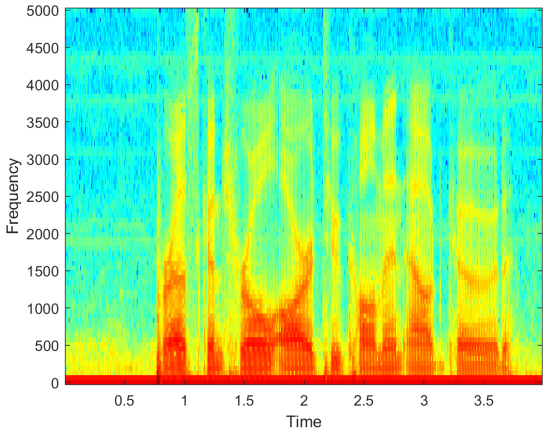

Figure 10: Spectrogram of Length 5 Boxcar Filter Applied to Speech Signal

By referencing the spectrogram of the original signal, we can see that the length 5 boxcar filter acts somehwat as a low-pass by attenuating the higher frequencies. The low-pass nature was more clear when we played back the sound; the higher frequencies were filtered out, resulting in a more muffled signal whose speech was lower on average and a little harder to make out clearly. The blue areas of the spectrogram correspond to the filtered-out frequencies. After repeating the same process for a length 10 boxcar filter, we concluded that the relationship between filter length of frequency response was that the longer filter was "stronger" by filtering out more of the background noise; however, it would also filter out more of the meaningful changes in frequency. We then implemented an infinite impulse response filter that acted as a low-pass filter by having one positive and one negative "a" coefficient. The low-pass nature was also evident in the playback because the voice sounded lower and there was less high-frequency noise. We also noted that the speech playback sounded clearer than the boxcar-filtered signal. We then changed the negative sign of one of the "a" coefficients to positive and noticed that the IIR filter became a high-pass filter. This could be heard in the audio playback because the voice was much clearer and the low-frequencies of the voice as well as the low-frequency background noise were gone.

In Part C of Experiment 7.3, we implemented a 5th order lowpass Butterworth filter. We found that this filter was much more ideal than the length-5 boxcar filter when we played the signal back; the signal sounded much less distorted. We then created a 50th order Butterworth filter and observed the unit-sample response of the filter depicted below in Fig 11

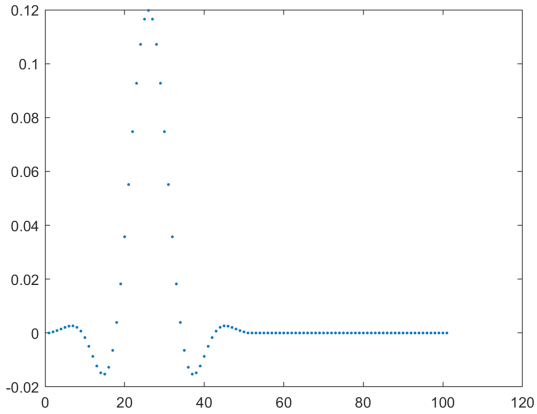

Figure 11: Unit Sample Response of 50th Order Butterworth Filter

The unit sample response is characteristic of band-pass filter; only a finite range of frequencies in the middle are allowed through. We then passed the signal through this bandpass filter and observed that the filtered signal sounded much more muffled than the high-pass filtered signal; this was because the bandpass was filtering out both high and low frequencies, and we only heard the midrange speech signals, which are only a fraction of the range of speech frequencies.

5 References

- https://www.ece.rice.edu/ dpr2/elec240/lab6/

6 Conclusion

In 6.1 we experimented with general ideas of sampling and quantization through the use of LabView and various ways that LabView was able to provide tools to assist in the use of signal reconstruction. We also looked at the importance of the Nyquist frequency and aliasing caused by too low of a sampling rate. Viewing the power spectrum of the discretely quantized wave was also done and was compared to its analog counterpart.

In 6.2, we started by understanding how to use two separate Op-Amp circuits as well as LabView to create a recorded vocal message. This message was saved as a series of output voltages and converted as a csv file to MATLAB. Through MATLAB, it was confirmed that this signal could be recovered, confirming that the circuits created in 6.2 actually worked as expected. When changing the sampling frequency in MATLAB, we could change the pitch and length of the output tone. Finally, we observed the spectrum of the signal, both through the spectrogram and the FFT of the signal. It was very clear from the recorded outputs which one of the spectrograms was a speech signal and which was a solid whistle due to the number of frequencies with strong outputs for each.

In 6.3, we explored various signal processing and filtering techniques. Simply varying the output of the sound changes the amplitude of the sound. Adding the digital representations of the sound adds the output sounds, but multiplying them scrambles the output, as expected. We also looked into boxcar filters, and how they act as low pass filter of sorts. We then looked at both IIR and FIR filters and noticed how while

IIR can be either highpass or lowpass, FIR can also function as a bandpass filter. These different filters, along with the Butterworth filter, were all viewed simply using the unit sample response and the frequency response to explicitly learn these methods and understand their general transfer functions.

7 Errors

One error we encountered was in the musical whistle signal during the recording process. The whistle was not a clean note; there were periods where the whistler was only blowing air into the mic, which likely resulted in high amounts of noise because of the large volume of air flowing over the microphone. This resulted in our data for this signal having a lot of noise and the whistle signal not being as prominent as we would've liked. In the future, we should produce a cleaner whistle so that it is more clear what happens when we filter this signal.