The
BRITISH UNIVERSITY
IN EGYPT

COMPUTER
ENGINEERING
DEPARTMENT

SIGNALS AND SYSTEMS

22ELEC14C

PROJECT REPORT

Submitted to

Dr. Hassan Saleh & Eng. Noura El-Shabasy

COMP. Dep., BUE

Students:

Mohamed Ahmed Abdel Galeil ***ID:*** *214618*

COMP. Dep. DY2

1. Introduction:

Signal processing is an important part in the modern world. In this report, we will process an audio signal, doing on them specific operations like; amplitude scaling, sinusoidal amplitude scaling, filtering, time limiting, shifting, and Fast Fourier Transform

2. Code:

1. Audio Signal

```
%% Input Audio
[f,fs] = audioread('Mohamed214618.mp3');
sound(f,fs)
subplot(7,1,1)
plot(f)
title('Audio signal')
xlabel 'Time'
ylabel 'Audio signal'
grid on
```

FIGURE 1: AUDIO SIGNAL INPUT CODE

In this section of code, we use the “audioread” function to read the audio file and store the signal in a variable called “f”, and its sampling frequency in another variable called “fs”. After storing the signal, it is graphed against time.

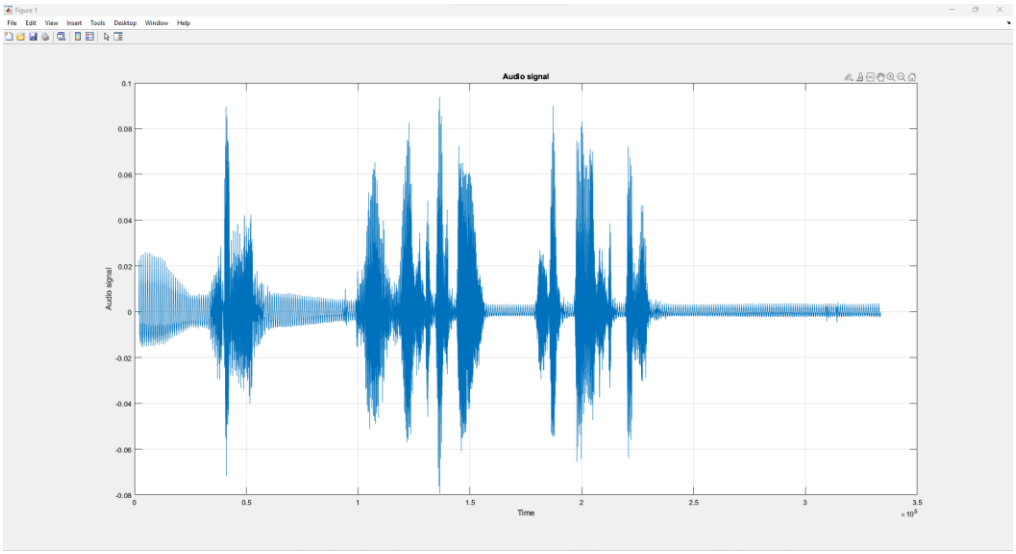

FIGURE 2: AUDIO SIGNAL GRAPH

This graph shows the unfiltered audio with large spikes when speaking and non-zero values due to noise.

2. Filtered Audio Signal

```
%% Design a bandpass filter that filters out between 1000 to 12000 Hz
n = 7;
beginFreq = 1000 / (fs/2);
endFreq = 12000 / (fs/2);
[b,a] = butter(n, [beginFreq, endFreq], 'bandpass');
fi = filter(b, a, f);                                    %% Filter the signal
sound(fi,fs)
subplot(7,1,2)
plot(fi)
title('Filtered audio signal')
xlabel 'Time'
ylabel 'Audio signal'
grid on
```

FIGURE 3: FILTER DESIGN AND IMPLEMENTATION CODE

In this section, a “bandpass” filter is designed to filter out noise and static, the conditions of the filter are stored in “beginFreq”, “endFreq”, and “n” which are used to design filter “[b, a]”. The filter is applied on the signal and stored in “fi”. The filtered signal is graphed against time.

