Student Name:	**Aaron Dinesh**
Student ID Number:	**20332661**
Assessment Title:	**DSP Lab 03**
Lecturer (s):	**Naomi Harte**
Date Submitted	**13/10/23**

I hereby declare that this assessment submission is entirely my own work and that it has not been submitted as an exercise for a degree at this or any other university.

I hereby declare that I have not shared any part of this submission with any other student or person.

I have read and I understand the plagiarism provisions in the General Regulations of the University Calendar for the current year, found at: http://www.tcd.ie/calendar

I have also completed the Online Tutorial on avoiding plagiarism 'Ready, Steady, Write', located at http://tcd-ie.libguides.com/plagiarism/ready-steady-write

I am aware that the module coordinator reserves the right to submit my exam to Turnitin and may follow up with further actions if required should I be found to have breached College policy on plagiarism

Signed: 

DSP Lab 03

Disclaimer

This lab requires a number of hand calculations. ALL calculations will be placed in an appendix at the end of this report.

Ex 1.1.1

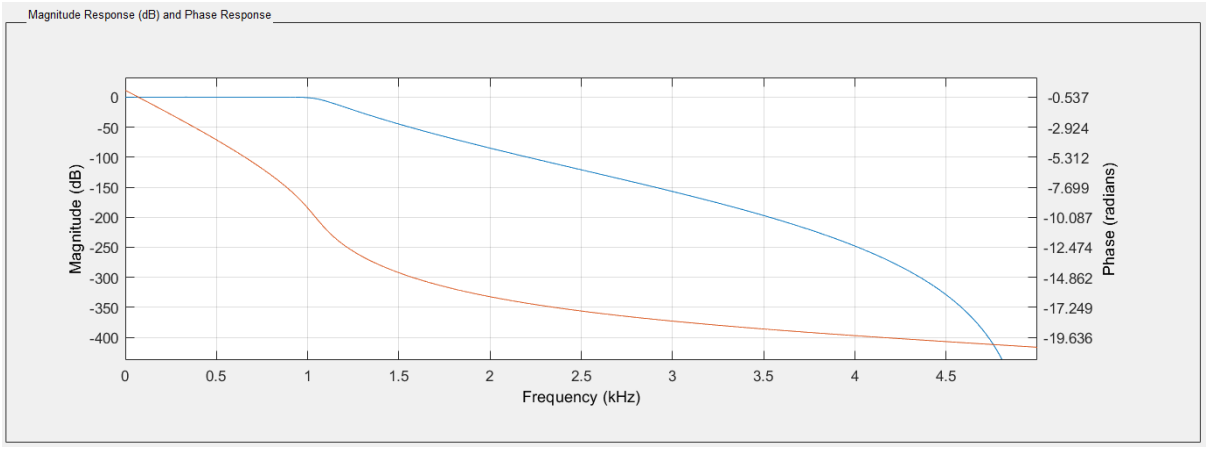

Figure 1 - Magnitude and Phase Response of Filter

After applying the specifications provided, the above magnitude and phase response plot for a Butterworth filter was generated, with a minimum order of 13. The phase is as expected for an IIR filter, as with IIR filters it is very hard to get a linear phase frequency response.

Ex 1.1.2

I varied the width of the transition region by changing the value of the stopband frequency. With a smaller stopband frequency, the magnitude response had to decay a lot faster, resulting in a steeper slope in the transition region. With a larger value for the stopband frequency the transition region had a much more gradual slope. I also noticed that as the transition region got narrower (approaching an ideal lowpass filter) the filter order had to increase. This corresponds to an increase in the number of coefficients that need to be stored to describe this filter, leading to a larger memory footprint. This is the trade of between the filter order and transition region. As the transition region becomes smaller the order has to increase to compensate and vice versa.

Ex 1.1.3

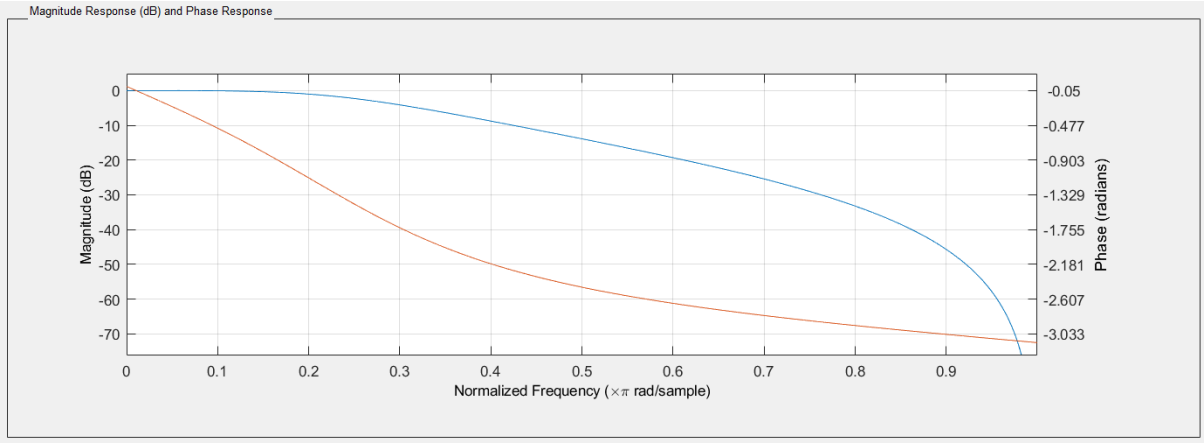

Figure 2 - Magnitude and Phase Response of designed filter

I used the Bilinear Transform method to design an appropriate Butterworth filter. From my calculations, I determined the minimum order to be 2. This was then confirmed by the Matlab filterDesigner tool.

Ex 1.1.4

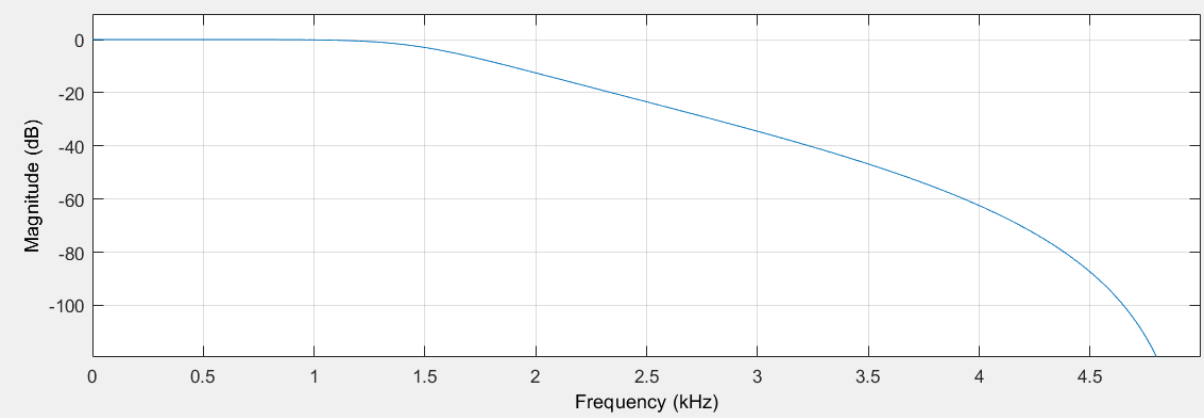

Figure 3 - Magnitude Response of 4th order Butterworth with 1.5kHz cutoff and 10kHz sampling frequency

Shown above is a 4th order Butterworth Lowpass filter with a cut-off frequency at 1.5kHz. There was also an additional requirement of a 40dB attenuation at 3kHz. An order 4 filter is not enough for this application since we don't achieve the required -40dB at 3kHz. This is assuming a sampling rate of 10kHz. From theoretical calculations I figured that a 7th order filter should be enough to match this specification.

