EMG Analysis and Voice Synthesis using Signal Processing and Filtering Techniques

BME 252 - Final Project Submission

Dr. Alaaeldin Elhady Ahmed
Ryan Jing (20941998), Binalpreet Kalra (20941262), Matthew Keller (20931412)
July 31, 2023

EMG Filter

Root Mean Square

The root-mean-square (RMS) is a scalar value that represents the square root of the squared mean values in a dataset. This acts as a measurement for the power present in a signal and is a valuable metric to assess the size of signals that oscillate about the y-axis, whose average will always be near zero.

Filtering

Input Signal

Two EMG signals represented as an array of floats representing voltage were observed. These signals had no filters or amplification applied and were likely subject to environmental noise, meaning that noise from the environment from which the EMG signals were measured is present. Two filters were applied to the raw EMG singles to remove power line interference and background noise.

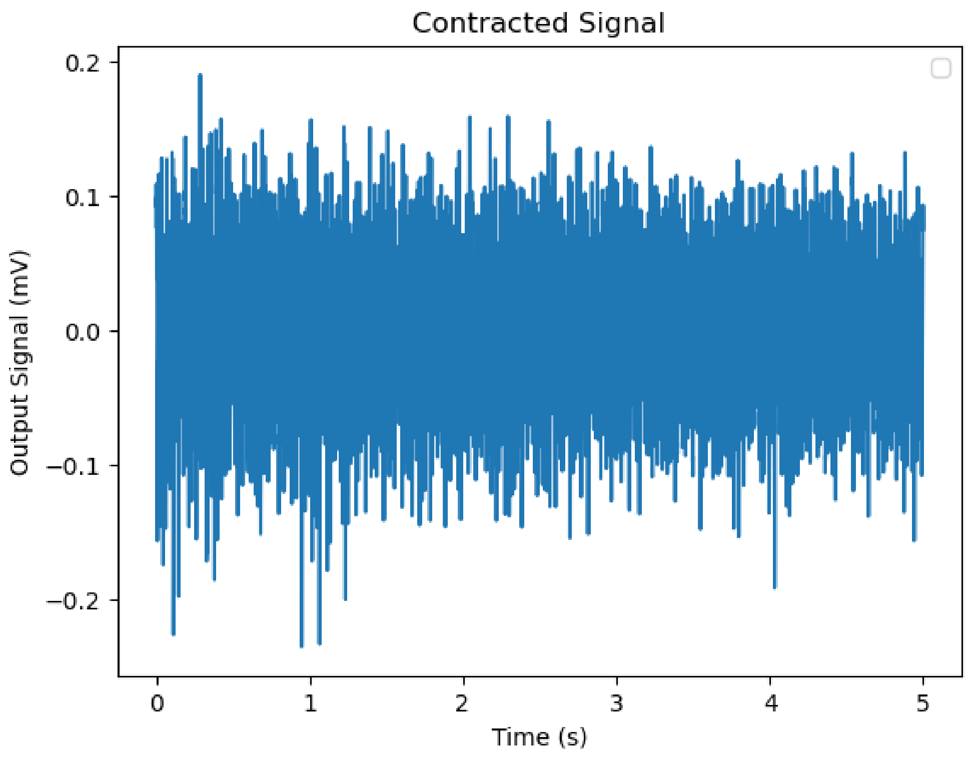

Fig. 1. Plot of Input Contracted EMG Signal in the Time Domain

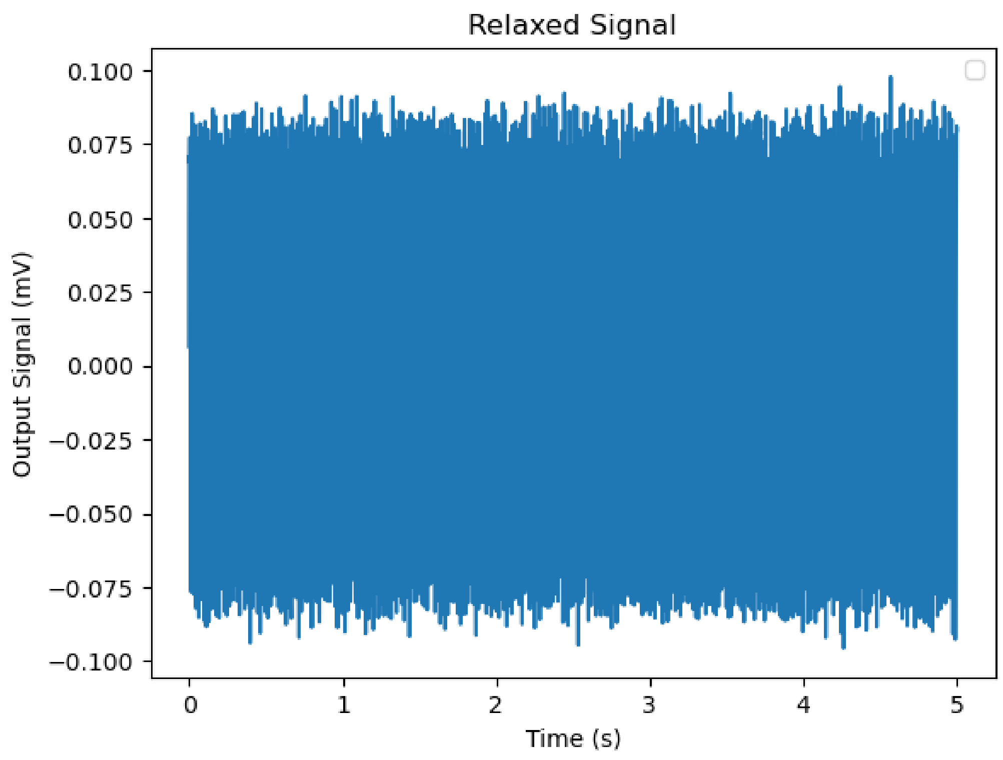

Fig. 2. Plot of Input Relaxed EMG Signal in the Time Domain

Power Line Interference

A notch Butterworth filter with a bandwidth of 6 and centre frequency of 60 Hz was used to filter out power line interference around 60Hz. A bandwidth of 6 was used to ensure that signals of frequencies surrounding 60Hz were entirely removed, while still ensuring that the majority of the frequency spectrum of the input EMG signal was preserved.

A Butterworth filter was used because it has a significant cutoff while not significantly distorting frequencies within the range of interest, such as in a Chebychev or elliptic filter. An order of 8 was used to ensure steep a cutoff. The transfer function of a first order notch filter is shown in Eq. (1). See Appendix Eq. (5) for the transfer function for the filter of order 8 used to remove power line interference.