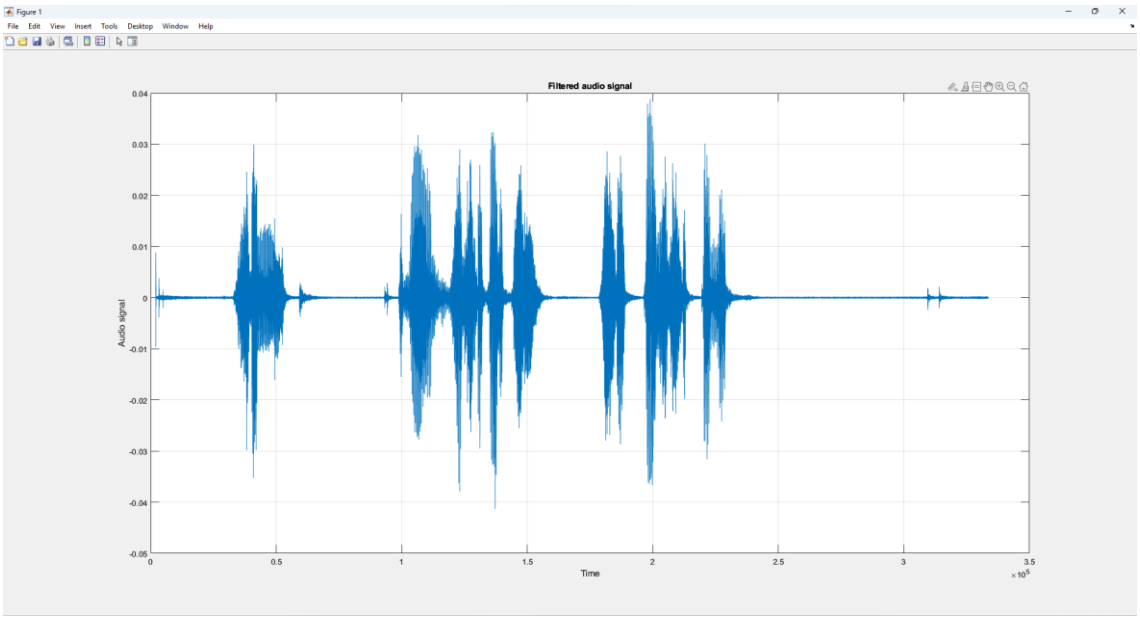

FIGURE 4: FILTERED AUDIO SIGNAL GRAPH

This graph shows the filtered audio with large spikes when speaking and almost zero values at silence.

3. Time-Limited Audio Signal

```
23      %% Time-limited audio signal
24      N = length(f);                                  %% sample length
25      t = linspace(0, 4, N);
26      f1 = f(1:fix(N/2),:);
27      soundsc(f1,fs)
28      subplot(7,1,3)
29      plot(t,f)                                       %% pplots the time-limited audio
30      xlim([0 10])
31      title('Time-limited audio signal')
32      xlabel 'Time'
33      ylabel 'Audio signal'
34      grid on
```

FIGURE 5: TIME-LIMITING CODE

In this section, the filtered signaled is time-limited by plotting it against “t” which is limited, we use “xlim” function to plot it correctly, “N” is the length of the signal, which is used to play back the correct segment of audio.