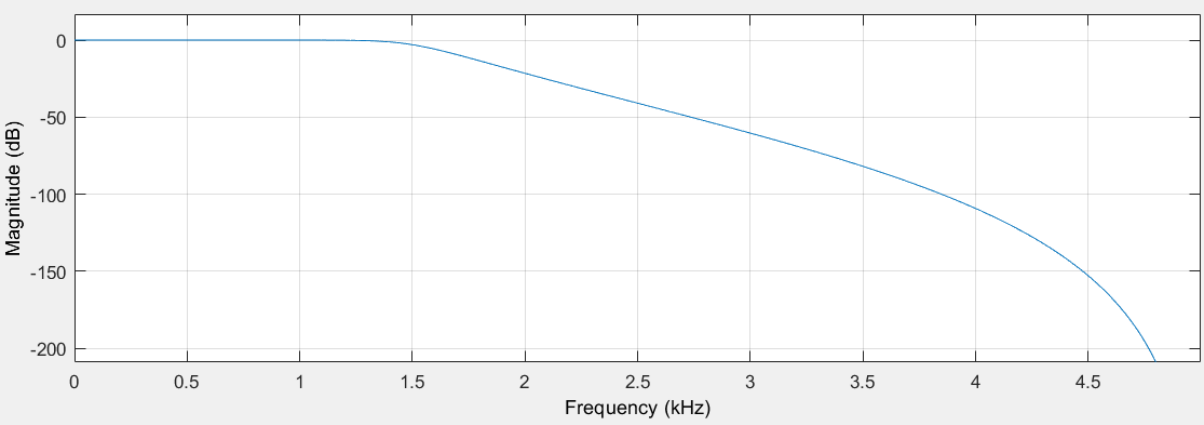

Figure 4 - Magnitude Response of 7th order Butterworth with 1.5kHz cutoff and 10kHz sampling frequency

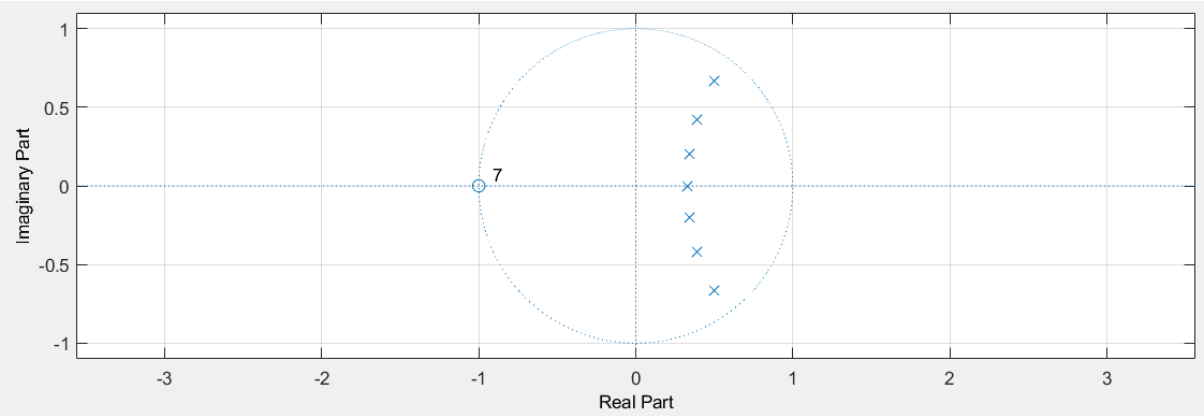

Figure 5 - Pole Zero plot of 7th order Butterworth with 1.5kHz cutoff and 10kHz sampling frequency

The above pole zero plot does not match up with the pole zero plot calculated when using the impulse invariance method. This is because the filterDesigner tool in MATLAB mainly makes use of the Bilinear Transform to discretise continuous time filters. The Bilinear Transform and the Impulse Invariance method produce different pole zero plots due to the difference in the mapping that both methods use to convert a continuous time filter to a discrete time filter.

Ex 1.1.5

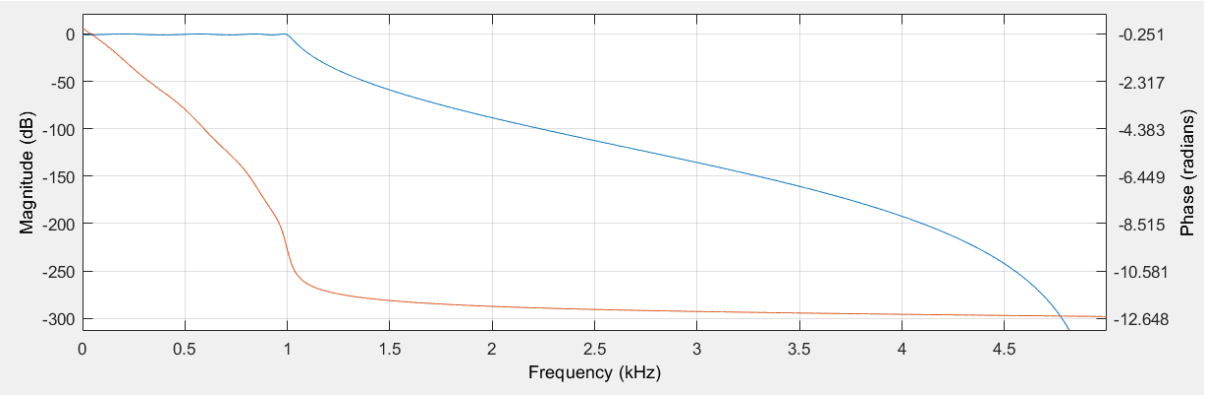

Figure 6 - 8th Order Chebyshev Type 1 Lowpass filter

The minimum order required to meet the specifications was 8.

Ex 1.1.7

Shown above is the magnitude and phase response of an 8th order Chebyshev Type 1 filter. Compared to the Butterworth filter in 1.1.1 we can see that in the passband, the Chebyshev filter is not maximally flat. Instead, we see some ripples in the passband. We can also see that compared to the Butterworth filter, it is not monotonically decreasing as well. We also notice that there seems to be a similar phase shift in the passband region. Where the Chebyshev filter really shines is its improvement in the order number. While the Butterworth required an order of 13 to meet the specifications, the Chebyshev filter only needed 8. This allows us to have a simpler filter.