$$H(s) \; = \; 0.991\left(\frac{(s-(0.9831+0.1828j))\,(s-(0.9831+-0.1828j))}{(s-(0.9742+0.1809j))(s-(0.9742+-0.1809j))}\right)$$

Equation 1. First order transfer function for bandstop filter

Background Noise

To filter frequencies outside the range of expected EMG signals, a bandpass Butterworth filter with a lower cutoff frequency of 0.1 Hz and a higher cutoff frequency of 450Hz were applied to the input after the notch filter. Other hyperparameters were kept consistent with the notch filter. Again, an order of 8 was used to ensure a steep cutoff, and a Butterworth filter ensured that frequencies within the bandpass region were not distorted. The transform function of the bandpass filter is shown in Eq. (2). See Appendix Eq. (6) for the transfer function for the filter of order 8 used to remove background noise.

$$H(s) \;=\; 0.452\left(\frac{(s-1)(s+1)}{(s-0.9997)(s-0.0955)}\right)$$

Equation 1. First order transfer function for bandpass filter

Note: *Although the Butterworth filters were an order of 8, the equation was too large to place in the body of the report. To see the full transfer function, refer to the appendix.*

Output Signal

The RMS of the output contracted signal was found to be 0.031. This implies that the after both filters were applied, the power of the signal decreased due to power line interference being filtered out. The output signal is likely a more accurate representation of the electrical potential produced from the subject's body, as it removes much of the environmental noise present in the input signal.

Additionally, the filtering of the 60Hz noise greatly reduces the corruption of our ability to decide if the muscle is contracted or not. It can be seen when comparing the signals from Figure 1-4 that the difference in average voltage output is much more apparent between contracted and relaxed after filtering. While the RMS of the contracted signal isn't extremely different, the RMS of the relaxed signal is much smaller, allowing for higher confidence in the differentiation between a contracted and relaxed muscle.

