University of Tehran
College of Engineering
School of Electrical and Computer Engineering

Digital Signal Processing

Dr.Akhaee

Computer Assignment 2

Soroush Mesforush Mashhad

SN:810198472

Tir 01

Contents

Abstract

In the first section we get familiar with image filtering in MATLAB, we study the effect of different filters such as **Average**,**Laplacian**,**Sobel** etc.

In the second section we attempt to reconstruct a signal from a sample with noise and echo, we plan to do this by designing an appropriate filter, we use concepts such as autocorrelation in this section, we also briefly use fdatool.

In the third section we attempt to perform bleep censoring, we do this by finding the normalized cross-correlation between two audio files and removing the desired word with the help of a $1000Hz$ Sine wave.

In the last section we get familiar with audio filtering with Matlab tools, we have a noisy sound and first remove part of the noise with our uniquely designed bandstop filter(we do this with fdatool) then we go on to apply two different filters(in two separate parts) on the filtered voice to study the new effects.

1 Question 1

1.1 Part A

The original photo is as follows.

Figure 1: The original picture

First of all, I'd like to to express that I've used the original values recommended by MATHWORKS for the initial appliance of all these filters.

We go on to load the original image in MATLAB with the help of **imread** then proceed to apply the desired filters accordingly.

1.1.1 Gaussian filter

We know that in signal processing a Gaussian filter is implied to be a filter whose impulse is a Gaussian function or an approximation of a Gaussian function. The shape of the impulse response of a typical Gaussian filter is included below.

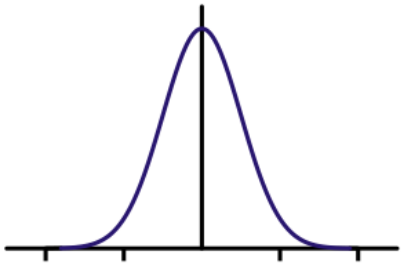

Figure 2: Gaussian filter impulse response

This filter in MATLAB is as follows:

h = fspecial('gaussian',hsize,sigma)

This implementations gives us a Gaussian lowpass filter with the size **hsize**, this filter is both rotational and symmetric, hsize is either a vector implicating the number of rows and columns, or a scalar which is a square matrix, sigma is also the standard deviation which is always positive.

All in all, the Gaussian filter perfoms something known as the Gaussian blur, In this phenomenon the pixels closest to the center of our kernel are given more weight, in other words averaging is performed, this average is the new value for the pixel in the filtered image, if the kernel is larger or in other words, hsize gets bigger the blur shall increase, also if we increase the standard deviation aka sigma the blur shall increas, this is tested in part B.

This filter is commonly used for the removal of detail and noise.

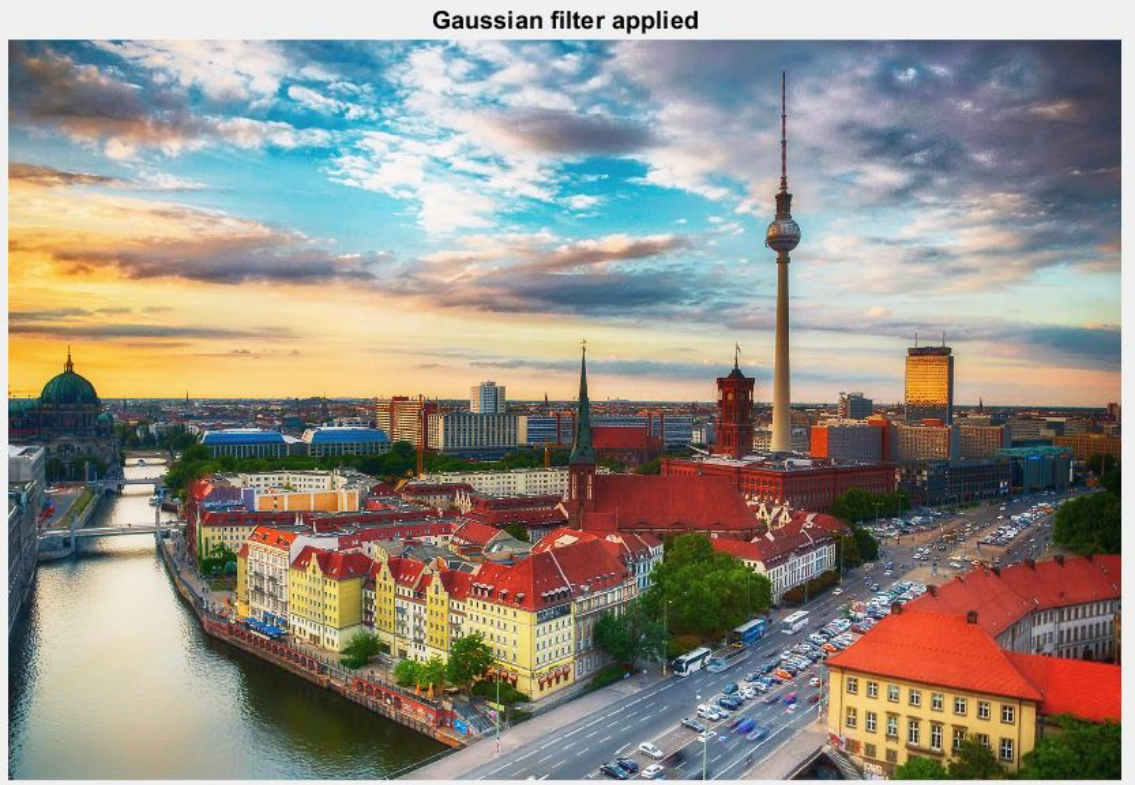

Figure 3: Gaussian filter applied

1.1.2 Sobel filter

This filter is most commonly used for edge detection, it is named after Irwin Sobel,technically speaking, this filter is a discrete differentiation operator, it calculates the gradient of image intensity at each pixel in our image and finds the direction of the largest increase from light to dark and the rate in that direction.

This filter in MATLAB is as follows:

h = fspecial('sobel')

This implementation gives us a filter that emphasizes the horizontal edges using a smoothing effect as explained above with the vertical gradient approximation.

This filter is most commonly appreciated because of its simplicity.

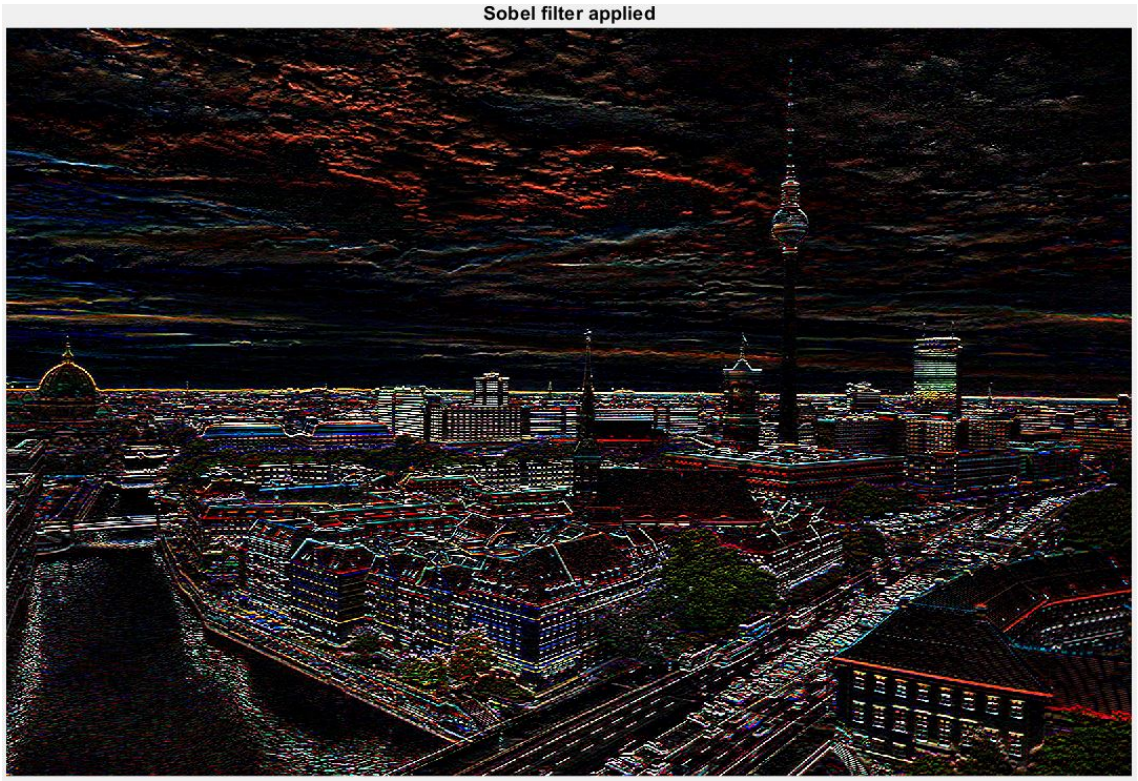

Figure 4: Sobel filter applied

1.1.3 Disk filter

This filter in MATLAB is as follows:

$$\text{h} = \text{fspecial('disk',radius)}$$

This filter returns a circular averaging filter also known as a pillbox, this filter is fitted inside a square matrix with a side of $2 \times radius + 1$.

It is safe to say that this filter is known by its radius, this radius determines the circle on which the average of the pixels of the image are calculated and replaced with the original pixels, this filter performs blurring and this blurring increases with the radius.

Figure 5: Disk filter applied

1.1.4 Unsharp filter

This filter, is a sharpening operator and performs an act called unsharped masking, it attempts to enhance edges by using a blurred or unsharp negative image to create a mask of the original image, this mask is combined with the original image and creates an image less blurry than the original, though this filter makes the results clearer, it may differ in detail and accuracy from the original image.

This filter in MATLAB is as follows:

$$\text{h} = \text{fspecial('unsharp',alpha)}$$

This implementation creates a contrast enhancement filter. The parameter alpha controls the shape of the Laplacian filter used to construct the Unsharp filter, this value is between 0 to 1.