Ex 1.2.1

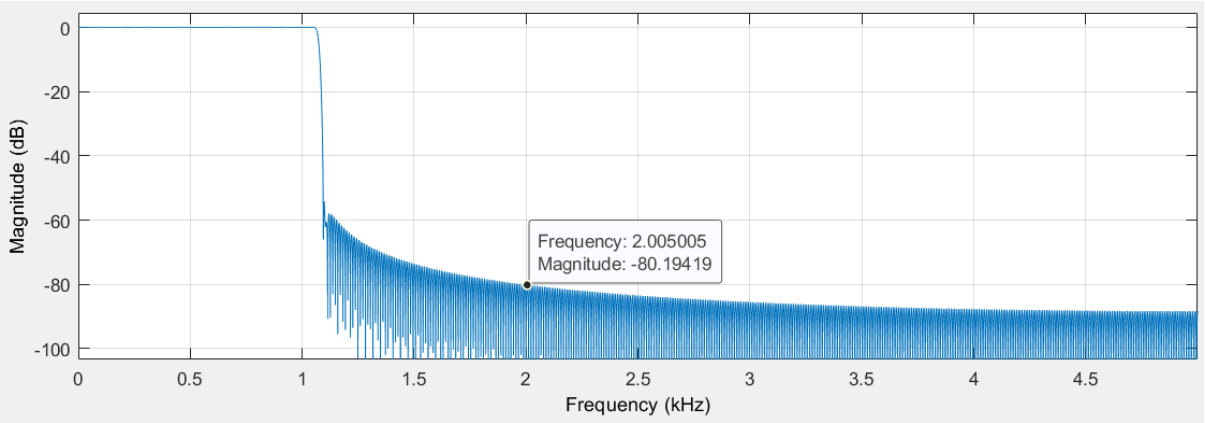

Figure 7 - Hamming Window Lowpass filter

In the graph above you can see a Hamming Window filter with a cutoff frequency of 1.075Khz and an order of 890. This filter fits the specifications outlined in Ex 1.1.1 of a 1dB passband

attenuation and an 80dB stopband attenuation at 2kHz. This continued past 2kHz. The sampling rate was chosen to be 10kHz.

As we can see from the phase plot (in orange), this FIR filter has linear phase compared to the nonlinear phase response we saw in the Butterworth filter. However, one downside to this is the ripple that we see present in the stopband.

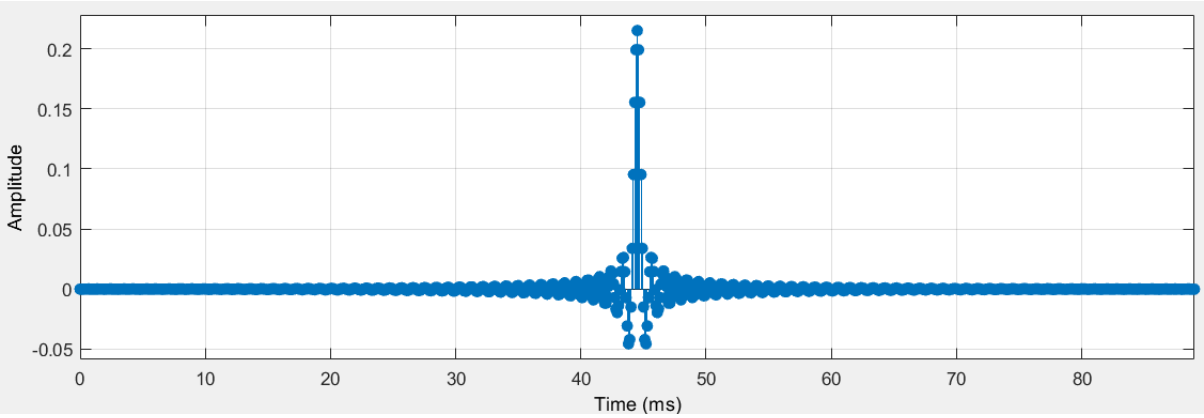

Figure 8 – Type 1 Filter Impulse Response

As we can see above, we have a Type 1 filter, since we have an even order length (890) and the impulse response is symmetric.

Ex 1.2.2

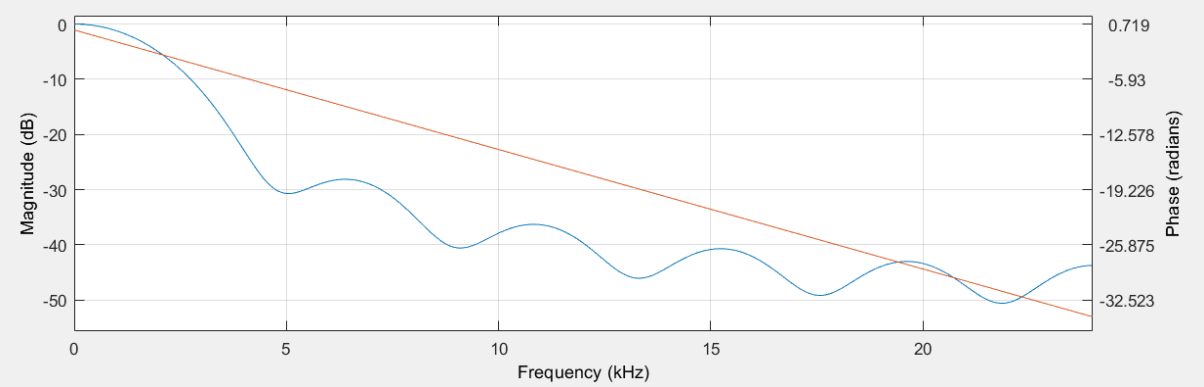

Figure 9 - Bartlett Window. Order 22. Sampling rate 47kHz. Cutoff 1.5kHz

The specs required a 40dB attenuation before 15kHz. I took this to mean that past 15kHz the filter was required to have a maximum allowed attenuation of 40dB. The above filter fits these specifications. I had to increase the filter order to 22 in order to keep an attenuation below 40dB past 15kHz. It should be noted that even though I specified an order of 22, MATLAB reported this filter to have an order of 21.

When compared with an ideal lowpass filter, with a cutoff at 15kHz, we see that the Bartlett window performs poorly, with major ripples in the stopband. However, this might be a trade

off we are willing to accept. If the frequencies we care about are below the cutoff frequency, then we don't really mind the ripples in the stop band. Also looking at the phase we see that this a linear filter across the entire range of inputs.