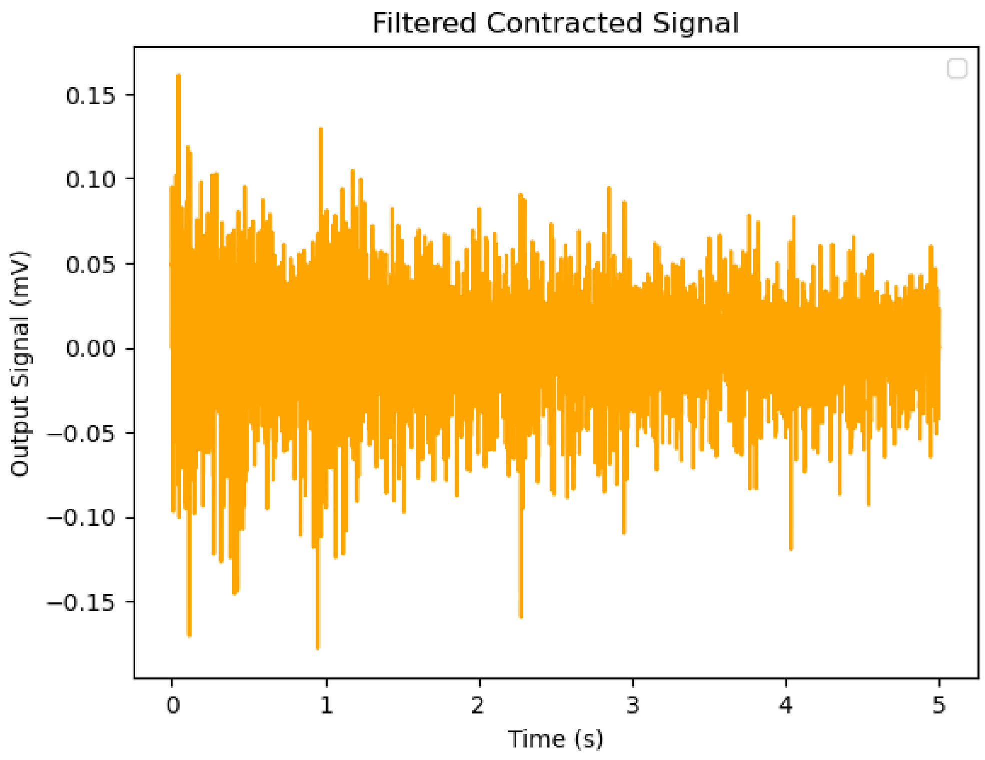

Fig. 3. Plot of Filtered Contracted EMG Signal

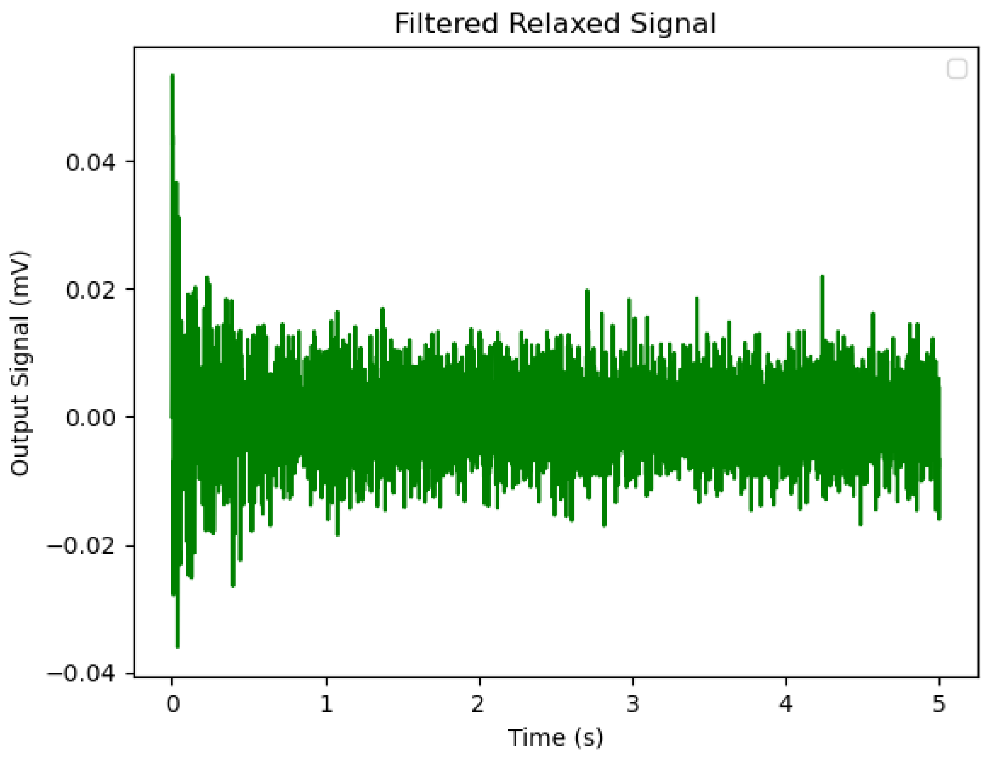

Fig. 4. Plot of Filtered Relaxed EMG Signal

Table. 1. RMS of Signals

Signal	**RMS**
Contracted Signal	0.0645046
Relaxed Signal	0.0568027
Filtered Contracted Signal	0.030966
Filtered Relaxed Signal	0.005995

Results

Fast Fourier Transform

To ensure that the signals being passed are within the expected range of EMG signals when flexed, a band pass filter within the frequencies of 0.1Hz and 450Hz was implemented.

As shown in Figure 7 and Figure 8, the frequency spectra of both the relaxed and contracted input EMG signals show no frequencies at 60Hz and significantly attenuate any frequencies above 450Hz.