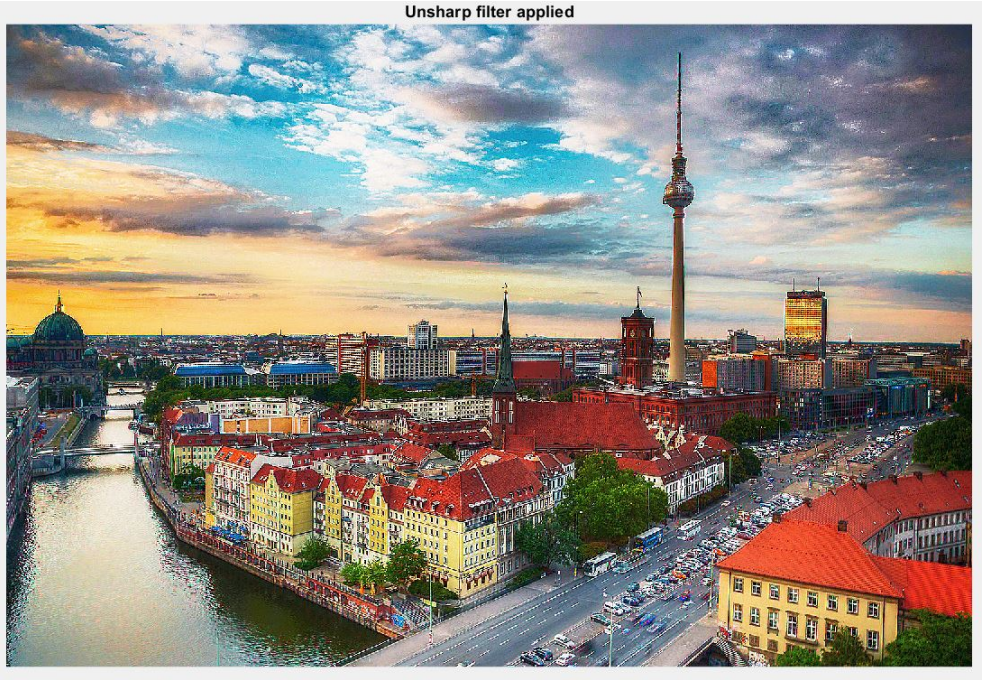

Figure 6: Unsharp filter applied

1.1.5 Average filter

This filter is used for smoothing images, it attempts to do this by decreasing the amount of intensity variation between pixels near eachother, this is done by replacing each pixel with the average value of the neighbouring pixels.

This filter in MATLAB is as follows:

$$\text{h} = \text{fspecial('average',hsize)}$$

This implementation returns an averaging filter with size of hsize, hsize may be a scalar or a vector. By increasing hsize the amount of blurring increases which is tested in part B.

Figure 7: Average filter applied

1.1.6 Laplacian filter

This filter is an edge detector and an edge sharpening filter, this filter computes the second derivative of the image to measure the rate of the change of the first derivative, this action determines if a change in an adjacent pixel comes from an edge or a continuous progression.

This filter in MATLAB is as follows:

$$\text{h} = \text{fspecial('laplacian',alpha)}$$

This implementation returns a Laplacian filter approximating the shape of the two-dimensional Laplacian operator, alpha controls the shape of the Laplacian.

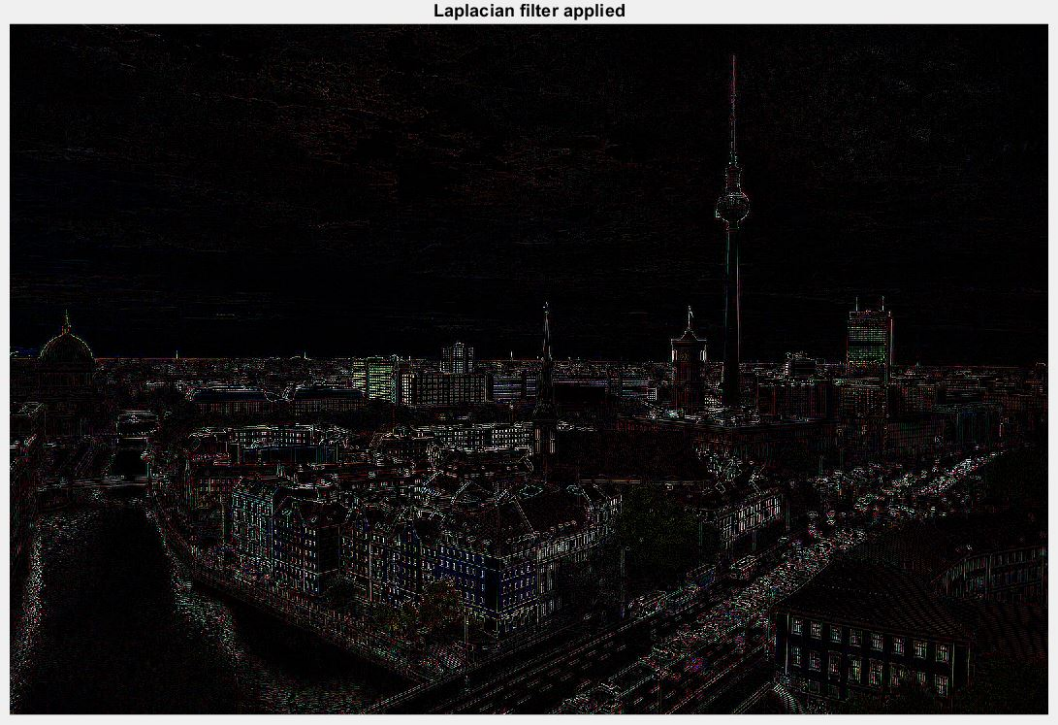

Figure 8: Laplacian filter applied

1.2 Part B

Here we observe the effect of changing the parameters in different filters.

1.2.1 Gaussian filter

We expect that when we increase hsize and sigma for the blur to increase, hence we perform the simulation.

Figure 9: Gaussian filter applied with different parameters

As we observe our deduction is correct.

1.2.2 Sobel filter

This filter has no special parameters to change.

1.2.3 Disk filter

This filter in MATLAB is as follows:

$$\text{h} = \text{fspecial('disk',radius)}$$

We expect the blur to increase if we increase the radius, hence we perform the simulation.

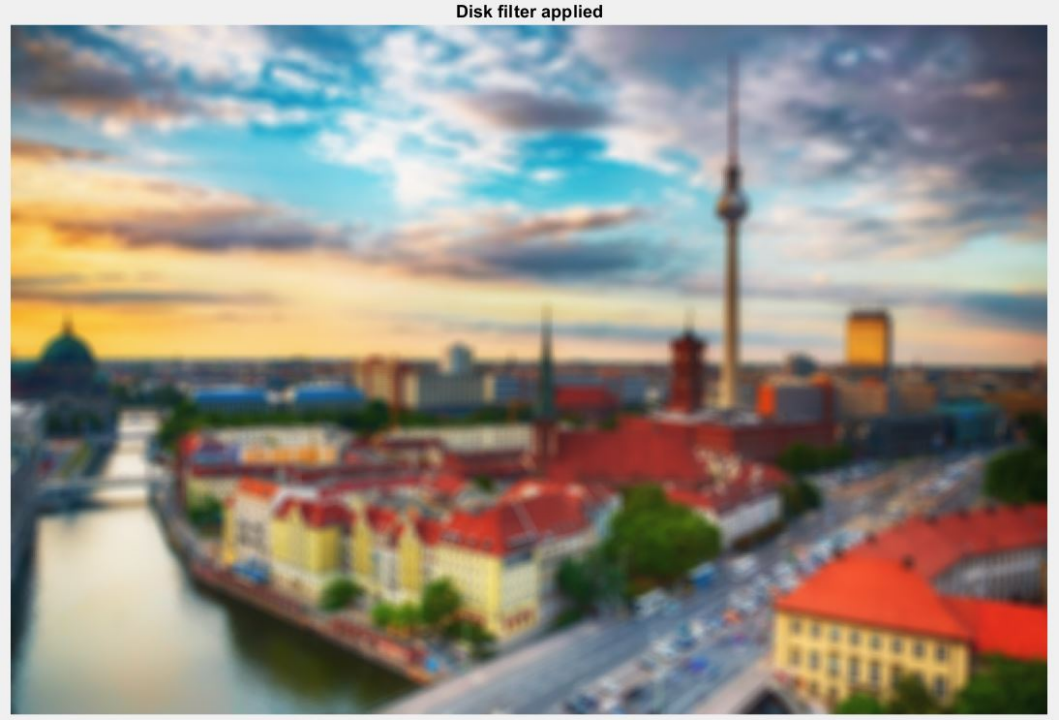

Figure 10: Disk filter applied with different parameters

As we observe our deduction is correct.

1.2.4 Unsharp filter

We expect that by increasing alpha, for the detail to reduce, hence we perform the simulation.

Figure 11: Unsharp filter applied with different parameters

As we observe our deduction is correct.

1.2.5 Average filter

We expect the blur to increase when increase hsize, hence we perform the simulation.

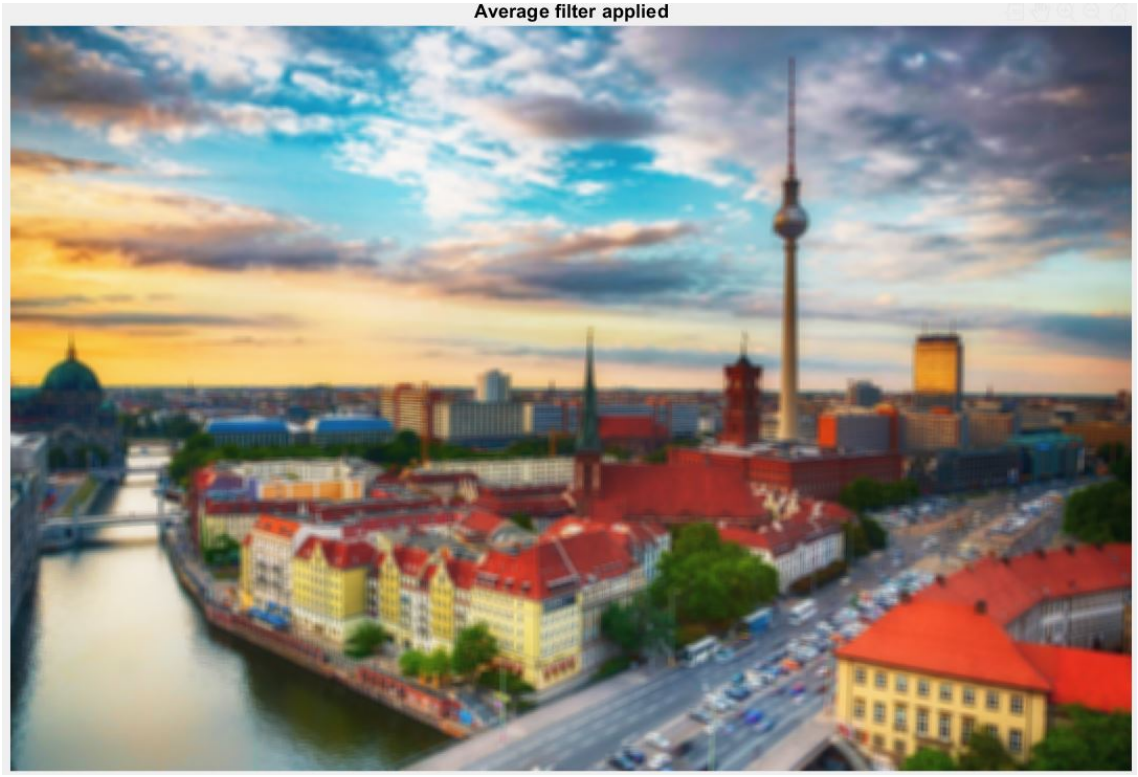

Figure 12: Average filter applied with different parameters.

