Cochlear Implant Project

Phase 2

Faduma Ahmed (20747788)

Elena Damus (20708019)

Martin Le (20628761)

BME 252

Prof. Nima Maftoon

July 15, 2019

Envelope Application Process and Decisions

To decide which filter types would be used within the bandpass filter bank, both infinite impulse response (IIR) and finite impulse response (FIR) were compared. IIR filters were eventually chosen rather than FIR due to two major reasons. The first is that IIR filters have faster computational times than FIR filters, meaning they will reduce the lag and the time of sound perception for patients using cochlear implants, which is likely a greater priority for patients over the very small alteration in sound quality that is noticed in IIR filters. The second reason is the fact that the time shifting of frequency bands when using an IIR filter is not significant; preliminary tests showed that no significant decrease in audio sound quality was detected, due to the non-linear phase shifting and distortion. These preliminary tests involved applying the bandpass filters which produced N channels, and summing all of those channels up in order to recreate the original audio file. The generated audio files were then compared to see if they were qualitatively similar to the original audio files. Ultimately, it was determined that the use of IIR filters did not produce any significant modulations in the generated sound file and was therefore found to be qualitatively equivalent to the original sound file.

After deciding which type of filter would be used, four different IIR filters were selected to be compared with each other for sound quality and filtering ability: Butterworth, Chebychev, Chebyshev 2, and Elliptic. The filters were created using MATLAB command *FDESIGN.BANDPASS*, with a sampling frequency of 16000Hz and an order of 10. The decision to design the filters with 10th order was made because, through research, it was determined that higher orders tend to have more accuracy and are able to more sharply differentiate between very close frequencies, due to the steep roll-off beyond the cutoff frequency[1]. Higher orders also allow for more flexibility with the rest of the filter design. For example, in using a Butterworth filter, the transition becomes sharper as the order increases, and the stop band is flatter, meaning the attenuation is more accurate[1]. In general, the higher the filter order, the closer the response is to the "ideal response", which looks like a step function - consisting of a slope approaching plus or minus infinity, with a vertical dropoff and horizontal stopband.

Additionally, the filters were all designed so that they would be created via the execution of a function that takes the bottom and top stopband frequencies as arguments. This reduces computation time for the processor, and also would theoretically reduce the audio processing time in a real cochlear implant.

The next step was to determine how many frequency bands would be used and how the frequency spectrum would be divided between the bands. To create the bank of bandpass filters as required for Task 4, the range of frequencies was originally split into 22 equidistant bands from 100 Hz to 8 KHz. The first band contained the frequencies 100-359 Hz, the second contained 360-718, and so on. The reasoning for this was that no initial data had been obtained regarding how well the signal would sound after being split into bands and then summed together again, so this was merely a starting point from which the bands could be further tweaked to improve the effectiveness of the bandpass filter bank. Also, 22 was chosen for N because research determined that cochlear implants generally utilize 8 to 22 channels in their signal processors, and theoretically, the higher number of channels, the better the sound quality[2].

After creating this bank of equidistant filters, running the audio file through them, and then summing the outputs back together, it was determined that the sound quality was qualitatively poorer compared to the original sound clip, and therefore the filter bank needed to be refined. Previous studies on digital audio signal processors were found that reported that the use of asymmetric frequency bands, with smaller bands for lower frequencies and larger bands for higher frequencies, provided much more accurate outputs[3]. Therefore, the original bank was redesigned so that it now consisted of 21 channels with increasing widths as the frequencies themselves increased. This corresponds with the physiological characteristics of the basal-cochlear membrane, as previous research has confirmed asymmetric frequency selectivities at higher frequencies[2]. The frequency bands for each channel were implemented as documented in Table 1 below.

Table 1: Predetermined Frequency Spectrum for Each Individual Bandpass Filter

Channel	# of Frequencies per Channel	Frequency range	Channel	# of Frequencies per Channel	Frequency range
1-3	100	100-400 Hz	**13**	320	2000-2320 Hz
4	110	400-510 Hz	**14**	380	2320-2700 Hz
5	120	510-630 Hz	**15**	450	2700-3150 Hz
6	140	630-770 Hz	**16**	550	3150-3700 Hz
7	150	770-920 Hz	**17**	700	3700-4400 Hz
8	160	920-1080 Hz	**18**	900	4400-5300 Hz
9	190	1080-1270 Hz	**19**	1100	5300-6400 Hz
10	210	1270-1480 Hz	**20**	1300	6400-7700 Hz
11	240	1480-1720 Hz	**21**	300	7700-8000 Hz
12	280	1720-2000 Hz			

Upon running the sample audio file through this bank of filters and summing the outputs once more, the result was found to be much more similar to the original input file. It was therefore decided that this bank would be used, instead of the equidistant bank, due to its improved ability to maintain the quality of the audio file.

It should be noted that in the Phase2Script.m file, there are 2 different sections for applying the bandpass filters to the input sound file. One uses the asymmetrical frequency channels, with frequency spectrum array *A*, and the other uses equidistant channels, with frequency spectrum array *B*. Although the frequency spectrum array *A* was ultimately selected, both were kept in the file to ensure the future ability to compare between the two filter banks.