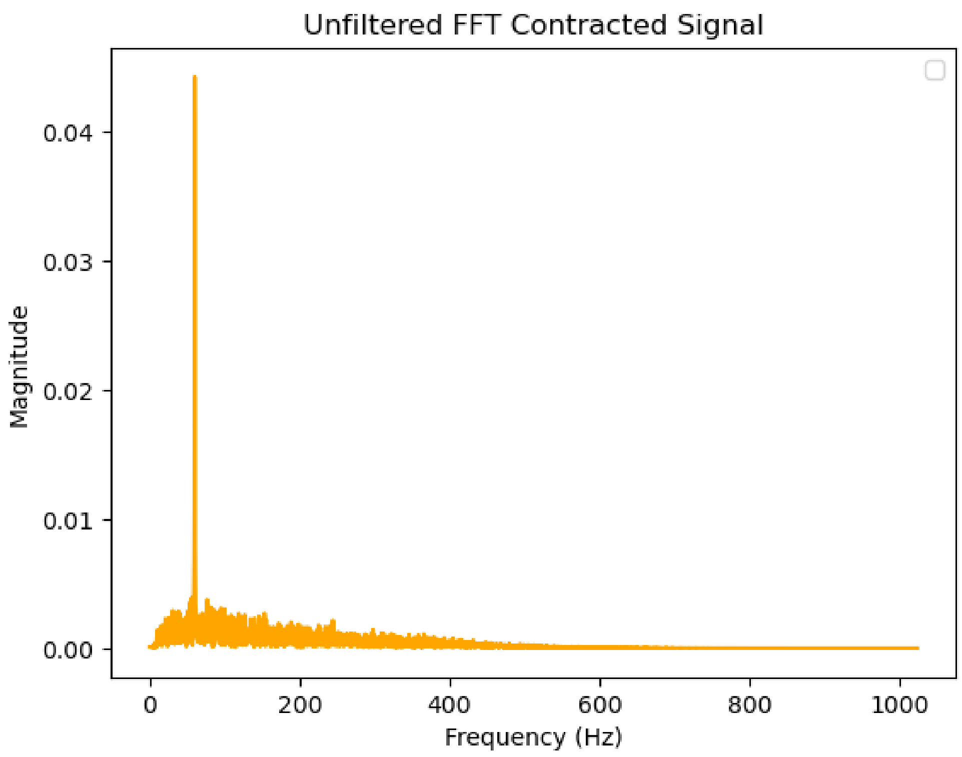

Fig. 5. FFT Plot of Unfiltered Contracted EMG Signal

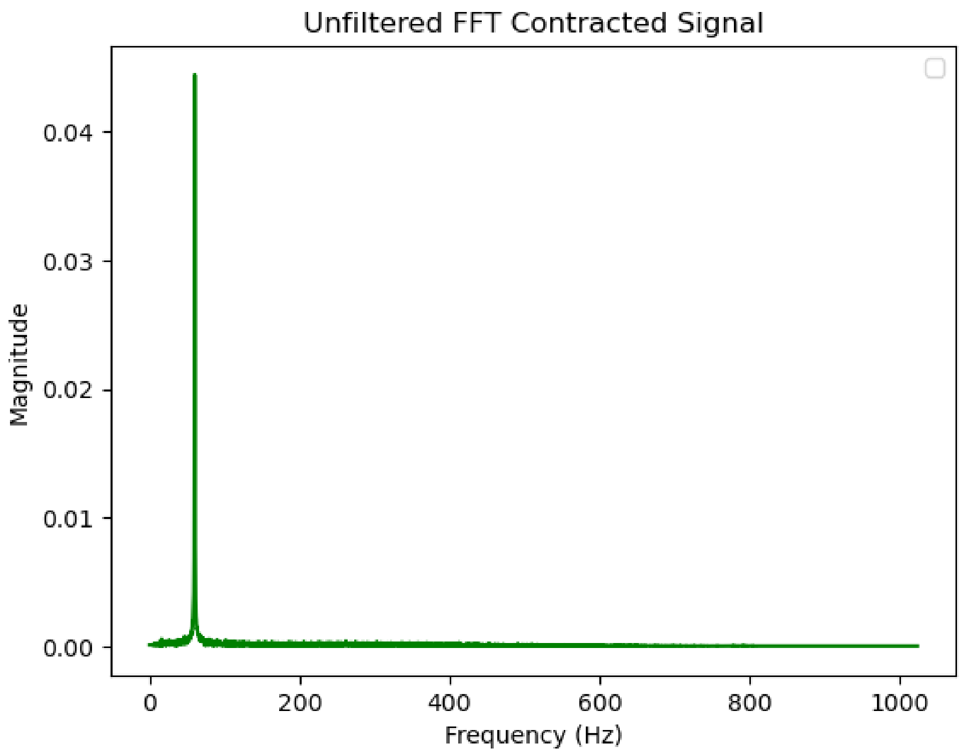

Fig. 6. FFT Plot of Unfiltered Relaxed EMG Signal

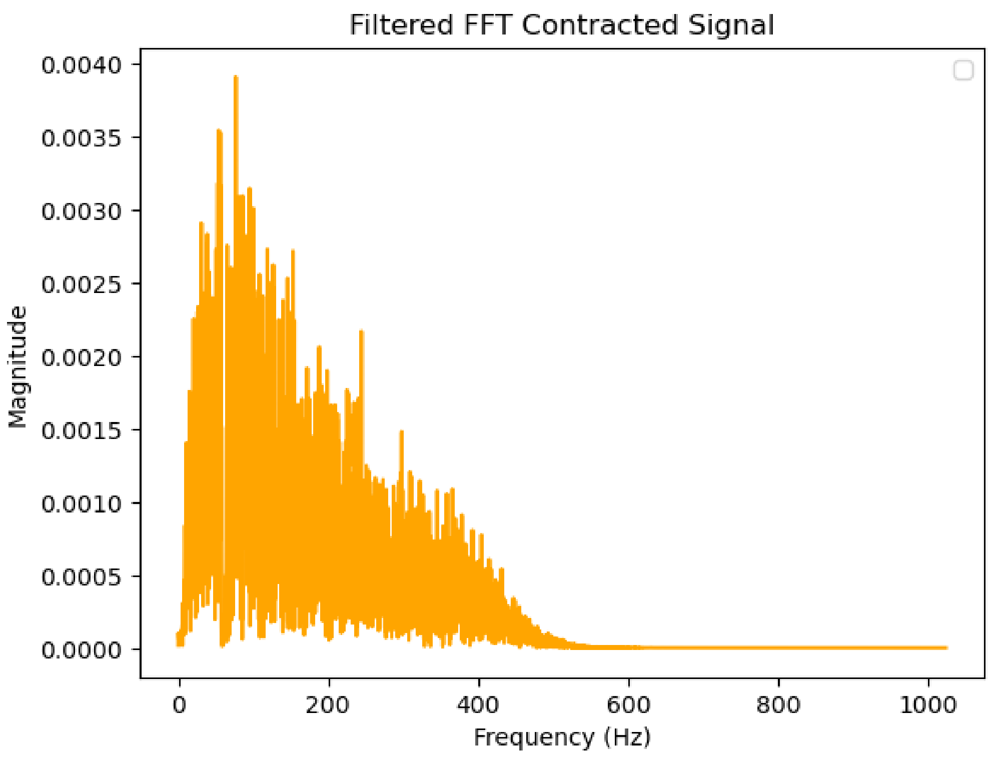

Fig. 7. FFT Plot of Filtered Contracted EMG Signal

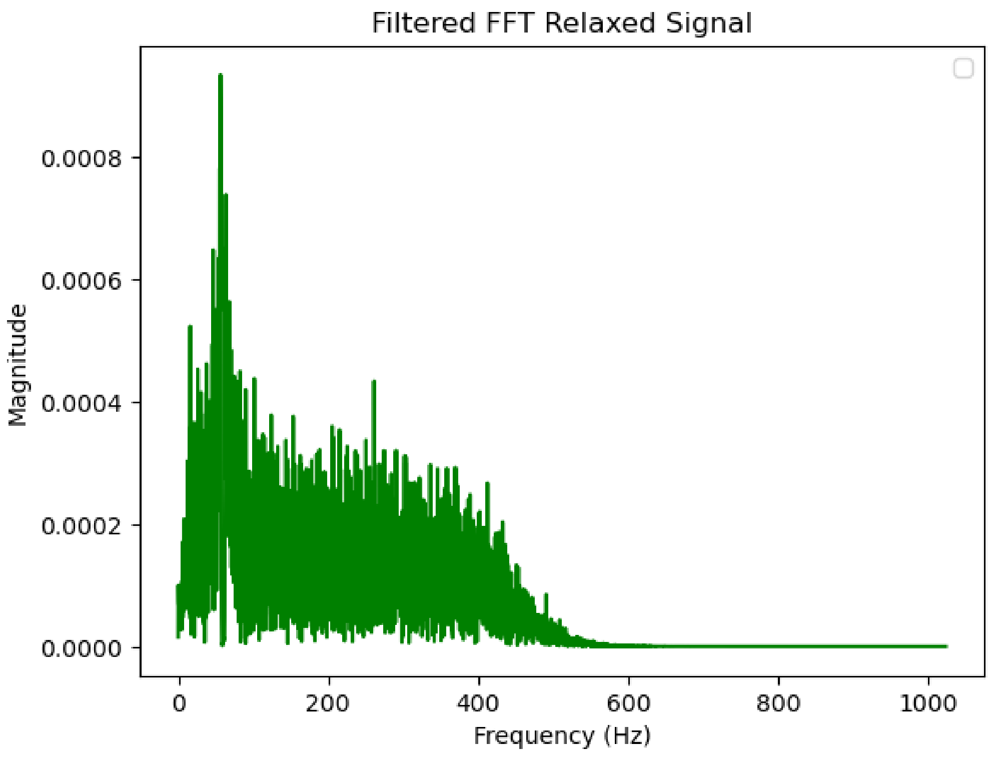

Fig. 8. FFT Plot of Filtered Relaxed EMG Signal

Discussion

The threshold voltage for detecting whether the signal is a contracted muscle should be ~0.025 mV. This threshold voltage would trigger during muscle contraction, while still being high enough that it will not be triggered by the relaxed signal. This value of slightly lower than the RMS value of the contracted signal will allow for the consistent triggering of the threshold, and as the RMS of the relaxed signal is around 5 times lower, a false positive threshold trigger is highly unlikely.

Robo-Voice Synthesizer

Background

Using various signal processing and filtering techniques, an unfiltered sound file of a person saying a popular movie quote was processed to output a sound file that sounds like a robot saying the same input phrase. The quote used was "*I'm sorry I ruined your lives and shoved 11 cookies into the VCR*" from [1].

Preprocessing

A .wav file was produced from a laptop recording of the input signal with a sampling rate of 16kHz. No downsampling was employed to change the sampling rate of the input file, and no channels were discarded in the preprocessing stage. The time stream and frequency spectrum of the input signal compared to the final output signals in Fig. 9 and 10.

Procedure