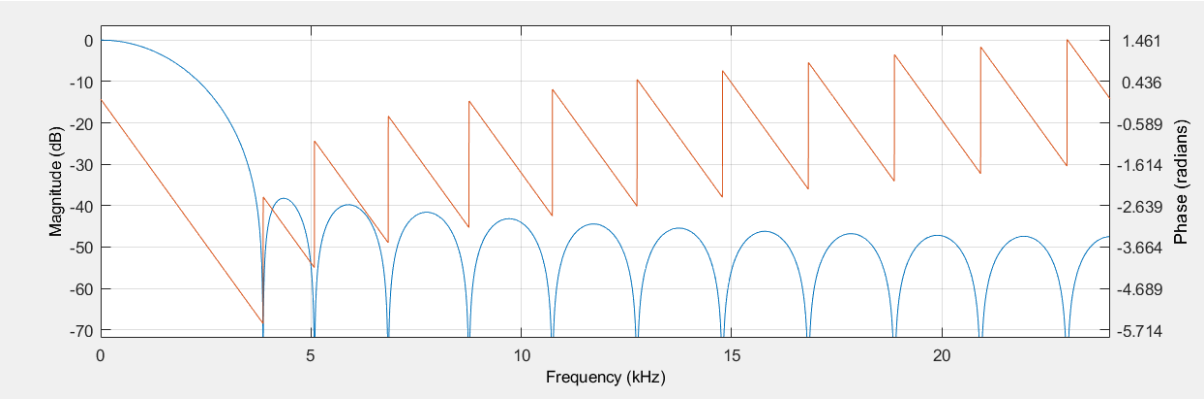

Figure 10 - Kaiser Window. Order 22. Beta parameter 3. Cutoff 1.5kHz. Sampling Frequency 48kHz

Shown above is the Kaiser window on a lowpass filter. I had to set the beta parameter to be 3 in order to achieve a 40dB attenuation before 15kHz and a maximum of 40dB attenuation past 15kHz. It is also important to note the linear phase response in the passband region. However, compared to the Bartlett window it has a much steeper drop off.

Ex 1.2.3

Assuming a 5kHz transition region the order required for the filter can be approximated with

$$N \approx \frac{A_{dB} F_s}{22\Delta F}$$

$$N \approx \frac{40 \times 50}{22 \times 5}$$

$$N \approx 18.182 = 19$$

Using this order number, I created a Kaiser Window lowpass filter with a cutoff frequency of 10kHz and a sampling frequency of 50kHz. The magnitude and phase response are shown in the graph below.

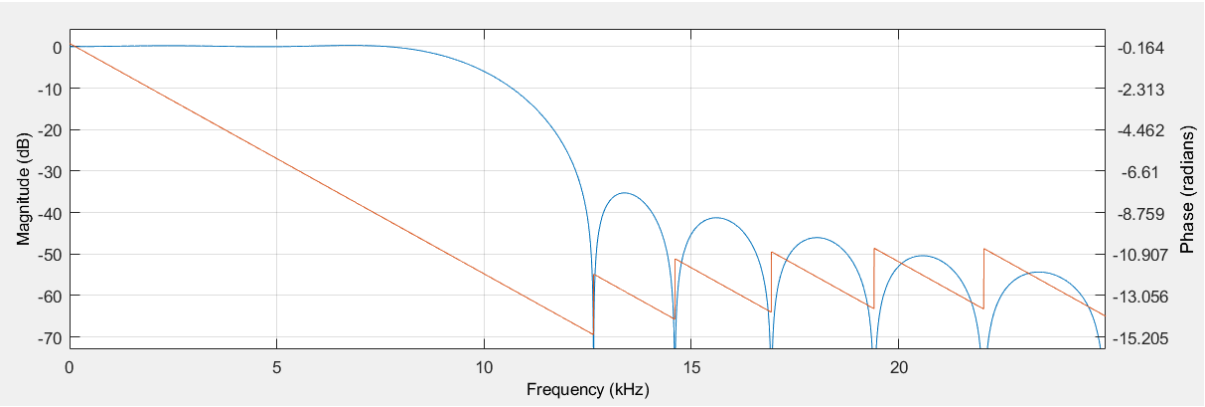

Figure 11 - Kaiser window. Beta parameter 2.5. Order 19. Cutoff 10kHz. Sampling Frequency 15kHz.

We can see from this example that the approximation for the order number is quite a good approximation when starting to design a filter. This can then be further finetuned to achieve an optimal filter.

Ex 1.3.2

This exercise required us to design the specifications for a filter that would be able to get rid of a 3.5kHz signal that corrupted the original speech. I generated a couple of filters and measured their performance by calculating the MSE between the filtered and the uncorrupted signal.

Filter 1

I designed an Equiripple Filter with the following specifications:

- Density Factor: 20
- Order: 40
- Sampling Frequency: 8kHz
- Passband Edge Frequency: 3kHz
- Stopband Frequency: 3.5kHz
- A_{pass}: 1dB
- A_{stop}: 80dB