In order to pass the file through the various filters, a *for loop* was created in which the type of filter (i.e. butterworth, elliptic, etc.) was specified, and then applied to each sample point in the audio file. The outputs were stored in new arrays for each cycle through the loop, so that by the end of the bandpass application process, 21 arrays had been created, each containing a portion of the intensity of the input signal. These new arrays were preallocated as a single array, called *fA*, of length *A*, where *A* is the array containing the 21 asymmetrical bands of the frequency spectrum. They were then populated by creating a new version of *fA* for each *for loop* instance. This implementation method helped cut down on overhead and code processing times compared to the alternative, in which the method would need to be written out 21 separate times.

Plots of the sample audio file being passed through the lowest and highest bandpass frequency channels are shown in Figure 1 below.

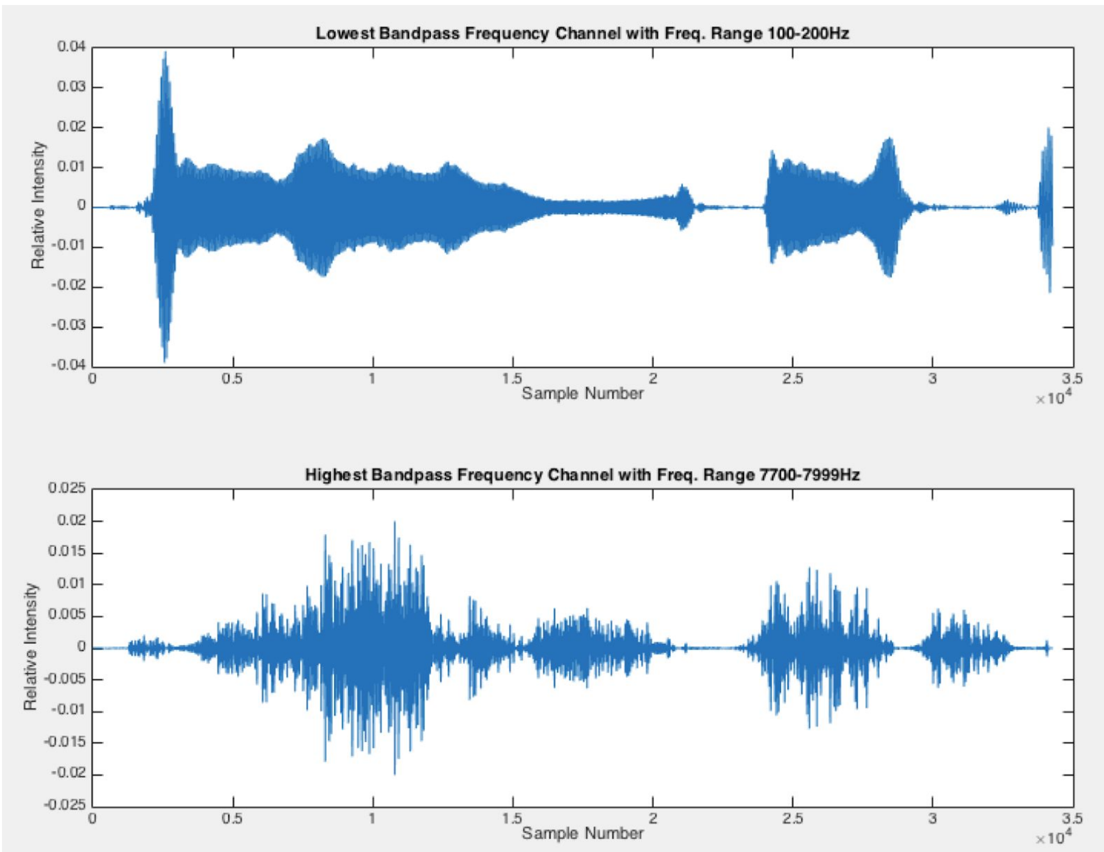

Figure 1: Output of the Lowest and Highest Frequency Bandpass Channels

Envelope Application Process and Decisions

In order to create an envelope for each of the 21 channels, the signal was rectified by taking the absolute value of each of the samples in each *fA* array and then passing these values through a low pass filter with frequency 400 Hz. The type of low pass filter can be changed to achieve a different result. For example, the Butterworth filter has the flattest pass-band out of all the analog filters but a poor roll-off rate, whereas the Chebyshev has some pass-band ripple but a better (steeper) roll-off rate.[4]

In the Phase2Script.m file, a butterworth filter (*LPButter400Hz*) is used. This filter was designed using *FDESIGN.LOWPASS*, with a sampling frequency of 16000 Hz and a cutoff frequency of 400 Hz. The *butter* method was called to apply the Butterworth type to the low pass filter. The Chebychev, Chebyshev 2, and Elliptic low pass filters were also created in a similar way, and again, all of the low pass filters were designed to be 10th order, for the same reason as outlined above for the bandpass filter banks. The Chebyshev filter was designed to have a passband frequency of 400 Hz and a passband ripple of 1 dB, the Chebyshev 2 filter had a stopband frequency of 400 Hz and a stopband attenuation of 80 dB, and the Elliptic filter had a passband frequency of 400 Hz, a passband ripple of 1 dB, and a stopband attenuation of 80 dB. These values are defaults provided by the filter designer in MATLAB, but it was determined that they did not require adjustment. For example, it is known that a low passband ripple of 1dB rejects most of the unwanted rippling in the Chebyshev 1 and Butterworth filters. Although a higher stopband attenuation of 80dB for both the Chebyshev and Elliptic filters would cause a steeper roll-off, it would minimize the rippling effect of the filter.[4] This is an optimal trade-off between these two parameters.

