Project 1 – EECS 152B

Cameron Peterson-Zopf

57999719

Question 1a: Windowing Method

For this question, I used the hamming window method. I found this window method was superior due to its low sidelobes allowing me to reduce the amount of the samples I needed while maintaining the restriction that the ends of the graph be below -50 dB. My code for this question is below in figure 1.

A screenshot of a computer program

Description automatically generated

Figure 1 – Code for Question 1a

Plotting the result, figure 2 displays the overall graph of the magnitude with respect to normalized frequency.

A graph of a normalized frequency

Description automatically generated

Figure 2 – dB Gain vs Normalized Frequency for Question 1a

Zooming in on each region, the requirements for the ranges of acceptable dB gain are satisfied as can be seen in the following graphs.

A graph of a normalized frequency

Description automatically generated

Figure 3 – 0 to 0.2pi Criteria

Figure 3 illustrates how the dB gain remains below -50 dB between 0 to 0.2pi.

A graph of a normalized frequency

Description automatically generated

Figure 4 – Criteria at w = 0.25pi

From the graph, it is evident that at w = 0.25pi, the criteria for dB gain between -5 and -3 is met. Furthermore, as seen in figure 5, the entire range between 0.25pi and 0.35pi remains between -5 and -3 dB.

A graph of a normalized frequency

Description automatically generated

Figure 5 – Criteria between 0.25pi and 0.35pi

Between 0.4pi and 0.6pi the graph must be in the range of 0 to 1 dB. Displayed in figure 6 is this region, where these requirements are met.

A graph of a normalized frequency

Description automatically generated

Figure 6 – Criteria for 0.4pi to 0.6pi

Moving on to 0.65pi to 0.75pi, this region requires the dB gain between -5 and -3 dB, which is satisfied as can be seen from figure 7.

A graph of a normalized frequency

Description automatically generated

Figure 7 – Criteria between 0.65pi and 0.75pi

Zooming in on 0.75pi, we see that at this point the dB gain is above the -5 dB threshold as seen in figure 8.

A graph of a normalized frequency

Description automatically generated

Figure 8 - Criteria at 0.75pi

Lastly, the requirements are satisfied at the 0.8pi – pi range. Figure 9 displays how the 0.8pi frequency is below the -50 dB threshold.

A graph of a normalized frequency

Description automatically generated

Figure 9 - Criteria at 0.8pi

As displayed, the minimum number of samples required to meet the criteria was 93. Below 93 samples, I could not create a filter where the requirements were satisfied. To create the filter, I added 3 bandpass filters together, where each bandpass was constructed through a shifted low pass filter. The sync functions create the low pass filters, with the complex exponential in the formulas serving to shift the center point of the low pass filter to create the bandpass filter. The hamming window was utilized to dampen the sidelobes in order to meet the -50 dB threshold while maintaining a reasonable slope in the transition band. I found through experimenting with the other bands that either the sidelobes were too big or the transition band’s slope was too shallow, meaning the -5 dB threshold could not be met by 0.25pi.

Question 1b: Parks-McClellan Method

The Parks-McClellan algorithm utilizes an algorithm to find the optimal Chebyshev FIR filter. This algorithm utilizes weights in order to bias reducing the error in certain sections rather than others, adding flexibility for the user in order to target stopbands and passbands. Through trial and error, I found that the least number of samples required to meet the specifications was 61. I determined what gain specifications to give my filter based off of initially inputting in the mean value of the range of satisfactory dB gains, and then modifying these slightly to get down to 61 samples. I weighted the stopband heavily in order to reduce the sidelobes. The resulting code is displayed below in figure 10.

A screenshot of a computer screen

Description automatically generated

Figure 10 – Code for Question 1b

Plotting the above code yields the following graph in figure 11.

A graph of a normalized frequency

Description automatically generated

Figure 11 – dB Gain vs Normalized Frequency for Question 1b

Zooming in on each region, the requirements for the ranges of acceptable dB gain are satisfied as can be seen in the following graphs.

A graph of a normalized frequency

Description automatically generated

Figure 12 – Criteria for 0.2pi

At 0.2pi, the requirement of below -50 dB gain is met as seen in figure 12. Furthermore, the -5 dB to -3 dB range is satisfied in the 0.25pi to 0.35pi region.

A graph of a normalized frequency

Description automatically generated

Figure 13 – Criteria for 0.25pi to 0.35pi

Furthermore, figures 14, 15 and 16 illustrate that the graph maintains a range between 0 and 1 dB between 0.4pi and 0.6pi.

A graph of a normalized frequency

Description automatically generated

Figure 14 – Criteria for 0.4pi

In addition, as we the ripples of the region we see that none of them go outside of the wanted range as demonstrated in figure 15.