As we observe our deduction is correct.

1.2.6 Laplacian filter

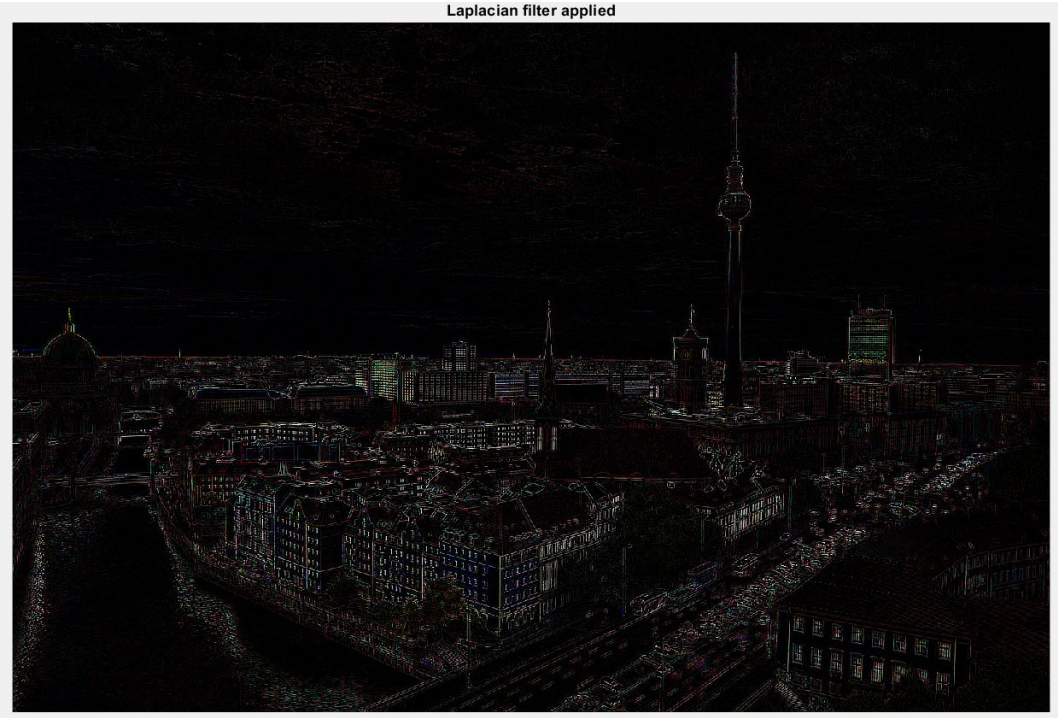

Figure 13: Laplacian filter applied

1.3 Part C

Here we begin by grayscaling the images with each filter applied to it, then we go on to plot the spectrum.