Plots of the envelope for the lowest and highest bandpass frequency channels are shown in Figure 2 below. The envelope outlines the minimum and maximum values that an *fA* signal can take for each sample point. When plotted overtop of one another, the envelope signal completely encases the corresponding *fA* signal as created by the bandpass filter bank.

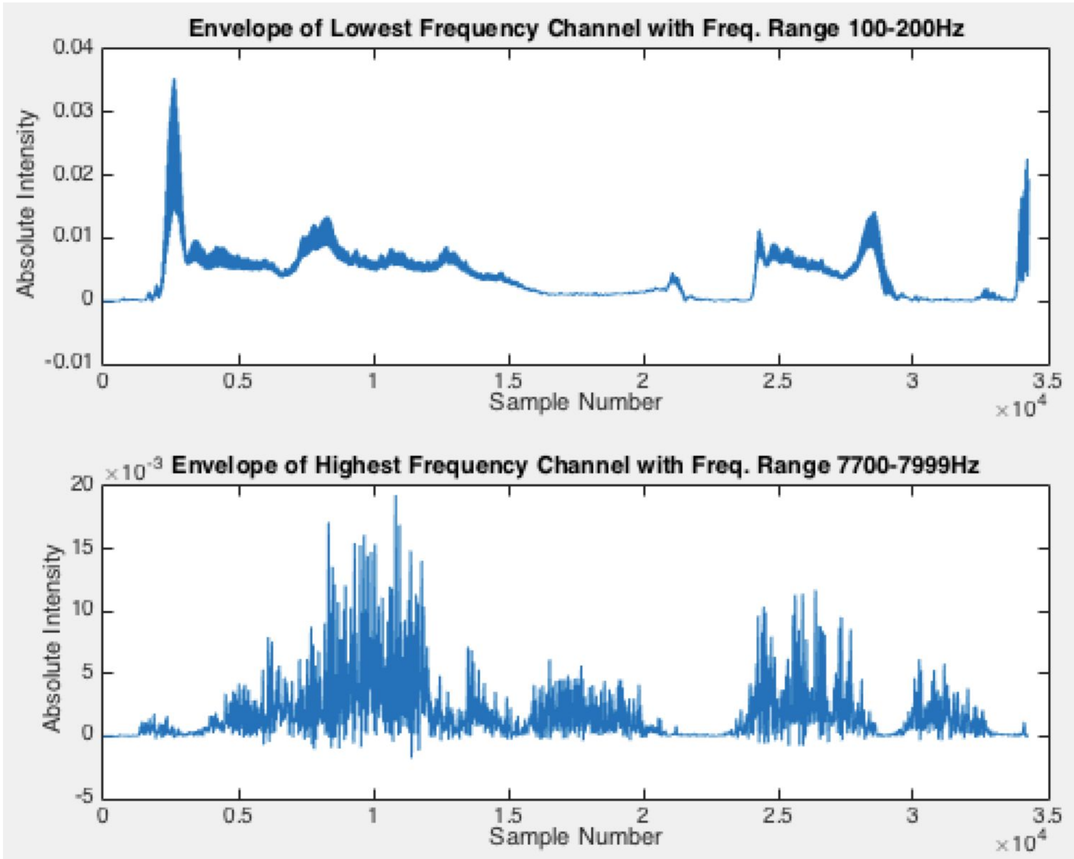

Figure 2: Lowest and Highest Bandpass Channel Envelopes

References

1. Electronics Tutorials, 2018. Butterworth Filter Design. Available online: https://www.electronics-tutorials.ws/filter/filter_8.html. [Accessed: July 13, 2019].
2. Fouts Patrick, J. and Clark, G.M. (1991). The Nucleus 22-channel cochlear implant system. *Ear and Hearing*, *12*(4), pp. 3S-9S. doi:10.1097/00003446-199108001-00002
3. Boucherit I. and Guerti M. (2016). Speech analysis in cochlear implant using auditory filter bank model. *2016 8th International Conference on Modelling, Identification and Control (ICMIC)*, pp. 274-278. doi: 10.1109/ICMIC.2016.7804122.
4. University of Rhode Island, 2010. “Classic Filters” (Lecture). *University of Rhode Island.* Available online: https://www.ele.uri.edu/courses/ele314/handouts/YS06_Classicfilters.pdf. [Accessed: July 12, 2019].