A graph with numbers and a line

Description automatically generated

Figure 15 – 0.4pi to 0.6pi Region

Analyzing the 0.6pi point with more zoom, we see that this point is above the 0 dB threshold.

A graph of a normalized frequency

Description automatically generated

Figure 16 – Criteria for 0.6pi

Next, looking at the 0.65pi to 0.75pi, the curve is between -3 dB and -5 dB as seen in figure 17.

A graph of a normalized frequency

Description automatically generated

Figure 17 – Criteria between 0.65pi and 0.75pi

Finally, looking at the 0.8pi location, it is evident from figure 18 that the dB gain is below -50 dB.

A graph of a normalized frequency

Description automatically generated

Figure 18 – Criteria for 0.8pi

Thus, this report has shown that with 61 samples the Parks-McClellan filter algorithm is able to meet the requirements of the question. Reducing the samples below 61 samples resulting in the dB conditions not being satisfied in one or more regions, thus 61 samples was the minimum length needed. Comparing the Hamming filter with the Parks-McClellan, the Parks-McClellan is superior due to less samples (61 vs 93) needed to meet specifications. Furthermore, analyzing curvature, we see that the Parks-McClellan has much more ripples than the Hamming filter, but that this ripples are roughly the same in peak-to-peak amplitude for a given section, hence why the requirements for dB could be met with fewer samples.

Question 1c: IIR Filter

The requirements for the previous two filters will now be implemented with an IIR filter. Specifically, the filter will be a parallel combination of elliptical IIR filters. The code for the IIR elliptical filter is shown below in figure 19.

A screenshot of a computer program

Description automatically generated

Figure 19 - Code for IIR Filter

The dB Gain is plotted in figure 20 with respect to normalized frequency.

A graph with a line and text

Description automatically generated with medium confidence

Figure 20 – dB Gain Plot for IIR Filter

From figure 20, it is clear that the requirements for the filter have been met in all of the ranges. Next, the phase response is shown in figure 21.

A graph with a line

Description automatically generated

Figure 21 – Phase Response for IIR Filter.

From the phase response plot, we see that this phase is not linear, as was the FIR filters. However, there are some linear regions. For the stop bands between 0 and 0.2pi and between 0.8pi and pi we see that the phase is roughly constant. Furthermore, in all of the passbands, the phase is roughly linear, though with different slopes in each. In the transition bands is where we see the most non-linear behavior as the phase must jump between the stopbands and pass bands. These results are further illustrated in the group delay plot of figure 22.

A graph with blue lines

Description automatically generated

Figure 22 – Group Delay for IIR Filter

From the plot, we see that all of the major spikes in group delay occur in the transition bands. Furthermore, the stopbands are very near linear, whereas the passbands are less linear in nature but till oscillate in a small region of values. Thus, while the IRR filters do not have linear phase, they do approximate phase in all regions except for the transition bands. Furthermore, the IIR filter was able to satisfy the requirements with an order of 5, much lower than the lengths required from the FIR filters.

Question 2: Frequency Sampling w/ IFFT

Question 2 requires students to sample a random DTFT in the shape of a batman, and then utilize the inverse FFT to create an impulse response. Students then utilize the DTFT function in MATLAB to create a DTFT of the impulse response that was previously calculated. This DTFT should match the original DTFT. Utilizing 41 samples, I was able to create a similar curve. Figure 23 shows the code for this process.

A screenshot of a computer screen

Description automatically generated

Figure 23 – Code for Question 2

The following batman curve is generated from my code.

A graph of a normalized frequency

Description automatically generated

Figure 24 – Batman Graph Generated from Sampling

As we see from figure 24, this curve is a very good representation of the original Batman curve. The signal’s peak values are around 2.7 and 3, and the troughs are a little less than 1 just as with the original. For comparison, the original curve is displayed in figure 25.

A graph of a normalized frequency

Description automatically generated

Figure 25 – Original Batman Curve

Furthermore, if the number of samples is reduced to 21, meaning the N = 20 and the length of the DTFT is 21, the shape almost resembles a batman, with some slight differences as can be seen in figure 26.

A graph of a normalized frequency

Description automatically generated

Figure 26 – Batman Curve with 21 Samples

With 21 samples, the amplitudes of the peaks and troughs are still roughly accurate; however, the troughs are not as smooth as before, and the straight line that connected the two peaks in the center of the graph is now gone. Furthermore, the filter goes to zero gain prior to 0.8pi, slightly faster than before. Decreasing the length any further caused extreme distortion to the signal, as the amplitudes of the various peaks and troughs began to change, and the filter got less smooth. Thus, with 41 samples, a near perfect representation is achieved, whereas with 21 samples, a decent approximation is achieved.