1.3.1 Gaussian filter

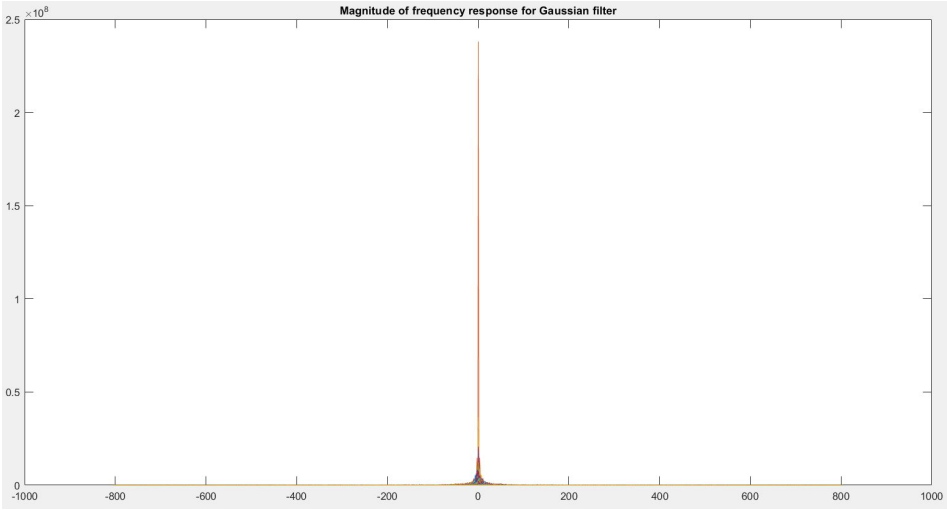

Figure 14: Gaussian filter spectrum

1.3.2 Sobel filter

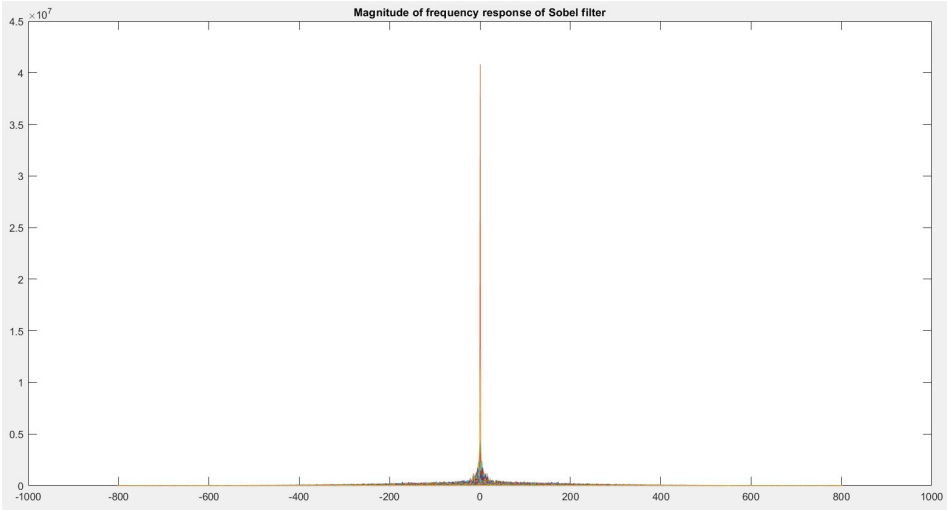

Figure 15: Sobel filter spectrum

1.3.3 Disk filter

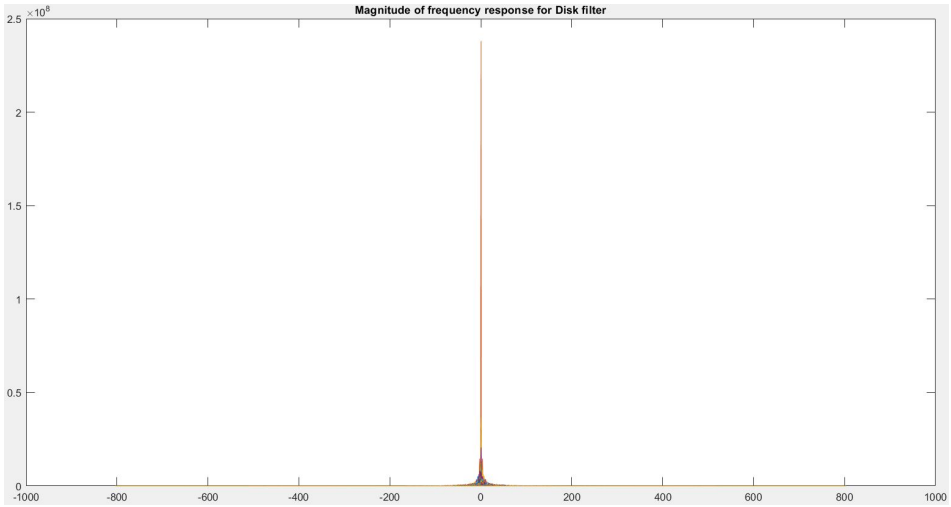

Figure 16: Disk filter spectrum

1.3.4 Unsharp filter

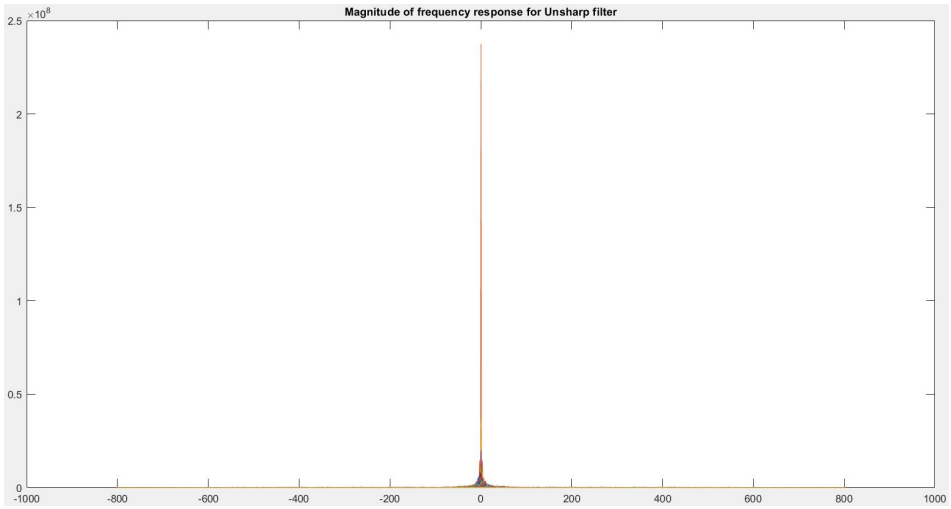

Figure 17: Unsharp filter spectrum

1.3.5 Average filter

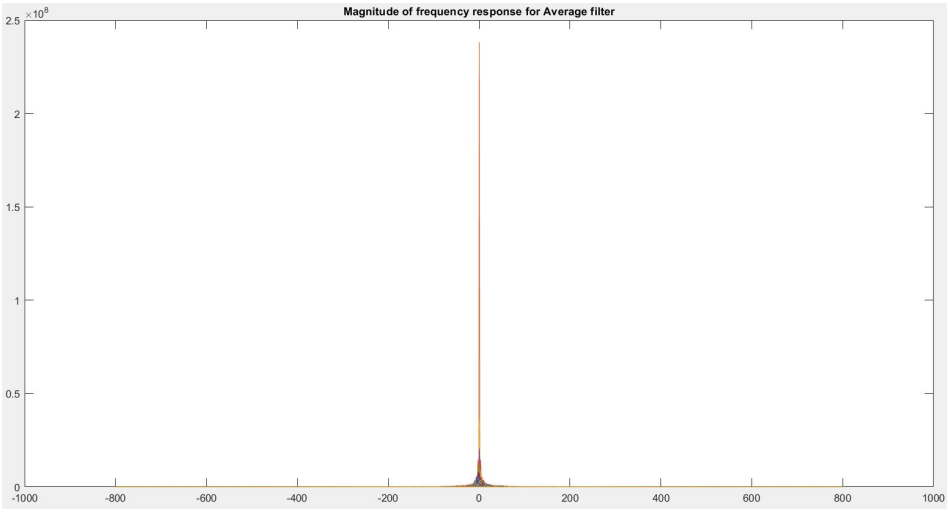

Figure 18: Average filter spectrum

1.3.6 Laplacian filter

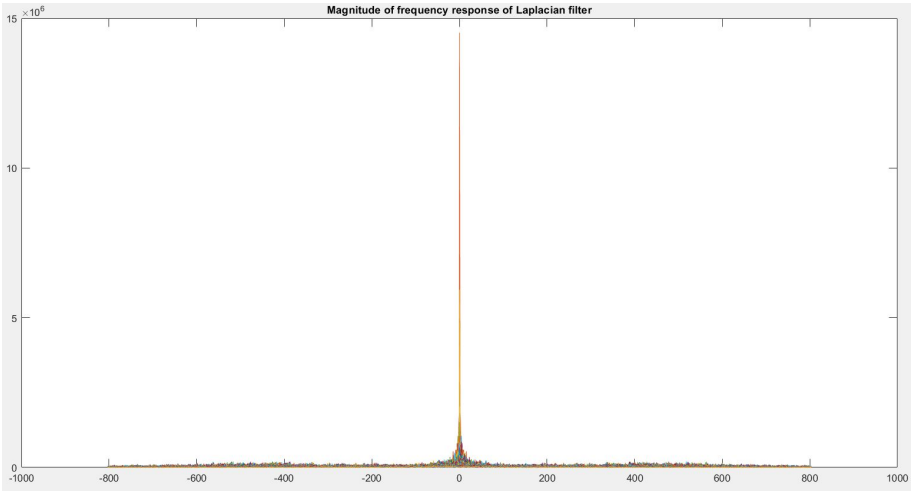

Figure 19: Laplacian filter spectrum

2 Question 2

In this part, we have a noisy signal with echo, we want to design an appropriate filter to be able to extract the original signal.

2.1 Part A

Using audioread, we save $y.wav$ in MATLAB and the sampling frequency is as follows.

$$F_s = 11025Hz$$

2.2 Part B

With the help of fft and fftshift, the magnitude and phase of the audio is given as follows.

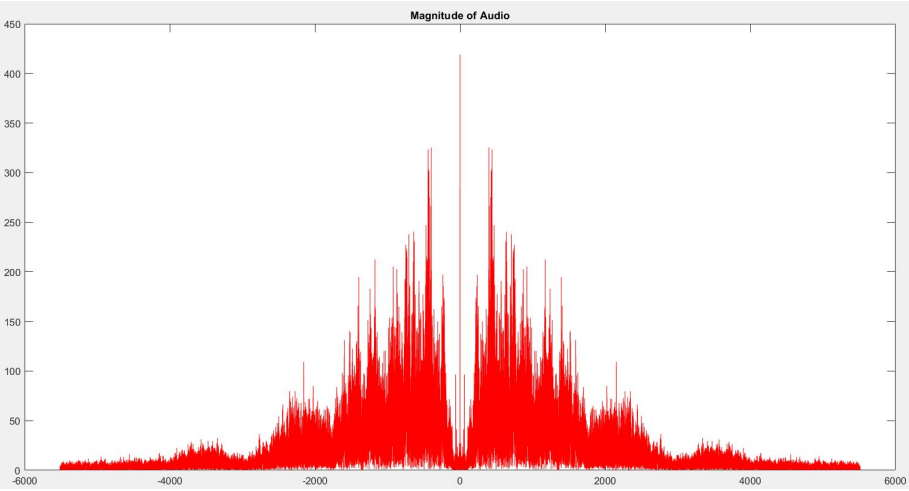

Figure 20: Magnitude of audio

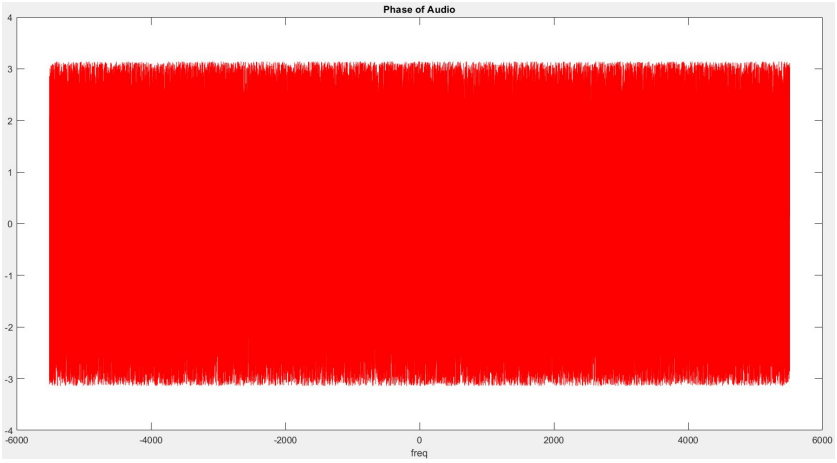

Figure 21: Phase of audio

To calculate the bandwidth due to the perfect symmetry of the plots with respect to the point zero it can be deduced that:

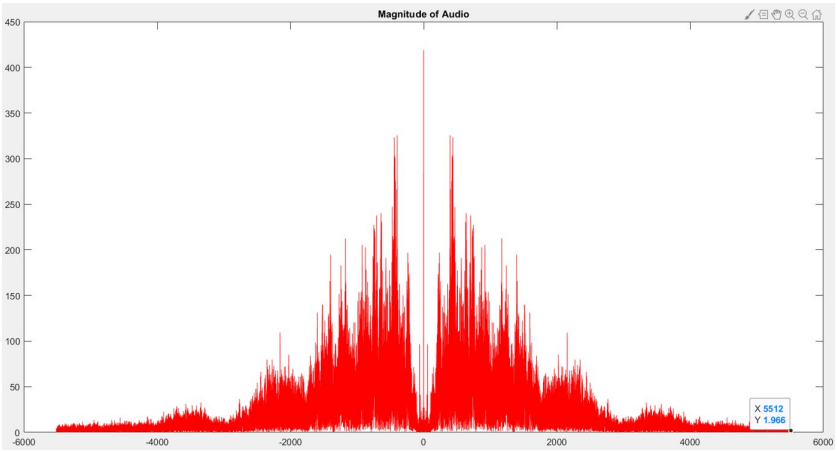

Figure 22: Bandwidth

$$B.W \approx 5500Hz$$

2.3 Part C

First of all we go on to plot and analyze the autocorrelation of the audio.

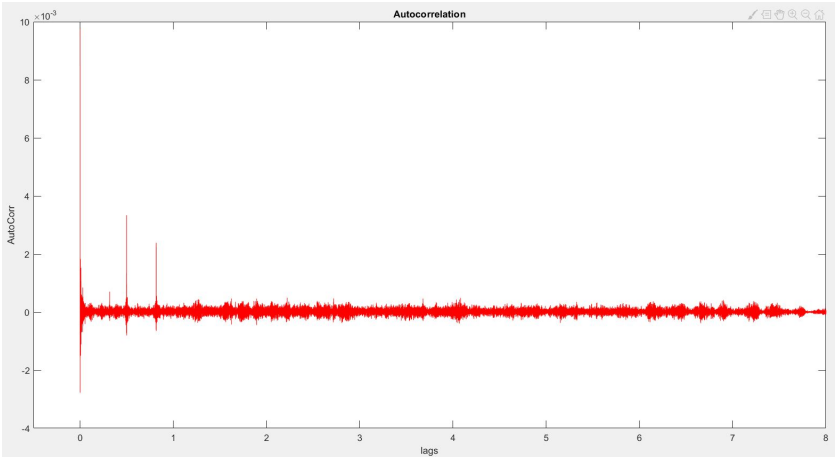

Figure 23: The Autocorrelation plot

Due to our knowledge from the principles of communication systems it is obvious to us that the maximum of our autocorrelation function occurs in the origin, this can be seen in the plot accordingly. But as we can see we have two undesirable jumps in the plot, we assume these to be k_1 and k_2 delays in the following formulae.

$$y[n] = x'[n] + \alpha x'[n - k_1] + \beta x'[n - k_2]$$

We shall attempt to demolish these peaks with a good choice for α and β which we shall obtain based on a mathematical approach.

$$R_{yy}[\tau] = \sum_{n=-\infty}^{\infty} y[n]y[n-k]$$

$$R_{yy}[\tau] = \sum_{n=-\infty}^{\infty} \left(x'[n] + \alpha x'[n - k_1] + \beta x'[n - k_2]\right)\left(x'[n-\tau] + \alpha x'[n - \tau - k_1] + \beta x'[n - \tau - k_2]\right)$$

$$\longrightarrow \sum_{n=-\infty}^{\infty} x'[n]x'[n-\tau] + \alpha \sum_{n=-\infty}^{\infty} x'[n]x'[n-\tau-k_1] + \beta \sum_{n=-\infty}^{\infty} x'[n]x'[n-\tau-k_2]$$

$$+ \alpha \sum_{n=-\infty}^{\infty} x'[n-k_1]x'[n-\tau] + \alpha^2 \sum_{n=-\infty}^{\infty} x'[n-k_1]x'[n-\tau-k_1] + \alpha\beta \sum_{n=-\infty}^{\infty} x'[n-k_1]x'[n-\tau-k_2]$$

$$+ \beta \sum_{n=-\infty}^{\infty} x'[n-k_2]x'[n-\tau] + \alpha\beta \sum_{n=-\infty}^{\infty} x'[n-k_2]x'[n-\tau-k_1] + \beta^2 \sum_{n=-\infty}^{\infty} x'[n-k_2]x'[n-\tau-k_2]$$

$$R_{yy}(\tau) = R_{x'x'}(\tau) + \alpha R_{x'x'}(\tau+k_1) + \beta R_{x'x'}(\tau+k_2) + \alpha R_{x'x'}(\tau-k_1) + \alpha^2 R_{x'x'}(\tau)$$

$$+ \alpha\beta R_{x'x'}(\tau+k_2-k_1) + \beta R_{x'x'}(\tau-k_2) + \beta^2 R_{x'x'}(\tau) + \alpha\beta R_{x'x'}(\tau+k_1-k_2)$$

$$R_{yy}(\tau) = \left(1+\alpha^2+\beta^2\right) R_{x'x'}(\tau) + \alpha\left(R_{x'x'}(\tau+k_1) + R_{x'x'}(\tau-k_1)\right) +$$

$$\beta\left(R_{x'x'}(\tau+k_2) + R_{x'x'}(\tau-k_2)\right) + \alpha\beta\left(R_{x'x'}(\tau+k_2-k_1) + R_{x'x'}(\tau+k_1-k_2)\right)$$

As previously discussed, the two undesirable peaks refer to k_1 and k_2 respectively, we improvise and calculate α and β with the help of the amounts in these peaks and the origin.

$$\alpha = \frac{Peak_1}{Peak_{origin}}, \quad \beta = \frac{Peak_2}{Peak_{origin}}$$

Now we go on to calculate α and β, the following photos depict the peaks.