A filtering and synthesis process was used to convert the input signal to a robotic-sounding output. The input signal was segmented into several arrays representing different time chunks. An array of processing steps was applied to each, wherein the RMS of a filtered chunk was calculated and used as the amplitude of a synthesized sine wave representing that time chunk. The output of each processing step was summed for each time chunk, and each time chunk was concatenated to produce an array representing a continuous stream of sound.

Time Segmentation

The input signal was segmented into time chunks of 40ms (or 640 samples with a sampling rate of 16kHz). Several time chunk durations were tested, and a 40ms chunk size was found to produce the most intelligible output sound. Using shorter time segments resulted in syllables being interrupted, and longer time segments joined together independent syllables. The duration of 40ms likely worked the best, as this is close to the duration of saying an English vowel, which can vary between 50 and 250 ms [2].

There was no overlap between time chunks, as this would have caused overlapping time segments to be synthesized twice, resulting in an output with inconsistent volume and tone. If this approach was used, the amplitude of overlapping time segments could be halved; however, this was not necessary given that non-overlapping chunks produced an intelligible output. The chunks used to produce the syntheiszed audio were consecutive without gaps between each chunk to ensure the word order of the input and output sounds were consistent and that all sounds from the input recording were represented in the output. The chunked audio is shown in Fig. 8.