Analyzing the code, the frequencies are sampled every 2/N in the normalized frequency domain as is standard with a DFT. The data is shifted with an FFT shift prior to the inverse FFT, because the IFFT assumes the data is between 0 and 2pi. When I did not have the IFFT, the batman curve had a flipped shape. Then, the resulting impulse function goes through a DTFT of the same length. I chose to have the DTFT the same length as the DFT from prior, because if the DTFT was longer, there would be zero padding which would mess up the curve of the graph. Finally, the results are plotted with normalized frequency.

Question 3: Beamformer Design

The first part of this question asks students to design a beamformer using the Parks-McClellan algorithm that passes signals between -20 and -5 degrees with a dB gain between -2 and 2 dB. Furthermore, there are no requirements on the length of the transition band, only that the sidelobes should be as low as possible. Furthermore, the number of antennas is 30, and thus the M is 29 in this problem. With the object of low sidelobes, the transition bands will be long. As the transition bands widen the sidelobes will lessen. However, the purpose is to build a filter focused between -20 and -5 degrees and not say -30 to 5 degrees. Thus, I will present a filter that has a moderate transition band with sidelobes that are still quite low. Hence, I am presenting a filter that maintains both objectives without compromising one or the other. The code for this part of question is given in figure 27.

A screenshot of a computer program

Description automatically generated

Figure 27 – Code for Parks-McClellan Beamformer Filter

The resulting filter generated is shown in figure 28.

A graph of a graph

Description automatically generated

Figure 28 - Parks-McClellan Beamformer Filter

Moreover, figure 29 illustrates the -20 to -5 degree region satisfying the requirements.

A graph with numbers and lines

Description automatically generated

Figure 29 – Passband Region of Parks-McClellan Filter

The second part of this question requires students to shift the filter to the region of 5 to 20 degrees, without utilizing a frequency shift. Besides the frequency shift transformation to the filter, the other quick way to repurpose the previous filter is to perform a time reversal. The discrete time Fourier transform states that if we perform a time reversal, then this will yield a frequency reversal in the normalized frequency domain. The following code is then added to figure 27 to yield figure 30.

Figure 30 – Code for Frequency Reversal of Beamformer Filter

A screen shot of a computer

Description automatically generated

As seen above, the only change to the code was the time reversal code on line 5. The “fliplr” function flipped the impulse response values about the y-axis and resulted in the following graph in figure 31.

A graph of a graph

Description automatically generated

Figure 31 – Plot for Frequency Reversal of Beamformer Filter

The plot above is exactly the same as previously, except for the range of the passband is now between 5 and 20 degrees. Figure 32 illustrates that the passband region satisfies the conditions of the question.

A graph with numbers and a line

Description automatically generated

Figure 32 – Plot for Frequency Reversal of Beamformer Filter in Passband Region

The time reversal property is easier to implement than the frequency shift property and displays how very little modification to the code is necessary to change the filter.

The final part of question 3 involves two narrow beams at -20 degrees and -5 degrees. I chose the Hamming window because this window most effectively separated the two beams from each other. The other windows had much less of a magnitude drop between the two beams and were thus not the preferred choice due to the inability to resolve the two peaks if the beams were moved closer together. The code for these two narrow beams is shown in figure 33.

A screenshot of a computer program

Description automatically generated

Figure 33 – Code for Two Narrow Beams

The results of figure 33 are plotted and displayed in figure 34.

A graph with blue lines

Description automatically generated

Figure 34 – Graph for Two Narrow Beams

The two beams peak at -20 and -5 degrees as wanted in the directions. Furthermore, the sidelobes are at least 30 dB lower than the beams, which is a satisfactory attenuation.

The second part of the beamformer design prompts students to move the beams closer until the two cannot be resolved. I am using the definition of resolved to be when I can see at least two small peaks. I determined that the two peaks could not be resolved when the exponentials in the time domain have frequencies of -0.275pi and -0.155pi. These frequencies correspond to the beams having ideal positions of -15.962 and -8.917 degrees respectively. The code involved in this part is shown in figure 35.

A screenshot of a computer program

Description automatically generated

Figure 35 – Unresolved Beams in Spatial Filter

The graph of these unresolved beams is shown in figure 36.

A graph with a line graph

Description automatically generated

Figure 36 – Plot of Unresolved Beams in Spatial Filter

The last section of the spatial filtering involves creating two beams that are one degree apart. I found that with 251 antennas the two beams can be resolved, but lower than that and the two beams cannot be clearly resolved. I chose the two beams to be at 1 and 2 degrees. The code for this filter is shown in figure 37.

A screenshot of a computer program

Description automatically generated

Figure 37 – Code for Closely Located Beams

The plot of these closely located beams is displayed in figure 38.

A graph with a line graph

Description automatically generated

Figure 38 – Code for Closely Located Beams

The two beams can be distinguished in figure 38, thus satisfying the requirements.