Introduction

In this project I used MATLAB to simulate a SONAR signal and compute the spectral analysis of said signal. The CTFT of the signal was calculated and using the Nyquist rate or the rate of sampling was found to be later used in our DFT processing to achieve desired amounts of aliasing. A second interfering SONAR signal was introduced to the system which was adjusted to determine the impact on the original signal and spectral analysis. Lastly, a FIR bandpass filter was created to extract the interfering signal from the combined signal.

Task 1: Choose Fs to Control Aliasing

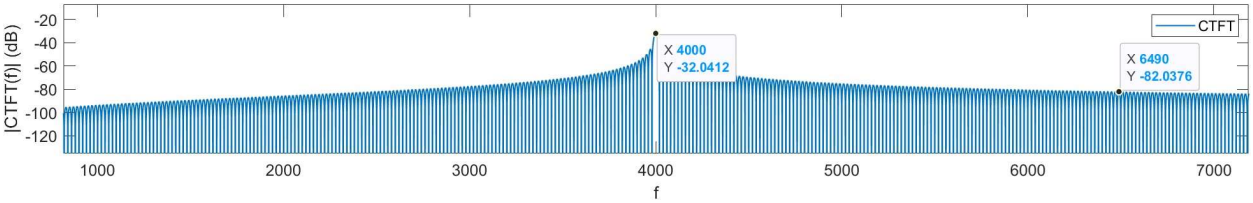
The first task was to compute a spectral analysis on the signal $x_1(t)$ to find an appropriate Fs that will limit any aliasing in the DFT. The given signal:

$$x_1(t) = p_{To}\left(t - \frac{T_o}{2}\right)\cos\left(2\pi f_o t\right),$$

where T_o is the pulse width in sec and f_o is the "carrier frequency" given in Hz. In my case $T_o = 50E-3\ Seconds$ and $f_o = 4000\ Hz$. Before values of T_o and f_o were plugged in the CTFT of the signal $x_1(t)$ was calculated using the Fourier transform table. The resulting signal is:

$$X_1(f) = \left(\left(\frac{T_o}{2}\right) * e^{j\left(\left(\frac{T_o}{2}\right)(2\pi f + f_o * 2\pi)\right)} * sinc\left(\left(\frac{T_o}{2\pi}\right)(2\pi f + f_o * 2\pi)\right)\right) + \left(\left(\frac{T_o}{2}\right) * e^{j\left(\left(\frac{T_o}{2}\right)(2\pi f - f_o * 2\pi)\right)} * sinc\left(\left(\frac{T_o}{2\pi}\right)(2\pi f - f_o * 2\pi)\right)\right)$$

When this signal is plotted against f in a non-dB scale it is expected to get 2 spikes, one at positive f_o and the other at negative f_o, because of the sinc function. I also expect the spikes to go up to $\left(\frac{T_o}{2}\right)$ as the amplitude because both $sinc(x)$ and e^{jx} will not go above one or negative one. In my project I expect the spikes to go up to 0.025 and they should be located at 4000 and -4000 Hz, which is what I am getting out of my code. Since the CTFT of my signal is what is expected in non-dB, it was plotted in dB vs frequency to calculate Fs. This was done by looking at the highest value on the dB plot and finding the frequency where the value of the plot is at least 50 dB below the highest point. The image below is a screenshot of my CTFT (dB) vs Frequency, where X is Frequency in Hz and Y is the CTFT in dB, it is shown that the value of the CTFT's highest point is at 4000Hz and -32dB and at 6500 Hz the CTFT is at -82dB. Therefore, that is our Nyquist frequency, and it is doubled to give the Fs of 13000Hz.



We can verify the Fs value we chose by computing the DFT and seeing if it resembles the CTFT. An array of samples n that goes from 0 to $Fs * T_o$(The length of the pulse) was created it

should look like 0,1,2,3…, $T_o * Fs$. The length of the pulse n is multiplied by the period T which is 1/Fs to get equal spaced time stamps (t_n) which can replace t in the $x_1(t)$ signal. This creates a new discrete time signal that has samples taken every n*T second. Since T_o was used to create n, the resulting signal will be a rectangular pulse which creates the p function in the signal. Fs determines the sampling rate which will allow DFT signal to closely match the original CTFT. The fftshift(fft()) command expects a discrete time signal so we can use this new signal to create the DFT. The DFT is plotted against f which is another array that is equal length to the DFT signal that goes from -Fs/2 to (Fs/2)-1. We can tell the signal is accurate because the DFT and CTFT look very similar and the DFT is basically at 0 by the time it reaches Fs/2. Therefore, there is a reasonable amount of aliasing in the signal. When looking at the DFT peaks zoomed in there is no sharp curves or straight lines and it is centered around f_o or 4000Hz.