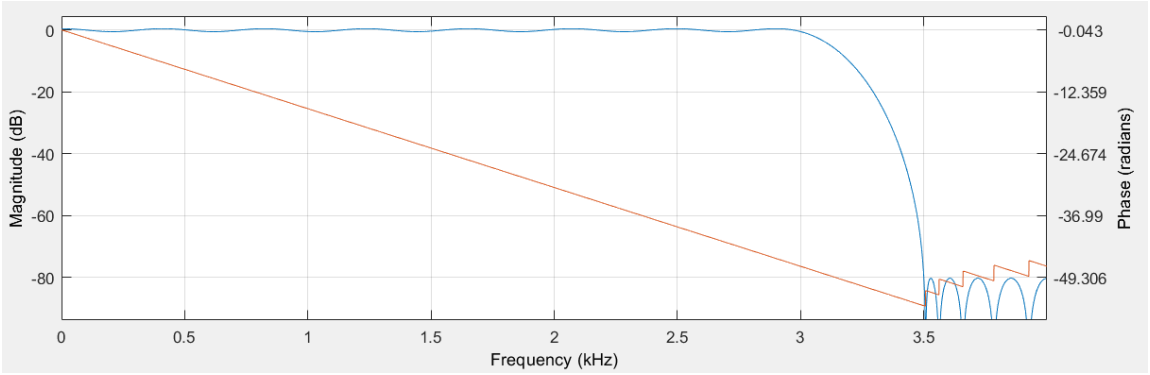

Figure 12 - Magnitude and Phase Response of Filter 1

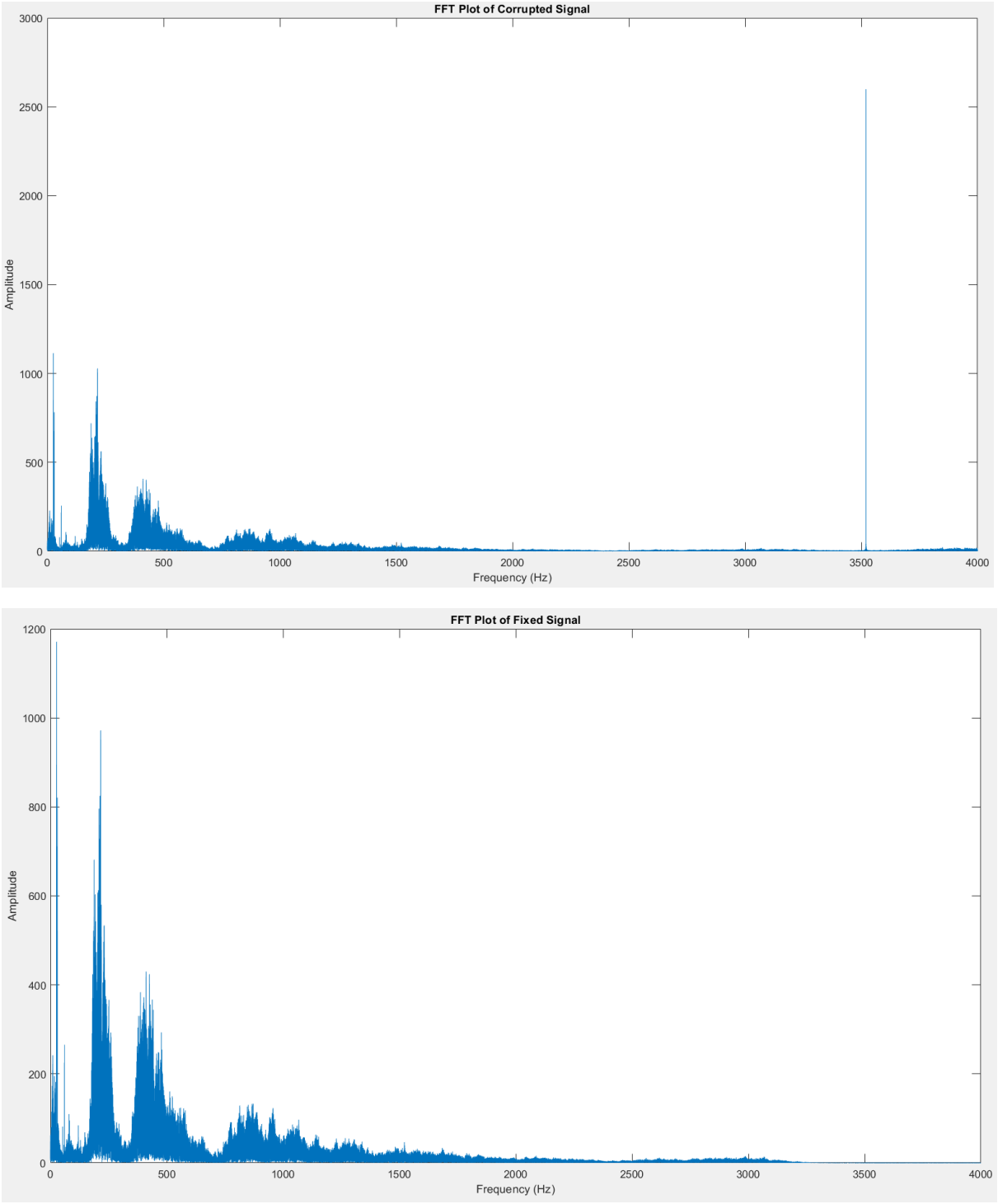

Figure 13 - FFT Plot Showing Corrupted (a) and Filtered Speech (b)

The MSE for this filter was 0.0049. For all the following filters, I will specify the relevant parameters, the magnitude and phase responses, and the MSE between the filtered and the original signal.

Filter 2

- Type: Equiripple Lowpass
- Density Factor: 20
- Order: 19
- Sampling Frequency: 8kHz
- Passband Edge Frequency: 2kHz
- Stopband Frequency: 3kHz
- A_{pass}: 1dB
- A_{stop}: 80dB
- MSE: 0.0045

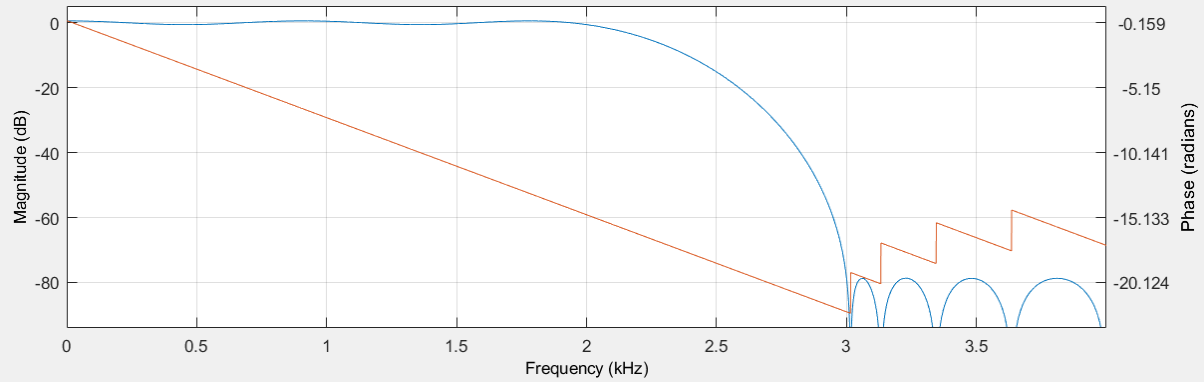

Figure 14 - Filter 2 Magnitude and Phase Response

Filter 3

- Type: Equiripple Lowpass
- Density Factor: 20
- Order: 12
- Sampling Frequency: 8kHz
- Passband Edge Frequency: 2kHz
- Stopband Frequency: 3.5kHz
- A_{pass}: 1dB
- A_{stop}: 80dB
- MSE: 0.0031

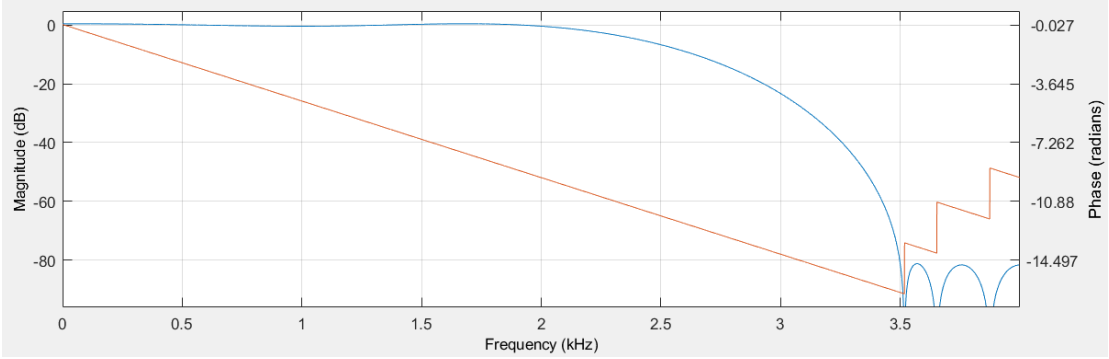

Figure 15 - Filter 3 Magnitude and Phase Response

Filter 4

- Type: Equiripple Lowpass
- Density Factor: 20
- Order: 9
- Sampling Frequency: 8kHz
- Passband Edge Frequency: 2kHz
- Stopband Frequency: 3.5kHz
- A_{pass}: 1dB
- A_{stop}: 60dB
- MSE: 0.0022