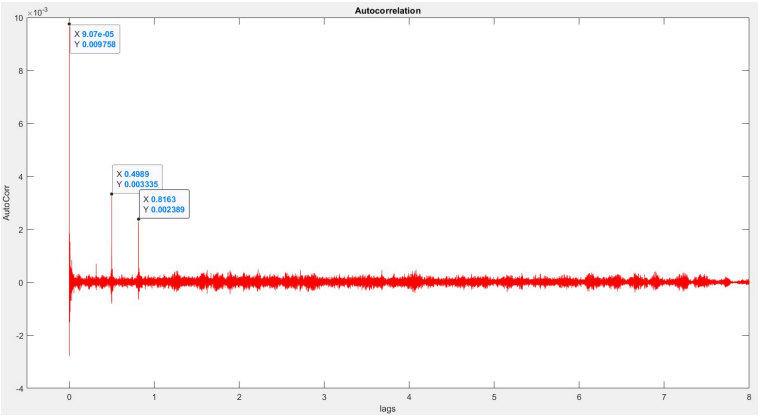

Figure 24: The peaks

$$\alpha = \frac{0.003335}{0.009758} = 0.3417708 \approx 0.3, \quad \beta = \frac{0.002389}{0.009758} = 0.2448247 \approx 0.2$$

2.4 Part D

The delays can easily be calculated as follows.

$Delay = Delay_{Location} \times F_s \quad k_1 = 0.4989 \times 11025 = 5500.375 \approx 5500\ Samples,$
$k_2 = 0.8163 \times 11025 = 8999.7075 \approx 9000\ Samples$

2.5 Part E

The impulse response of the echo system can be simulated as below.

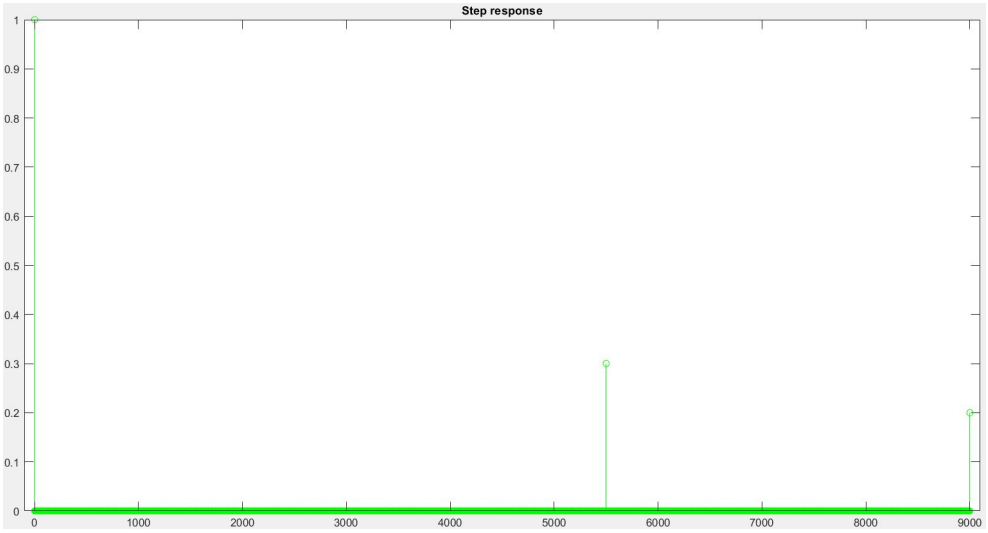

Figure 25: Impulse response of echo system

So we conclude that:

$$h[n] = \delta[n] + 0.3\delta[n - 5500] + 0.2\delta[n - 9000]$$

2.6 Part F

Here we attempt to obtain x' with the filter function and we listen to it, it is observed that the echo is completely removed.

2.7 Part H

Here we attempt to design a filter to remove the noise as much as we can.

I've chosen a Butterworth filter using the Kaiser window, the order of the window is 600 in this design.

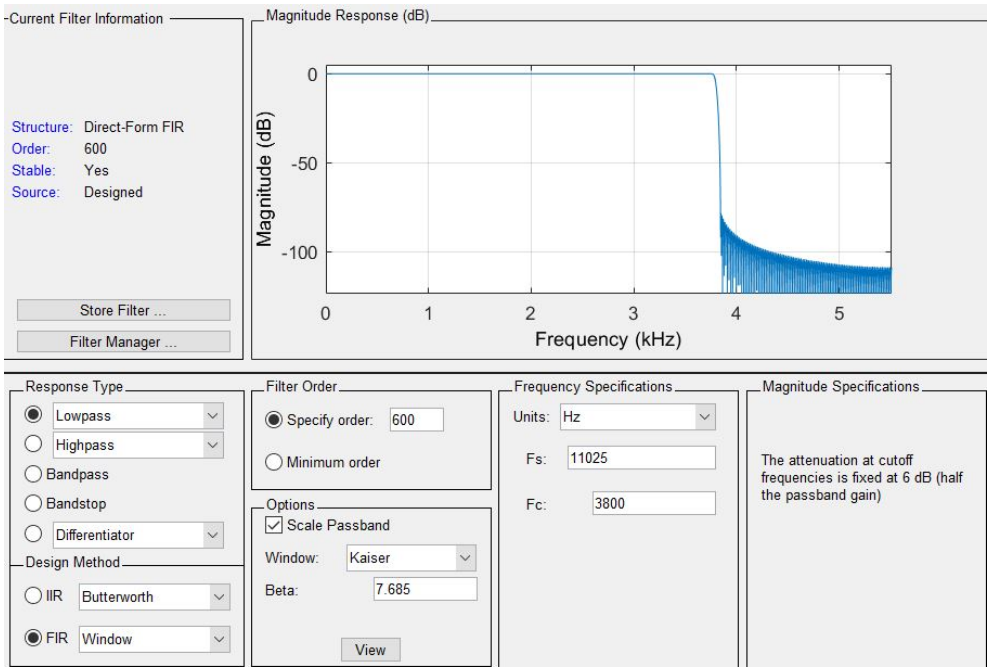

Figure 26: The filter

Now we use the freqz command to display the frequency response.

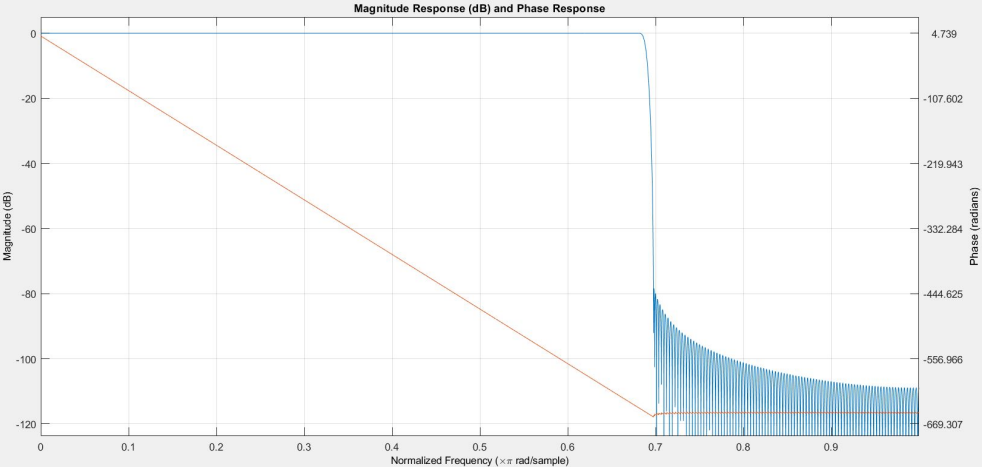

Figure 27: Filter frequency response

We see that this filter is a lowpass filter, This filter is linear in the passband but does not behave linearly in the transition band which is compatible with the material we studied in the course.

2.8 Part I

Here we shall use our designed filter to reduce the noise in the voice file, we will listen to it and analyze the result.

After listening to the voice we deduce that the noise is partially reduced. But all the noise isn't removed because some of the noise is in the low frequencies of the human voice, so most of the noise is removed with this lowpass filter but some of it remains intact.

2.9 Part H

In this part we plot x and x' in the time domain, the results are as follows.