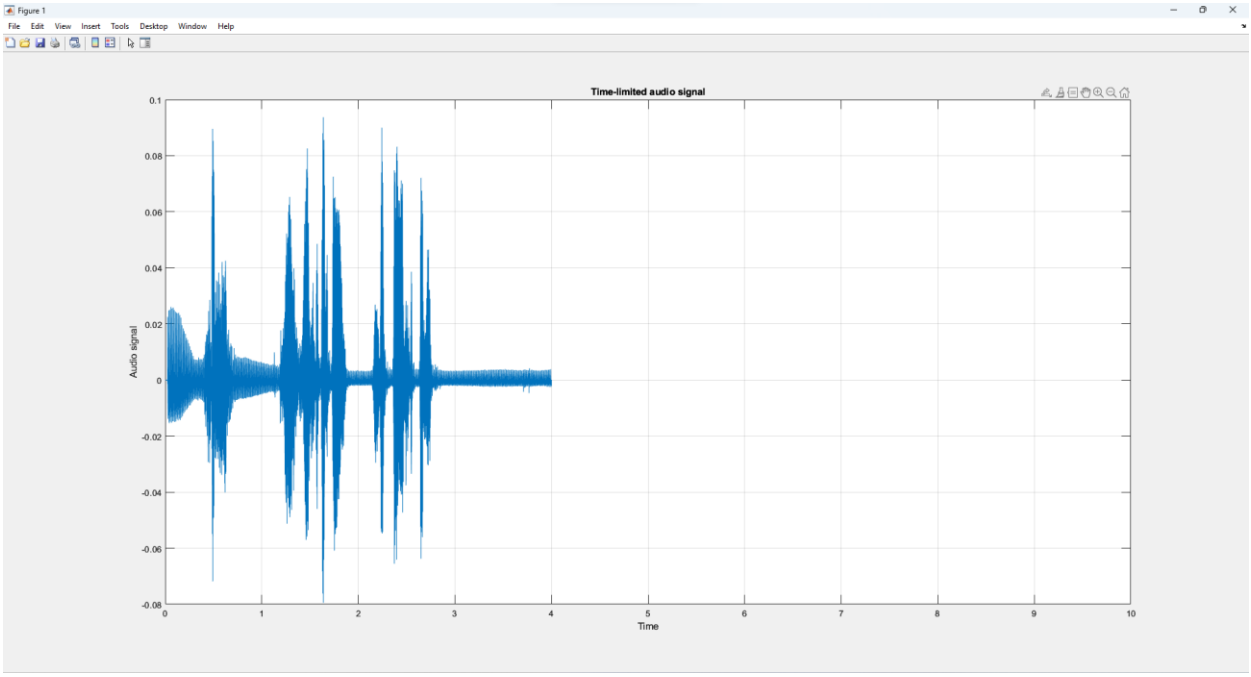

FIGURE 6: TIME-LIMITED FILTERED AUDIO SIGNAL GRAPH

This graph shows the time-limited filtered audio with large spikes when speaking and almost zero values at silence.

4. Shifted and Time-Limited Audio Signal

```
35  %% Time-limited and shifted audio signal
36  x = t + 4;
37  delay2 = 3*fs;
38  Z = zeros(delay2,1);
39  f2 = [Z;f(1:end-delay2)];
40  sound(f2,fs)
41  subplot(7,1,4)
42  plot(x,f)                                   %% pplots the time-limited shifted audio
43  xlim([0 10])
44  title('Time-limited and shifted audio signal')
45  xlabel 'Time'
46  ylabel 'Audio signal'
47  grid on
```

FIGURE 7: TIME-LIMITING AND SHIFTING CODE

In this section, the filtered signaled is time-limited by plotting it against “t + 4”, which is limited and shifted, we use “xlim” function to plot it correctly. “delay2” is a similar sampling frequency, combined with signal “Z” which is an empty signal, which is used to store the shifted signal in “f2”, to play back the correct shifted segment of audio.

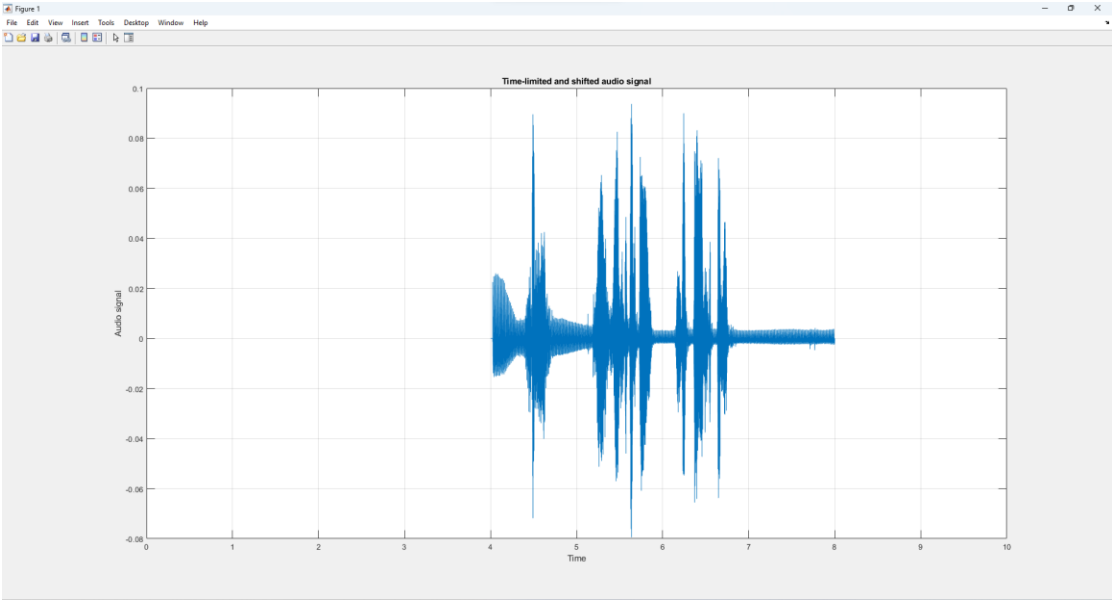

FIGURE 8: SHIFTED TIME-LIMITED FILTERED AUDIO SIGNAL GRAPH

This graph shows the shifted time-limited filtered audio with large spikes when speaking and almost zero values at silence.