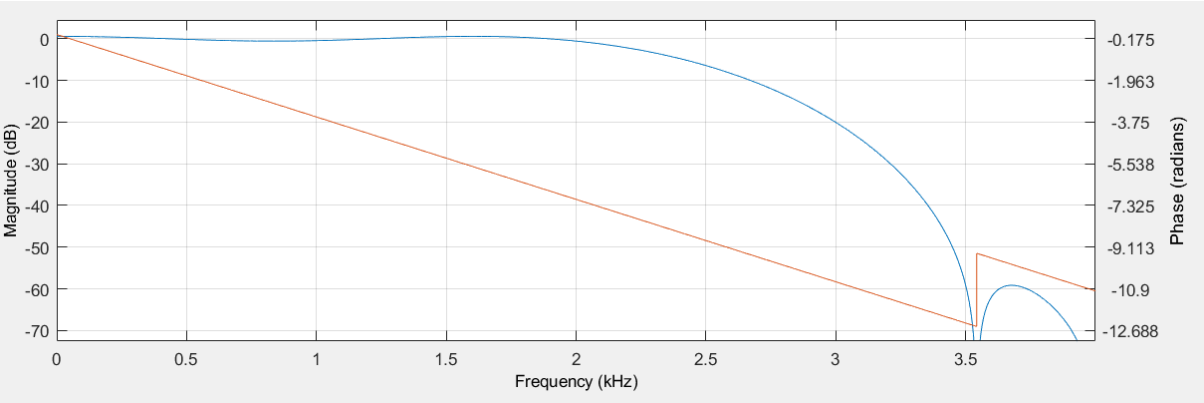

Figure 16 - Filter 4 Magnitude and Phase Response

Filter 5

- Type: Generalised Equiripple Lowpass
- Density Factor: 20
- Phase: Minimum
- Order: 8
- Sampling Frequency: 8kHz
- Passband Edge Frequency: 2kHz
- Stopband Frequency: 3.462kHz
- A_{pass}: 1dB
- A_{stop}: 60dB
- MSE: 0.0022

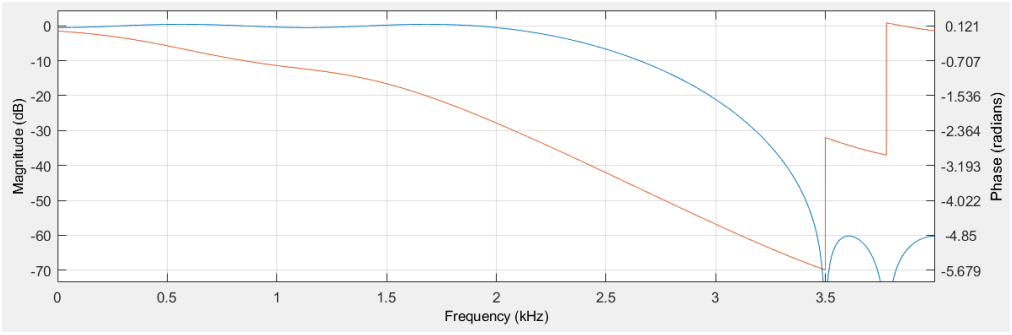

Figure 17 - Filter 5 Magnitude and Phase Response

Filter 6

- Type: Butterworth IIR Filter
- Order: 5
- Sampling Frequency: 8kHz
- Passband Edge Frequency: 2kHz
- Stopband Frequency: 3.5 kHz
- A_{pass}: 1dB
- A_{stop}: 60dB
- MSE: 0.0049

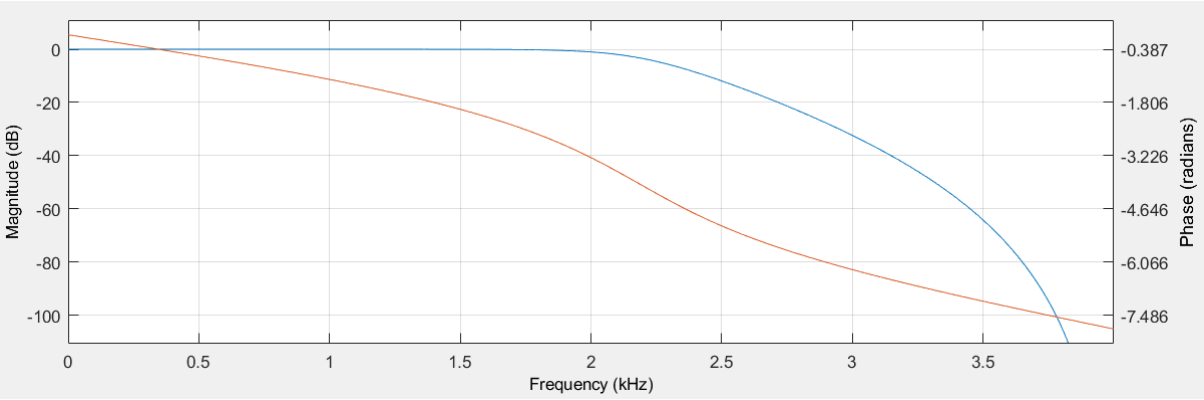

Figure 18 - Filter 6 Magnitude and Phase Response

For most of the filters I designed an Equiripple filter in MATLAB since this makes use of the Parks-McClellan Method for filter design which often provides lower order filters than Impulse Invariance or Bilinear Transform design methods. Then from looking at the FFT plot of the original and the corrupted signal I saw that the tone was centred around 3.5kHz. So, I could relax my stopband frequency to be around that frequency. This allowed me to drop the model order while achieving a better MSE. For filter 4, I realised that an 80dB attenuation was extremely large and I could probably get away with a 60dB attenuation. When I implemented this change, the filter order dropped from 12 to 9. For filter 5, I switched from using an Equiripple filter to a Generalized Equiripple with minimum phase. I also slightly changed the stopband frequency to be 3.462kHz. This allowed me to drop the filter order to 8. However, it is important to note the non-linear phase response in filter 5. Depending on the application, and the magnitude of phase change, this nonlinearity can have significant impact on the filtered signal. But when listing to the resulting audio I didn't notice much difference between the linear phase filters and filter 5. For filter 6, I decided to explore the realm of IIR filters. I decided to design a minimum order Butterworth filter with a 60dB attenuation at 3.5kHz. After inputting the specifications into the filterDesigner tool in MATLAB, it generated an order 5 filter. However, one important thing to note is the significant non-linearity of this filter. Also, considering that IIR filters are less stable than FIR filters, I wouldn't recommend using an IIR filter over an FIR one, even though the model order is lower.