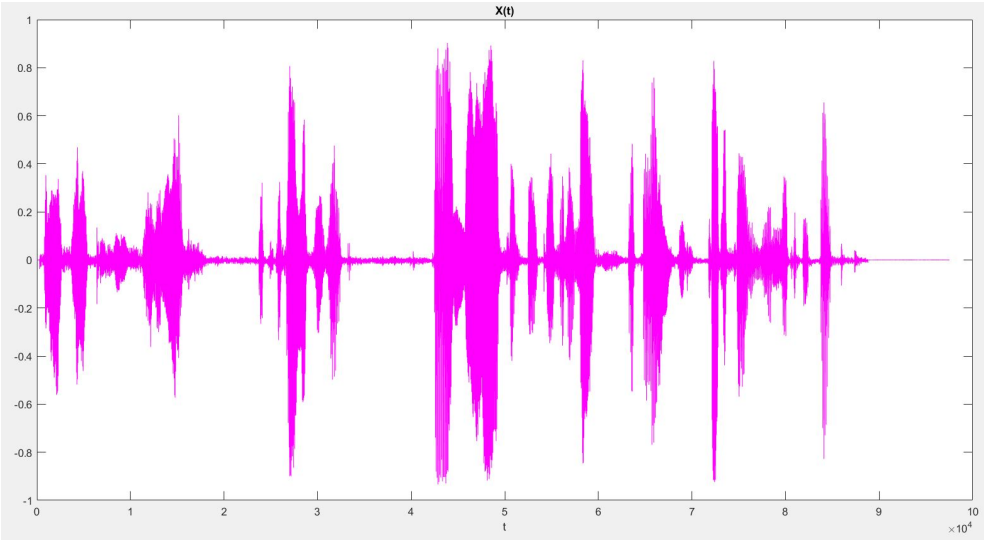

Figure 28: x

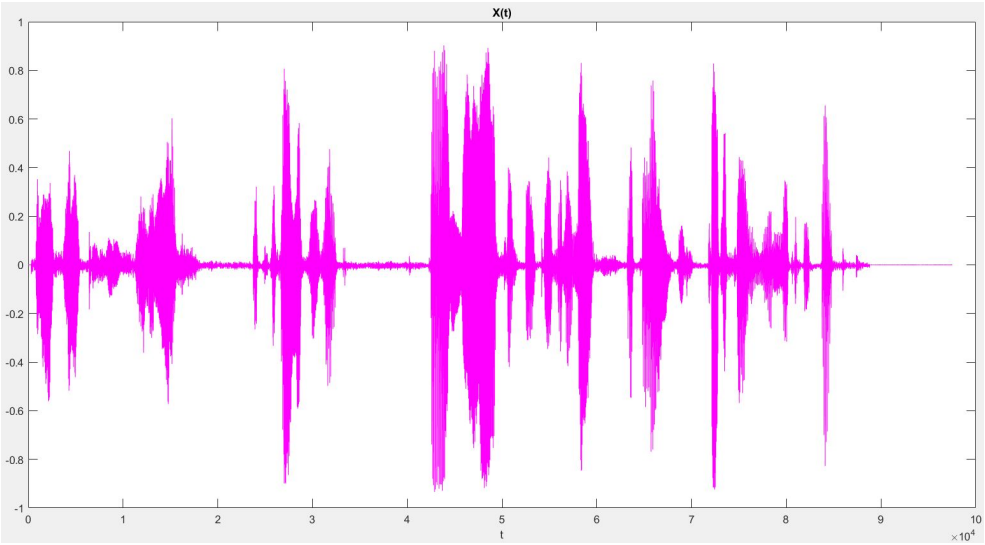

Figure 29: x'

As we can see the structure of both these signals are more or less the

same, yet x has a little less noise than x', this is due to the filter we applied, in other words that filter reduces the bandwidth from $5500Hz$ to $3800Hz$ which is the threshold for the passband, this shows itself in the time domain with less ripples and noise.

We also include the Fourier transforms to see the effect of the Kaiser window.

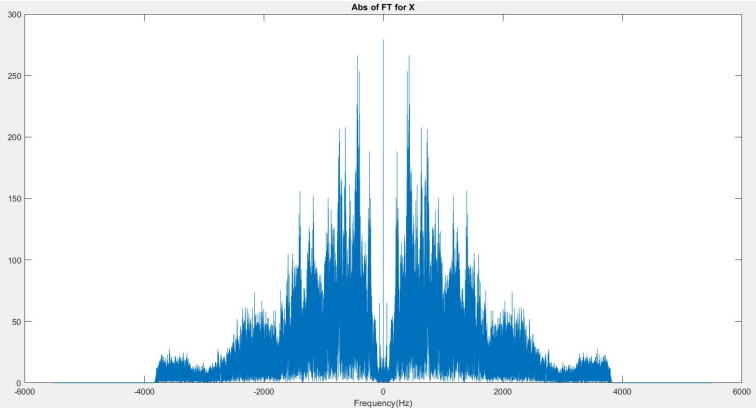

Figure 30: Fourier domain for x

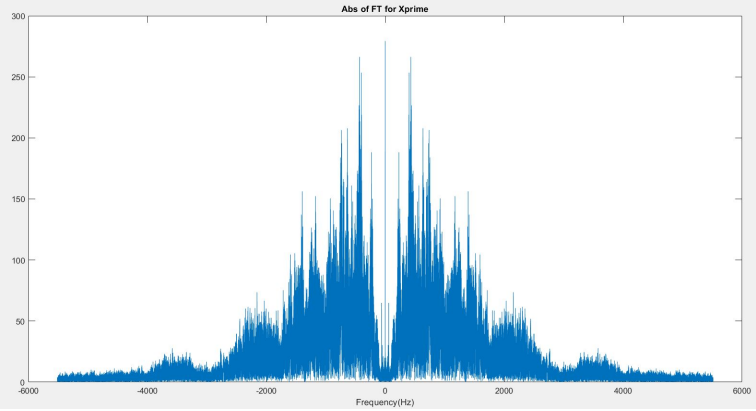

Figure 31: Fourier domain for x'

3 Question 3

In this question we attempt to perform bleep censoring, this bleep is generated with a $1000Hz$ sine wave. We shall attempt to remove the word **California** from Arnold Schwarzenegger's speech when he was elected as the governor of California.

3.1 Part A & B

In this part we read and save the audio files im MATLAB with audioread and listen to them accordingly.

3.2 Part C

In this part we obtain the normalized cross-correlation function with the help of **xcorr**(we must pay attention that we only keep the positive lags and discard the rest) then we go on to plot this cross-correlation which is depicted as follows.

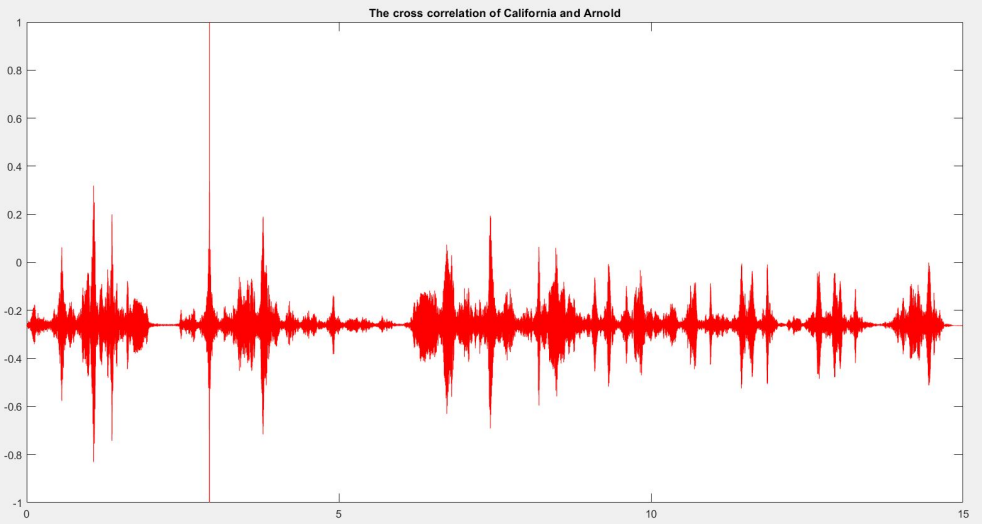

Figure 32: Cross-Correlation

We observe that a peak occurs somewhere in the spectrum, this is where the word **California** is beginning to be uttered by Mr. Schwarzenegger, after this we easily go on to swapping this part of the array with a bleep generated with the same length of the California audio.

3.3 Part D

We listen to the censored voice and save it as requested.

3.4 ATTENTION

Due to the fact that the file **California.wav** is sampled from the beginning of the original speech, its correlation and the **Arnold.wav** file only gives us the location of the first time he utters **California**,This is because the intonation and pronunciation style in different parts of the voice isn't exactly the same hence the peaks aren't very distinguishable, so we perform the bleep censoring only on the first time, this was confirmed by the corresponding TA.

4 Question 4

Here we shall perform sound filtering with MATLAB tools.

4.1 Part A

In this part we read the audio file and listen to it, the sound is horrible! Now we go on to plot its Fourier transform both in a normal and logarithmic scale in the given interval.

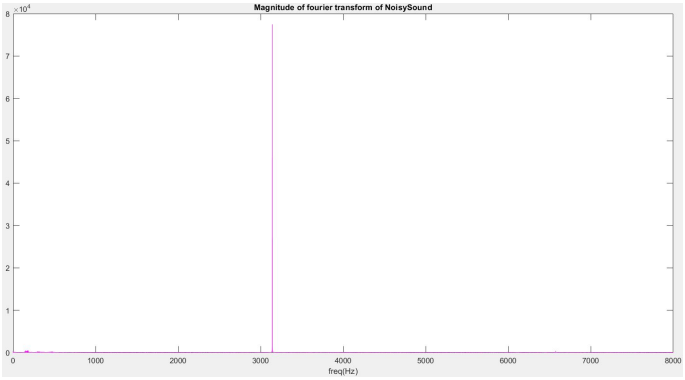

Figure 33: Magnitude of Fourier transform

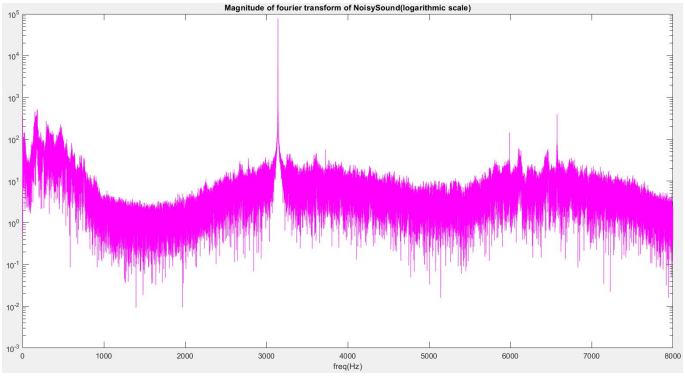

Figure 34: Magnitude of Fourier transform in logarithmic scale

4.2 Part B & C

Here we find the frequency of the disturbing signal as follows.

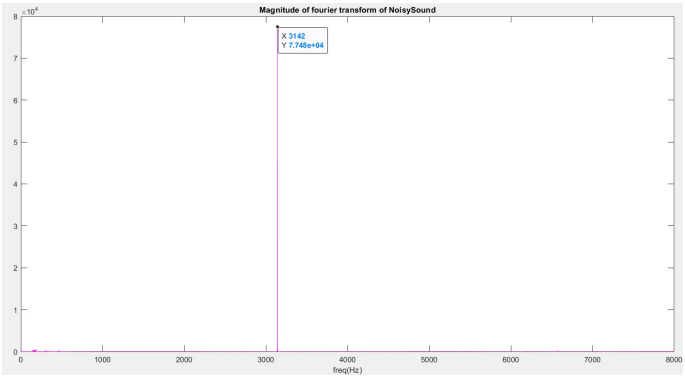

Figure 35: Noise frequency

So we easily have that:

$$Noise_{Freq} = 3142Hz$$

Now we go on to design a bandstop filter in fdatool as follows.