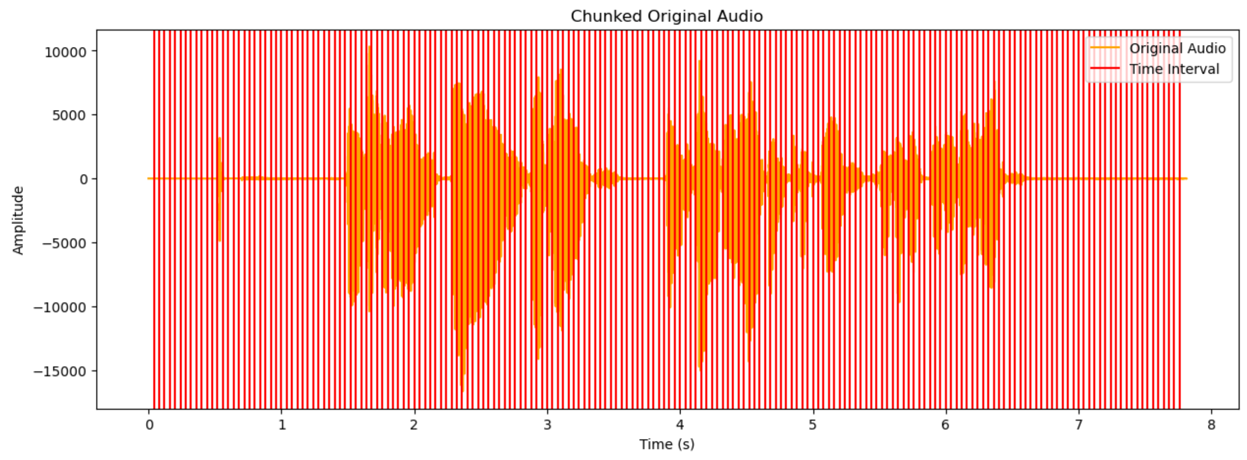

Fig. 8. Original Audio in Time Domain with Divided into Chunks

Processing System

For each chunk, an array of 150 processing pipelines were applied which consisted of filtering the time chunk, calculating the RMS of the filtered output, and multiplying the RMS by a sine wave to synthesize a robotic sound representing a specific frequency spectrum.

Each filter applied was a Butterworth band pass filter with an order of 3. The purpose of applying a filter was to find the RMS value of a segment of a frequency spectrum of the input signal. The first centre frequency used was 100Hz which increased by 50Hz for each filter, while a bandwidth of 100Hz for each filter was kept consistent for each filter. A bandwidth of 100Hz was used, as it was shown to produce an output signal with a similar amplitude as the input signal. The centre frequency was increased by half the bandwidth, such that each frequency segment was represented more than once in the synthesized signal. This ensured the RMS of the array of processing pipelines followed more over a rolling average. A relatively low filter order of 3 was used to ensure adjacent bandpass regions had more overlap.

As there were over 100 filters applied to the input signal, only one transfer function is discussed in this report. Eq. (3) represents a the first Butterworth filter applied to each chunk with a bandpass region of 1Hz to 100Hz and order of 3.

$$H(s) = 7.07 * 10^{-6} \times \frac{(s-1)^3(s+1)^3}{(s-1)^3(s-(0.98+0.03i))(s-0.96)(s-(0.98-0.03i))}$$

Equation 3. Transfer function of a butterworth bandpass filter applied to synthesize audio

The RMS of each filtered chunk was multiplied by a sine wave with the same frequency as the central frequency of the bandpass filter applied. This ensured that the synthesized wave produced represented the same frequency spectrum as the signal input.

Concatenating and Voice Output

The outputs of the processing system applied to each time segment were concatenated to form an output signal that is the same size and shape as the input signal. This vector output was output to a .wav file using [3] so it can be played as a sound.

Results

The resulting .wav audio presents an intelligible synthesized recording of the original quote. As shown in Fig. 9, the frequency spectra of the original and synthesized audio are similar; however, the synthesized audio has spikes for certain frequencies. This is because the frequency of the sine waves used to synthesize each chunk of output audio only represented the frequency of the central frequency of each filter's bandpass region. This resulted in frequencies in intervals of 50Hz being represented more times than others in the synthesized output. This was also why the the overall shape of the output signal in the frequency domain to be similar to the input signal, only compressed into 50Hz spikes.