Task 2: Exploring the Impact of a Second Signal (Even)

This task wanted to see how the addition of a second SONAR signal would affect the original signal we were given. The second signal was given as:

$$x_2(t) = Ap_{To}\left(t - \frac{T_o}{2}\right)\cos\left(2\pi(f_o - \Delta f)t\right)$$

where A < 1 and Δf>0. In my task I was given $\Delta f = 0.5f_o$. The purpose of this takes was to find the smallest value of A that will still allow the DFT processing to still recognize the second signal. There was no criteria for how A was found, so I took the trial-and-error method. I started with A = 1, which is the highest I would expect it to go in this example. It was obvious that the second signal was there with peaks at a value of around 300. My process is to decrease A by a factor of 10 every time and see where the second signal becomes unrecognizable. When A was 0.1 the amplitude of the second signal went to around 30 and was still clearly visible. When A was 0.01 the amplitude of the second signal went to around 4 which in my opinion is too low considering the lowest part of the graph is around 1. I wanted the amplitude to be at least 10 times the lowest point, so I increased A by 0.01 until the amplitude of the second signal was at least 10 which happened at A = 0.03. I choose a difference of 10 from the lowest point on the graph to the height of the second signal because I wanted at least 1dB of difference between the two values so that the DFT processing can still consider it a second signal instead of merging it into the first. A signal could still be seen at values of A lower than 0.03 but I felt that they were too low to be able to confidently say the second signal was there.

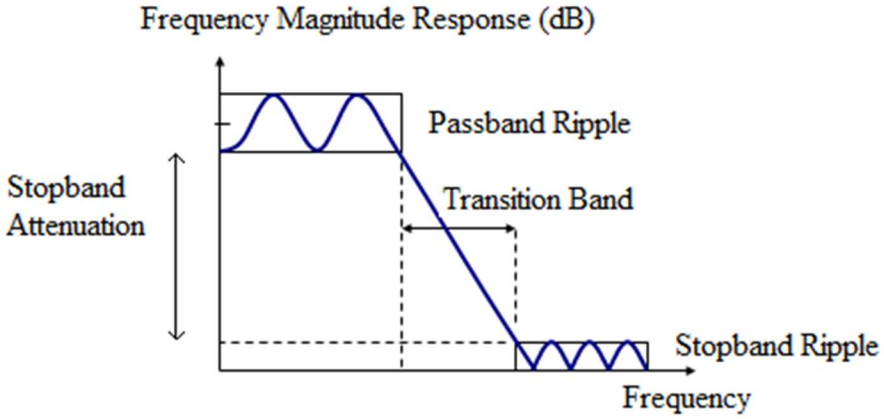
Task 3: Designing and Applying a Filter to Extract the Second Signal

This task gave a new second signal:

$$x_2(t) = Ap_{To}\left(t - \tfrac{T_o}{2}\right)\cos\left(2\pi f_2 t\right),$$

where $f_2 = 0.6 * f_o$ and A=1. This signal was added to the original signal much like the second task. However, the goal of this task is to design a bandpass filter that will allow the second signal through but will block the original signal. The combined signal $x_T(t)$ was created by adding or combining $x_1(t)$ and $x_2(t)$. The combined signal was plotted over frequency/Fs (Normalized) to find the location of the passband edges of the second signal. This resulted in 4 spikes, two for

$x_1(t)$ and two for $x_2(t)$ going from -0.5 to 0.5. The output should be the two spikes for $x_2(t)$ but without the two spikes for $x_1(t)$. This is done by creating a bandpass filter with the bandpass centered around $x_2(t)$ and small enough that $x_1(t)$ is not included. The filter created needed to fit within certain set parameters. The passband must be centered around f2, the passband ripple must be 1dB, the stopband attenuation must be 80dB, the filter order must be less than 200, and must be even. The picture below shows what passband ripple and stopband attenuation is on a lowpass filter for the band bass there would be a stop band on either side of the pass band. The filter order is calculated by the firpmord command and is dependent on the start and stop of the band edges.



The firpmord command used takes 4 inputs, AA(which is a vector that tells firpmord what type of filter it should expect), dev (which determines the passband ripple and stopband attenuation based on values rp, and rs in dB which are 1 and 80 respectively for the filter we want), the sampling rate, Fs, and the frequencies of the band edges (for a band pass there are 4 of these values, the start of the band has two frequencies the start and end of the first band edge, and the end of the band has two more for the start and end of that band) The normalized graph provided the numbers location of the start and stop of the signal normalized to Fs. For my signal I found those values to be 0.15 and 0.22. I went through a lot of trial and error to find the correct values for the frequencies that would result in the order that was desired. The firpmord command creates the parameters for the firpm command that will create the filter. The output signal was created which is the combined signal after it goes through the filter. The DTFT of the signal was taken and plotted to see if it is what I expected. I expect to see only the second signal coming out which I do see. I verified this by adding the graph of the DTFT of the second signal before it was added to the original signal and plotted it below my output plot and the two graphs are the same excluding a small bit of attenuation at the bottom of the spike.

Conclusion

This project used MATLAB to simulate and analyze SONAR signals using spectral analysis, DFT processing, and designing filters. The first task included finding the CTFT of a given signal and using that to select the Nyquist rate that would mitigate the aliasing when the signal is brought over to discrete time. This really helped show the difference between discrete and continuous time, as well as how frequency and time affect functions differently. The second task

showed how interfering signals can sometimes get lost in another signal and gave a range of amplitudes that would be sufficient for the signal processing to pick up and not get lost in the noise. Lastly, the third task showed how useful filters are at removing unwanted frequencies from a signal and how that affects the signal you want to keep. Overall, this project provides insight into a lot of topics discussed in class especially when converting signals between CTFT and DFT and DTFT.