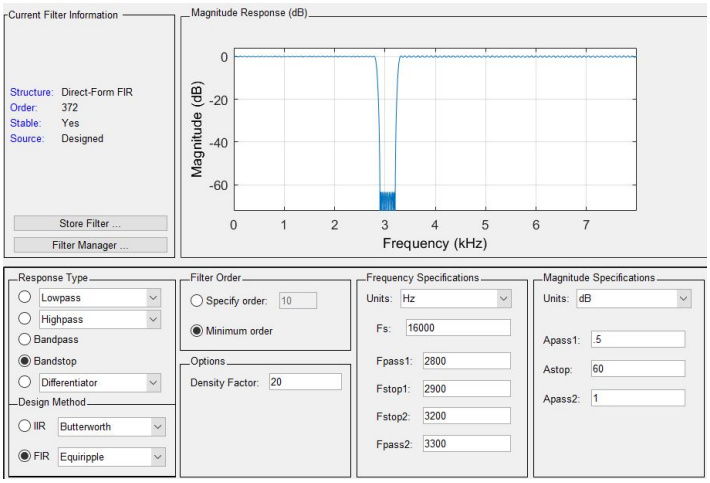

Figure 36: Bandstop filter design sheet

After this we plot the frequency response of our design filter.

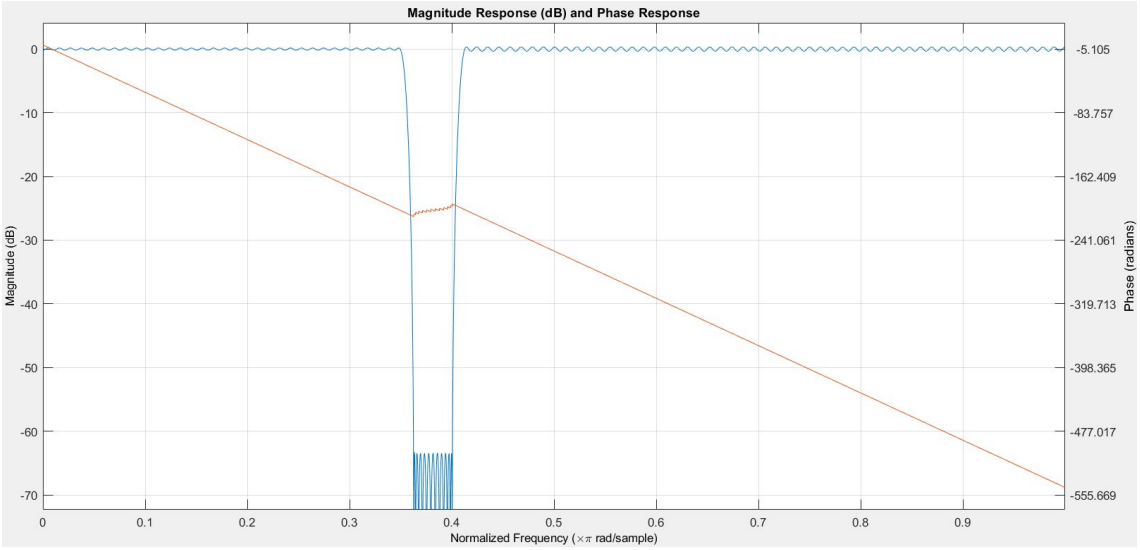

Figure 37: Bandstop filter frequency response

Now we apply our filter to the noisy audio file with the earspilitting noise. We get very satisfactory results which our included in the uploaded file.

4.3 Part D

Here we apply a lowpass filter, an equiripple filter in this case with the given properties, first we design it in FDA tool.

The filter design sheet is as follows.

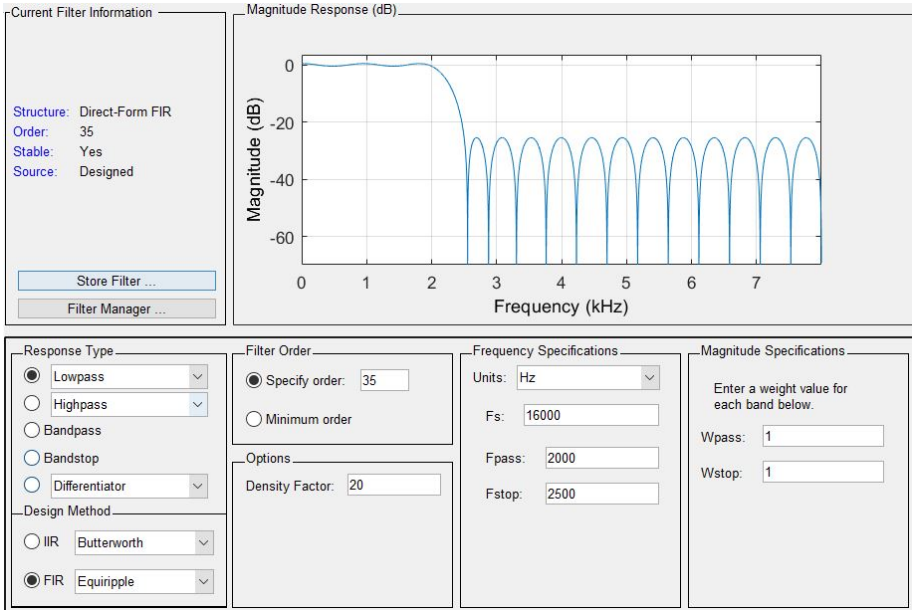

Figure 38: Equiripple filter design sheet

4.3.1 Part A

We listen to the audio before and after applying this filter accordingly.

4.3.2 Part B

We plot the signals before and after filtering.

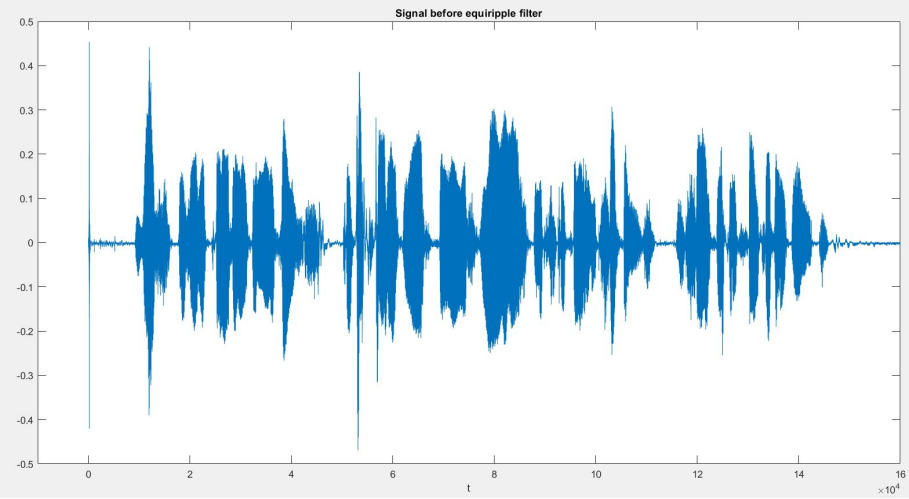

Figure 39: Signal in time domain before equiripple filtering

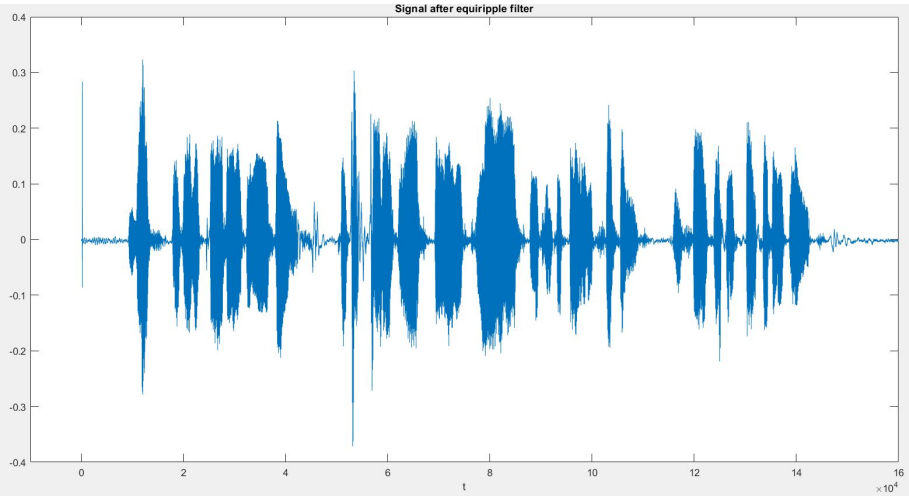

Figure 40: Signal in time domain after equiripple filtering

If we pay attention, we can deduce that the high frequency noises have been removed, if we look at the signal in $t = 0$, before applying the equiripple

filter there is a very large noise figure on it, but after applying it, this noise greatly decreases.

4.3.3 Part C

Here we shall plot the magnitude of the Fourier transform of the signal before and after applying the equiripple filter.

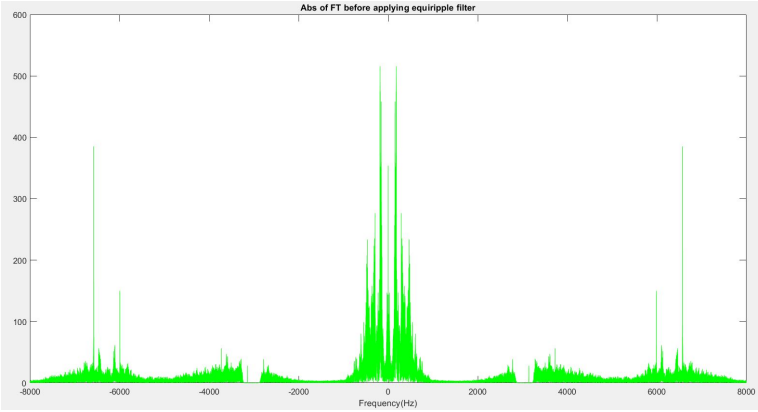

Figure 41: Magnitude of Fourier transform before applying equiripple filter

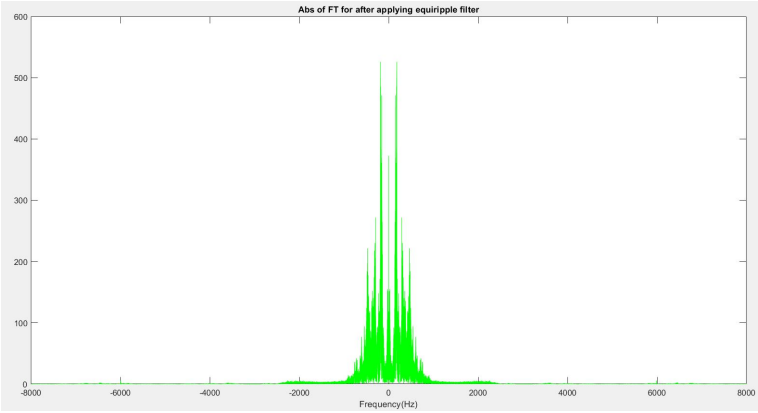

Figure 42: Magnitude of Fourier transform after applying equiripple filter

As we can see the high frequency noise has been beautifully removed.