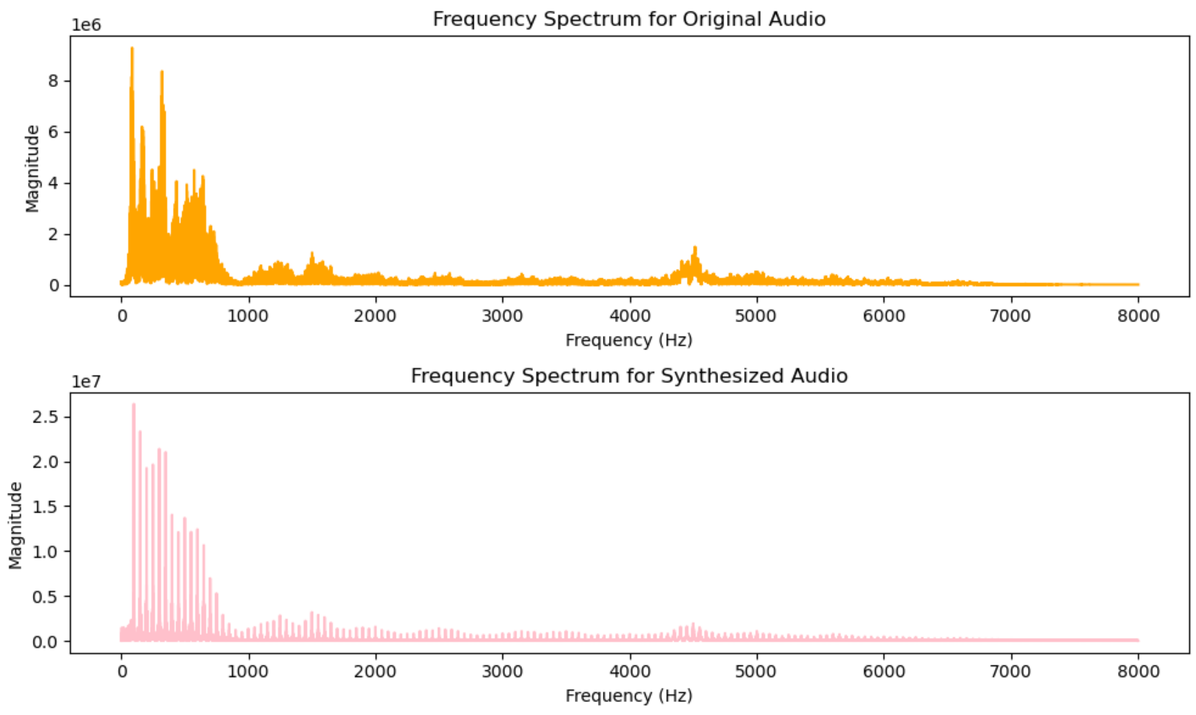

Fig. 9. Original and Synthesized Audio in Frequency Domain

The original and synthesized audio are also similar in the time domain; however, the synthesized audio appears more blocky and with significantly higher amplitude peaks compared to the original audio signal as shown in Fig. 10. This is also likely due to overlapping bandpass regions in the filtering stage, resulting in amplitude spikes in the time domain. As shown in Table. 2, the RMS of the original audio wave is higher than the synthesized audio, however, the maximum absolute value of the synthesized audio was 19.2% higher than that of the original audio wave as shown in Table. 3. See Appendix Eq. (4) for the calculation of percent difference between the RMS of the input and output signals.

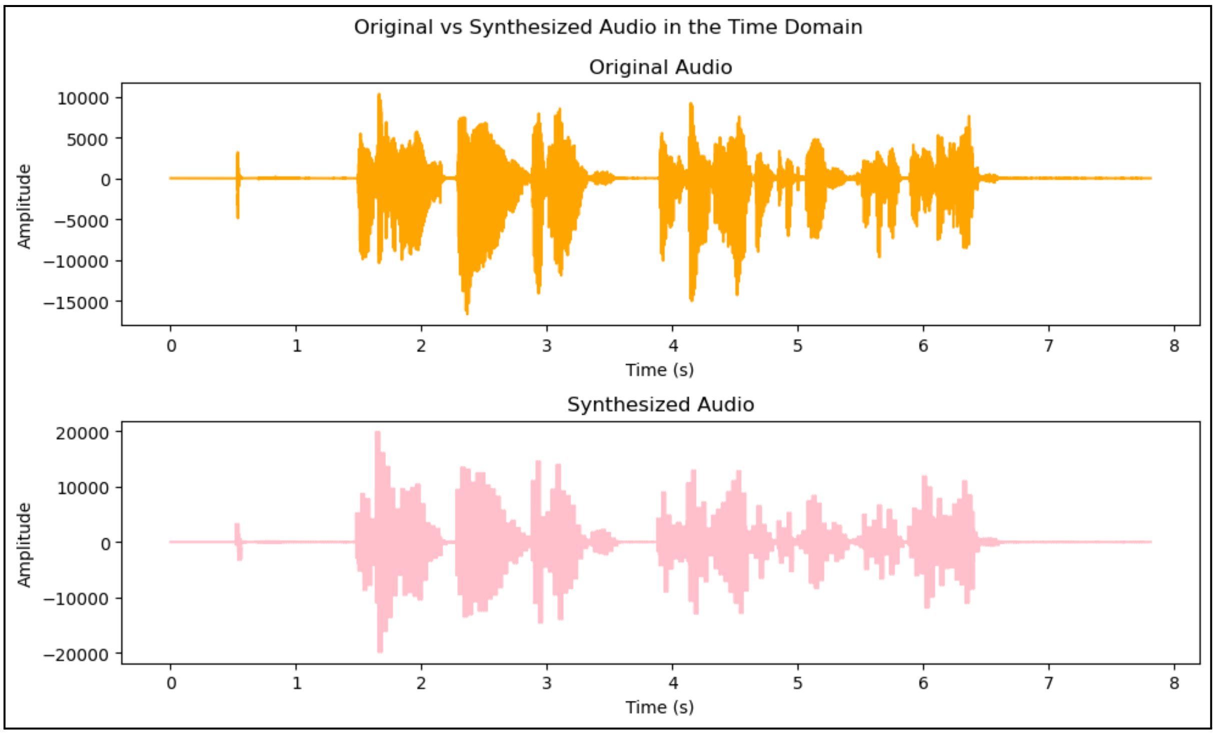

Fig. 10. Original and Synthesized Audio in Time Domain

Table. 2. RMS of Original and Synthesized Audio Signals

Signal	**RMS**
Original Audio	1614.43
Synthesized Audio	1436.66

Table. 3. Maximum Absolute Amplitude of Original and Synthesized Audio Signal

Signal	**Maximum Absolute Amplitude**
Original Audio	16652.00
Synthesized Audio	19852.32

Individual Contribution Statements

Ryan Jing

I processed, analyzed, and graphed the EMG signals in the provided dataset for this project. This included creating scripts to process, filter, and calculate RMS values for the contracted and uncontracted signals provided. I also reported on differences between filtered and unfiltered EMG signals and described the reasons for changes in RMS values and the shape of amplitude graphs between filtered and unfiltered signals. I generated the transfer function for the filters applied to EMG data.