5. Amplitude Scaled Audio Signal

```
48  %% Amplitude Scaling
49  fOut = 6*fi;
50  p = audioplayer(fOut, fs);                    %% Construct audioplayer object and play
51  %p.play;
52  subplot(7,1,5)
53  plot(fOut)
54  title('Amplitude Scaling')
55  xlabel 'Time'
56  ylabel 'Audio signal'
57  grid on
```

FIGURE 9: AMPLITUDE SCALING CODE

In this section, the filtered signal “fi” is multiplied to a constant to boost the signal, and then stored in “fOut”, “fOut” is put in an “audioplayer” function to play it back, and then “fOut” is graphed against time.

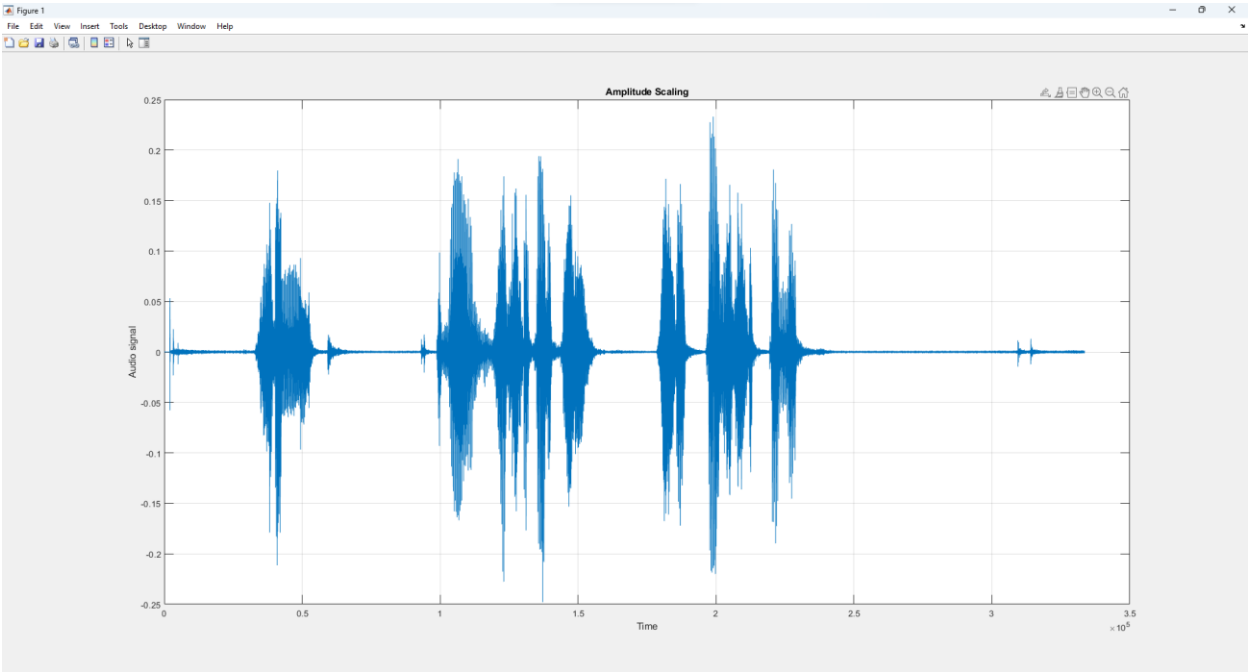

FIGURE 10: AMPLITUDE BOOSTED FILTERED AUDIO SIGNAL GRAPH

This graph shows the amplitude boosted filtered audio with large spikes when speaking and almost zero values at silence.

6. Sinusoidal Amplitude Scaled Audio Signal

```
%% Sinusoidal Amplitude Scaling
dt = 1/80;                                  %% seconds per sample
StopTime = 3.5;                             %% seconds
u = (0:dt:StopTime);                        %% seconds
Fc = 60;                                    %% hertz
si = sin(2*pi*Fc*u);
fsi = fi * si;
stereoSound = fsi(:, 2:2);                  %% Extract first two channels and ignore remaining channels.
sound(stereoSound,fs)
subplot(7,1,6)
plot(fsi)
title('Sinusoidal Amplitude Scaling')
xlabel 'Time'
ylabel 'Audio signal'
grid on
```

FIGURE 11: SINUSOIDAL AMPLITUDE SCALING CODE

In this section, “si” is a sinusoidal wave generated, the filtered signal “fi” is multiplied to “si” to boost the signal, and then stored in “fsi”, “fsi” is played back, and then “fsi” is graphed against time.