4.3.4 Part D

Here we plot the frequency response of the equiripple filter.

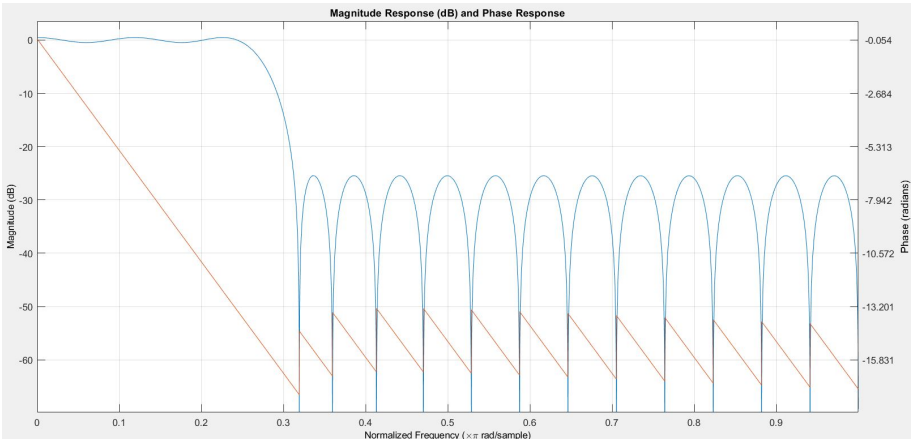

Figure 43: Equiripple filter frequency response

Analysis for Butterworth

Here we repeat all we did for a highpass Butterworth filter.

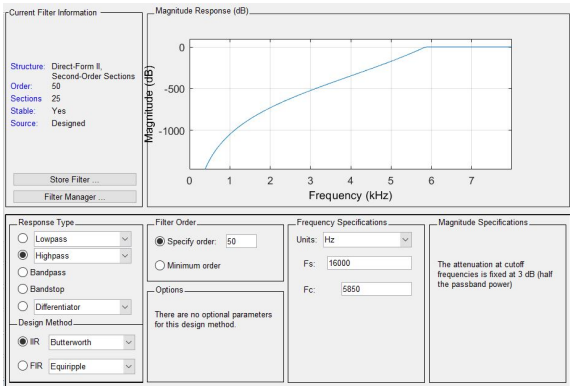

Figure 44: Butterworth filter design sheet

4.3.5 Part A2

We listen to the audio before and after applying this filter accordingly.

4.3.6 Part B2

We plot the signals before and after filtering.

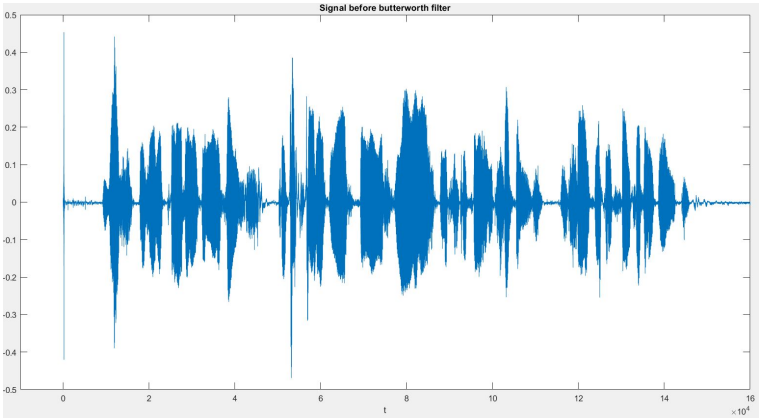

Figure 45: Signal in time domain before butterworth filtering

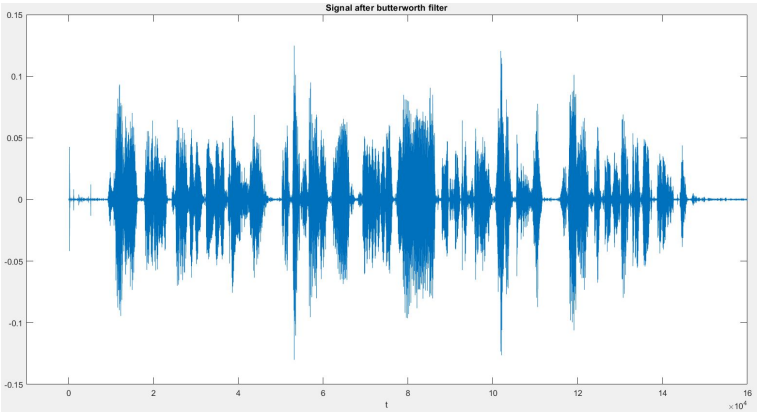

Figure 46: Signal in time domain after butterworth filtering

As we can see, most of the noise remains intact, and the high frequency noise is passed this is better seen in the next section.

4.3.7 Part C2

Here we shall plot the magnitude of the Fourier transform of the signal before and after applying the Butterworth filter.

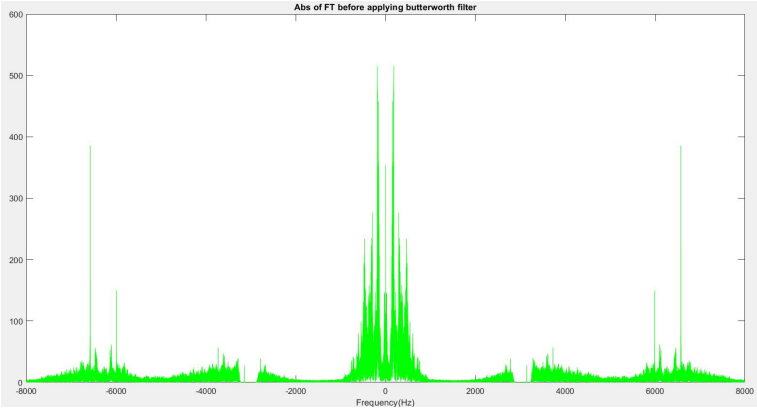

Figure 47: Magnitude of Fourier transform before applying Butterworth filter

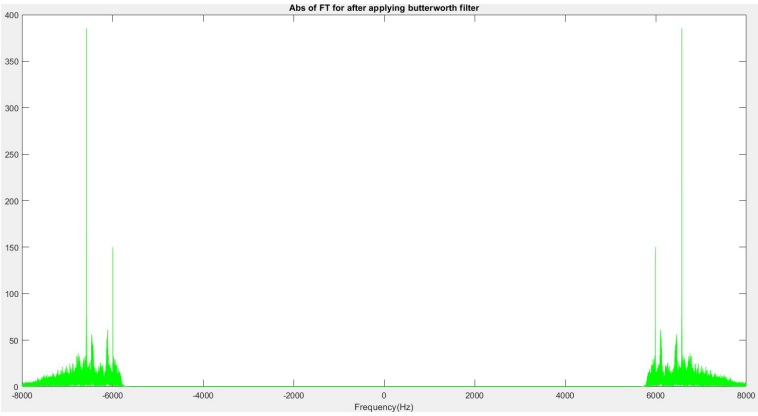

Figure 48: Magnitude of Fourier transform after applying Butterworth filter

As we can see the high frequency noise has remained intact, so in other words, a large portion of the speech has been removed.

4.3.8 Part D2

Here we plot the frequency response of the Butterworth filter.

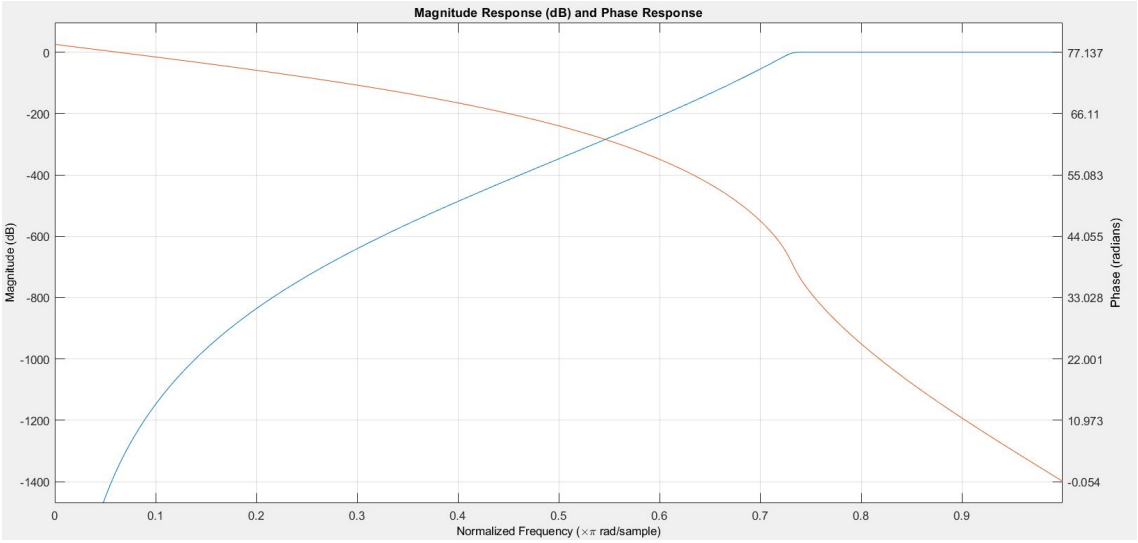

Figure 49: Butterworth filter frequency response

References

[1] Mohammad Ali Akhaee, *Digital Signal Processing lecture notes, Spring 01*

[2] Ali Olfat, *Principal of communication systems lecture notes, Fall 00*

[3] Amirmasoud Rabiei, *Signals and systems lecture notes, Spring 00*