Binalpreet Kalra

I tested the synthesis system for the robo-voice section of the report. I tested using different bandpass regions, orders of filters, types of filters, number of filters, and chunk lengths. I also analyzed why the final hyperparameters reported were able to produce an intelligible audio file using the theory of signal processing and researching the time length of English syllables. I found the transfer function equation for one of the filters applied to the input audio in the robo-voice synthesizer filters.

Matthew Keller

I set up the environment and system to process an audio file to produce a robotic voice. This included setting up an Anaconda environment and creating an easily configurable Python script to synthesize audio from a human recording by changing constant variables at the top of the file, saving the final signal to text and audio files, and automatically graphs the output signal in the time domain. I also analyzed and compared the synthesized audio to the original input to make sense of trends and behaviour of the RMS values and amplitude graphs produced from either.

References

[1] D. Bernbaum, Elf. New Line Cinema, 2003.

[2] I. Toiven, L. Blumenfeld, A. Gormley, L. Hoiting, J. Logan, N. Ramlakhan, A.Stone, Vowel Height and Duration, Cascadilla Proceedings Project, 2014.

[3] "Wave - read and write WAV files," Python documentation, https://docs.python.org/3/library/wave.html (accessed Aug. 2, 2023).

Appendix

$$P = \frac{A_S - A_O}{A_0} = \frac{19852.32 - 16652.00}{16652.00} * 100\% = 19.2\%$$

Equation 4. Percent increase (P) of absolute amplitude from original to synthesized Audio, where $A_0$and A_S are the maximum absolute amplitudes of the original and synthesized original, respectively

H(s) = 4.1125e-03 * ((s - (1.0000 + 0.0000j)) * (s - (1.0000 + 0.0000j)) * (s - (1.0000 + 0.0000j)) * (s - (1.0000 + 0.0000j)) * (s - (1.0000 + 0.0000j)) * (s - (1.0000 + 0.0000j)) * (s - (1.0000 + 0.0000j)) * (s - (1.0000 + 0.0000j)) * (s - (-1.0000 + 0.0000j)) * (s - (-1.0000 + 0.0000j)) * (s - (-1.0000 + 0.0000j)) * (s - (-1.0000 + 0.0000j)) * (s - (-1.0000 + 0.0000j)) * (s - (-1.0000 + 0.0000j)) * (s - (-1.0000 + 0.0000j)) * (s - (-1.0000 + 0.0000j))) / ((s - (0.9999 + -0.0003j)) * (s - (0.9998 + -0.0003j)) * (s - (0.9997 + -0.0002j)) * (s - (0.9997 + -0.0001j)) * (s - (0.9997 + 0.0001j)) * (s - (0.9997 + 0.0002j)) * (s - (0.9998 + 0.0003j)) * (s - (0.9999 + 0.0003j)) * (s - (0.1586 + 0.8083j)) * (s - (0.1223 + 0.5284j)) * (s - (0.1042 + 0.3004j)) * (s - (0.0964 + 0.0976j)) * (s - (0.0964 + -0.0976j)) * (s - (0.1042 + -0.3004j)) * (s - (0.1223 + -0.5284j)) * (s - (0.1586 + -0.8083j)))

Equation 5. Transfer function of Butterworth bandpass filter

H(s) = 9.5391e-01 * ((s - (0.9831 + 0.1828j)) * (s - (0.9831 + 0.1828j)) * (s - (0.9831 + 0.1828j)) * (s - (0.9831 + 0.1828j)) * (s - (0.9831 + 0.1828j)) * (s - (0.9831 + 0.1828j)) * (s - (0.9831 + 0.1828j)) * (s - (0.9831 + 0.1828j)) * (s - (0.9831 + -0.1828j)) * (s - (0.9831 + -0.1828j)) * (s - (0.9831 + -0.1828j)) * (s - (0.9831 + -0.1828j)) * (s - (0.9831 + -0.1828j)) * (s - (0.9831 + -0.1828j)) * (s - (0.9831 + -0.1828j)) * (s - (0.9831 + -0.1828j))) / ((s - (0.9830 + 0.1739j)) * (s - (0.9797 + 0.1745j)) * (s - (0.9768 + 0.1764j)) * (s - (0.9748 + 0.1793j)) * (s - (0.9748 + -0.1793j)) * (s - (0.9768 + -0.1764j)) * (s - (0.9797 + -0.1745j)) * (s - (0.9830 + -0.1739j)) * (s - (0.9796 + -0.1915j)) * (s - (0.9765 + -0.1894j)) * (s - (0.9745 + -0.1863j)) * (s - (0.9739 + -0.1827j)) * (s - (0.9739 + 0.1827j)) * (s - (0.9745 + 0.1863j)) * (s - (0.9765 + 0.1894j)) * (s - (0.9796 + 0.1915j)))

Equation 6. Transfer function of Butterworth bandstop filter