The best performing filter for this case would be filter 5, since it strikes a balance between the trade-off between filter order, MSE and complexity.

Ex 1.3.3

This required us to corrupt the original signal with note 40, which corresponds to a frequency of around 82Hz. This can be seen by the large spike in the fft of the corrupted signal.

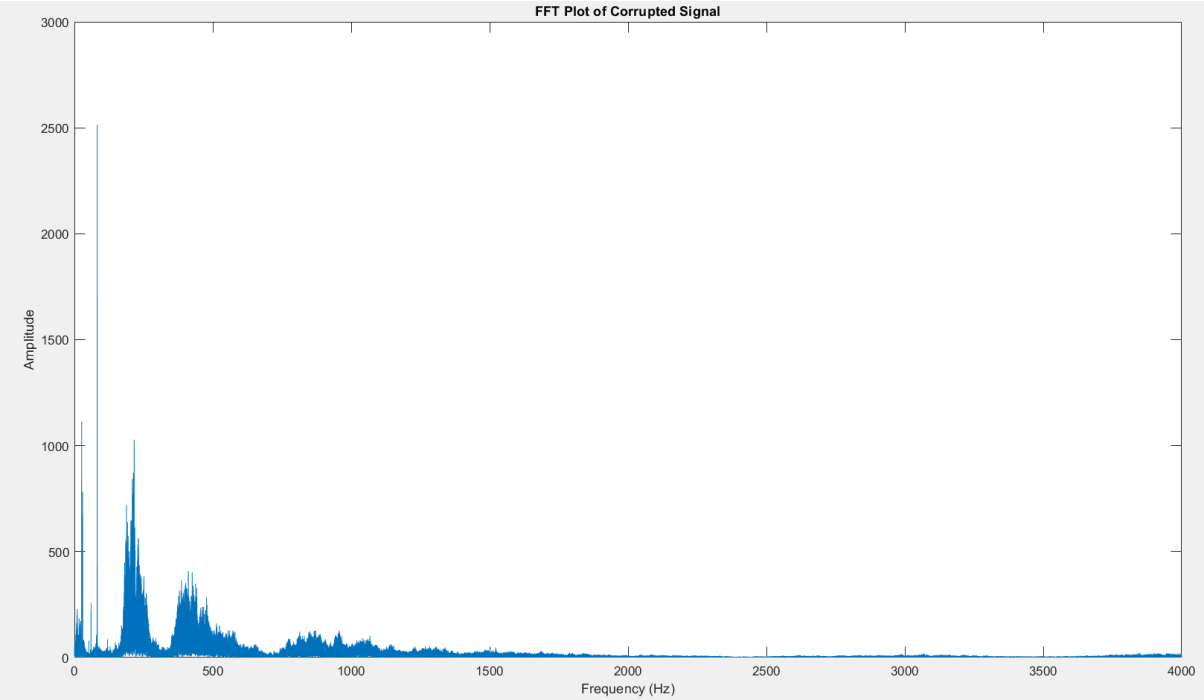

Figure 19 - FFT of signal corrupted with note 40

Since this signal is within the bandwidth of the signal, the appropriate filter to use would probably be a bandstop filter, since then we could only remove the frequency, we need and leave the rest of the frequencies unchanged. When trying to design an appropriate bandstop filter, I noticed that to get the performance I needed, the filter order would be quite high, on the order of 1000s. This isn't feasible from a practical standpoint. After looking at the distribution of the frequencies, I noticed that most of the power in the frequency range was past 100Hz. Which makes sense when considering human speech. A female voice on average only covers the frequencies from 350Hz to 17kHz, with male voices going as low as 100Hz. So, I could use a highpass filter with a stopband frequency of 100Hz. This would not only get rid of the noise while leaving the majority of the signal unchanged, but it would also allow me to drastically drop the filter order (< 100).

After many iterations in the filterDesginer tool, I converged on the following filter specifications:

- Type: Generalised Equiripple Highpass
- Label: HighpassFilter8
- Density Factor: 20
- Phase: Minimum

- Order: 10 (Minimum Order)
- Sampling Frequency: 8kHz
- Passband Edge Frequency: 760Hz
- Stopband Frequency: 120 Hz
- A_{pass}: 1dB
- A_{stop}: 30dB
- MSE: 0.0024

This filter had the following Magnitude and Phase response plot:

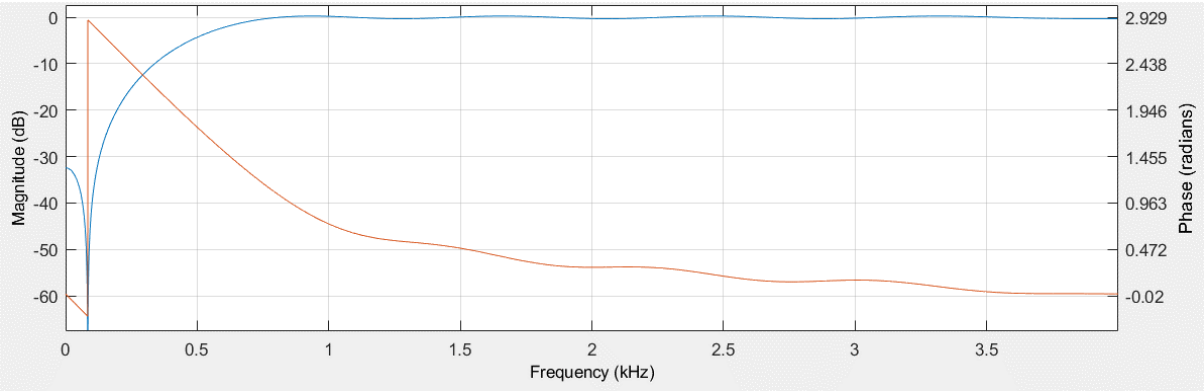

Figure 20 - Magnitude and Phase response of HighpassFilter8 highpass filter

This filter has a number of interesting properties. Since we know exactly where the noise is we can tailor the filter to specifically remove that frequency. Here I exploited the ripple effect that is characteristic of this filter to have a sharp dip exactly where the noise signal was contained, while allowing the other frequencies to pass. Since the main part of the speech in this audio clip lies past 150Hz, they also largely pass through unaltered.

This filter also has the effect of clearing up the audio. In the original audio I heard a lot of low frequency noise that made the speech hard to understand. As soon as this noise was filtered out, the speech became much clearer. The filtered audio is labelled "Fixed_ER_Highpass_Filter_8.wav".

Ex 1.3.4

This part of the lab required us to superimpose two tones on the signal, one lower than note 40 and one higher than note 60. So, for this I chose note 20 as my low note and note 105 as the high note. The corrupted file is labelled as "OSR_20_105_Corrupted.wav". After graphing the FFT plot of the signal (shown below), I noticed it was possible to just chain the two filters I had already created in the previous two exercises instead of having to create a new bandpass filter for this specific case. The resulting audio is labelled "Fixed_ER_Dual_Filter.wav".

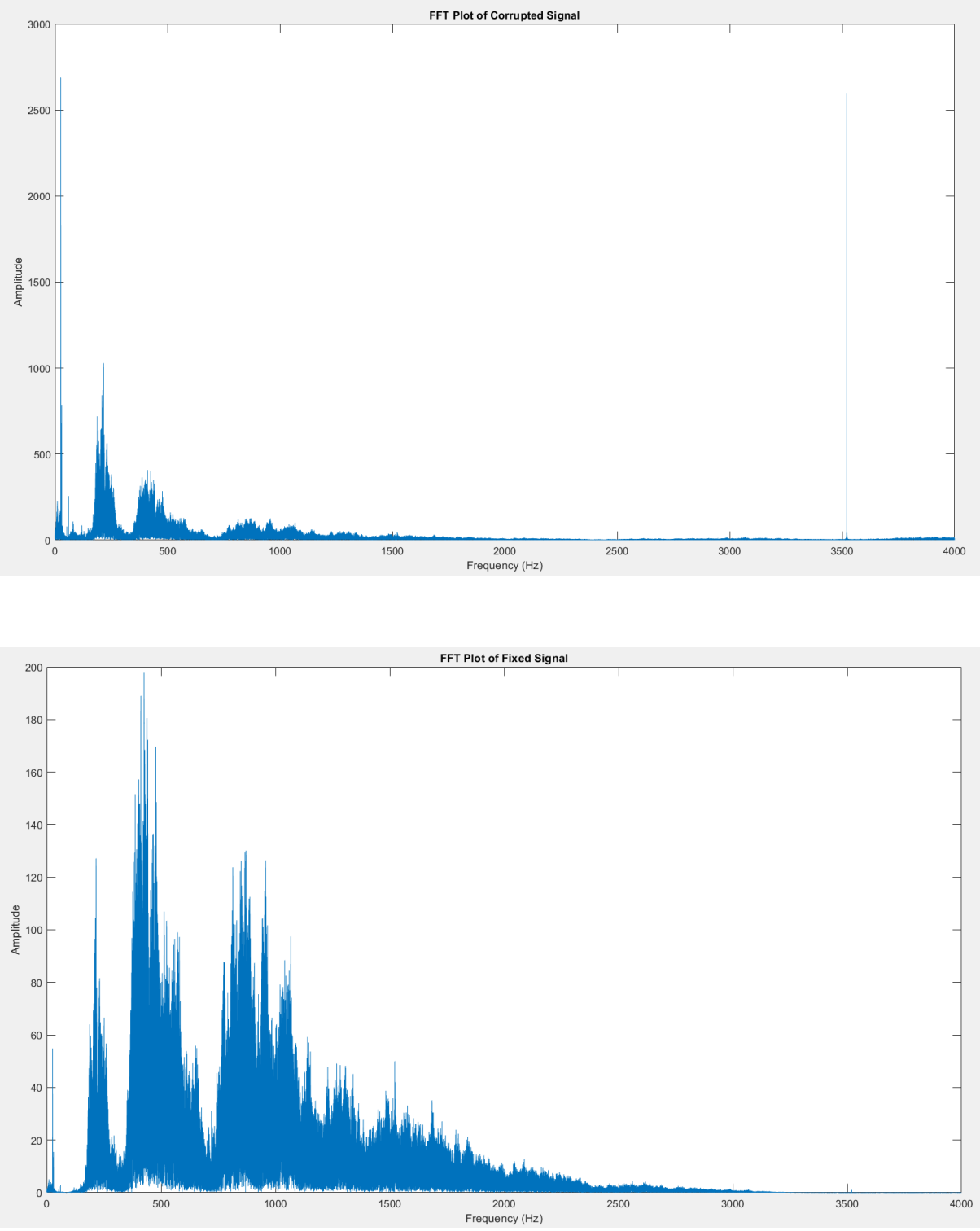

Figure 21 - FFT Plots of Corrupted (a) and Fixed (b) audio signals

Q1.1.1 Butterworth Lowpass

$$0.89 \leq |H(e^{j\omega})| \leq 1, \quad 0 \leq \omega \leq 0.2\pi$$
$$|H(e^{j\omega})| \leq 0.18, \quad 0.6\pi \leq \omega \leq \pi$$

Assume that $T_d = 2$ implying $\Omega = \tan\left(\frac{\omega}{2}\right)$

Prewarp the edge frequencies:

$$\Omega_1 = \tan\left(\frac{0.2\pi}{2}\right) = \tan(0.1\pi) \qquad \Omega_2 = \tan\left(\frac{0.6\pi}{2}\right) = \tan(0.3\pi)$$

Determine CT transfer function:

$$|H_c(j\Omega)|^2 = \frac{1}{1 + \left(\frac{\Omega}{\Omega_c}\right)^{2N}}$$

$$0.89^2 = \frac{1}{1 + \left(\frac{\tan(0.1\pi)}{\Omega_c}\right)^{2N}}$$

$$\left(\frac{\tan(0.1\pi)}{\Omega_c}\right)^{2N} = \left(\frac{1}{0.89^2}\right) - 1 \qquad (1)$$

$$0.18^2 = \frac{1}{1 + \left(\frac{\tan(0.3\pi)}{\Omega_c}\right)^{2N}}$$

$$\left(\frac{\tan(0.3\pi)}{\Omega_c}\right)^{2N} = \left(\frac{1}{0.18^2}\right) - 1 \qquad (2)$$

$\frac{(1)}{(2)}$

$$\left[\frac{\tan(0.1\pi)}{\tan(0.3\pi)}\right]^{2N} = 8.788 \times 10^{-3}$$

$$2N = 3.2795$$
$$N = 1.6397 \Rightarrow N = 2$$

Ex 1.1.4 Butterworth filter of order 4 and cut off frequency @ 1.5kHz. Also needs a 40dB attenuation at 3kHz

Assume $T_d = 1$ so $\Omega = \omega$

We want the attenuation to be 40dB at 3kHz with a cutoff of 1kHz

Converting dB to linear:

$$40 = 20\log(x_1)$$

$$x_1 = 10^{-4}$$

$$10^{-4} = \frac{1}{1 + \left(\frac{3\times10^3\times2\pi}{1.5\times10^3\times2\pi}\right)^{2N}}$$

$$10^{-4} = \frac{1}{1 + 2^{2N}}$$

$$2^{2N} = \frac{1}{10^{-4}} - 1$$

$$N = \frac{\ln\left(\frac{1}{10^{-4}} - 1\right)}{2\ln(2)} = 6.64$$

$$\Rightarrow 7$$

We should then expect 14 poles evenly spread around a circle of radius $2\pi \times 3000$ at angles of θ_k:

$$\theta_k = \frac{(N+1+2k)\pi}{2N} = \frac{2(4+k)\pi}{7}$$

for $k = 0, 1, \cdots, 13$