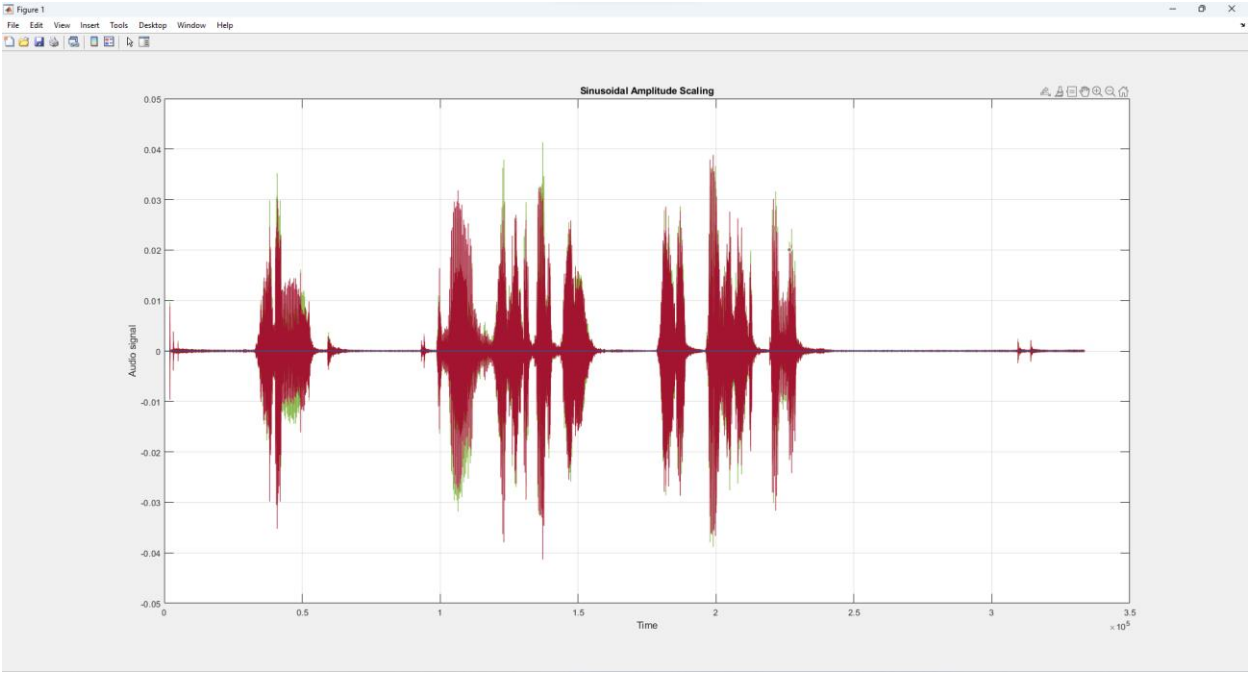

FIGURE 12: SINUSOIDAL AMPLITUDE BOOSTED FILTERED AUDIO SIGNAL GRAPH

This graph shows the sinusoidal amplitude boosted filtered audio plotted against time.

7. Fourier Transform Unfiltered

```
73    %% Fast Fourier Transform Unfiltered Signal
74    A = size(f,1);
75    df = fs / A;
76    w = (-(A/2):(A/2)-1)*df;
77    y = fft(f(:,1), A) / A;
78    y2 = fftshift(y);
79    subplot(10,1,7)
80    plot(w,abs(y2))                                    %% Plot the spectrum
81    title('Fast Fourier Transform, Frequency Domain, Unfiltered Signal')
82    xlabel 'Frequency'
83    ylabel 'Audio signal'
84    grid on
85    subplot(10,1,8)
86    plot(w,angle(y2))
87    title('Fast Fourier Transform, Phase Domain, Unfiltered Signal')
88    xlabel 'Phase'
89    ylabel 'Audio signal'
90    grid on
```

FIGURE 13: UNFILTERED FAST FOURIER TRANSFORM CODE

In this section, “A” is the size of the unfiltered signal “f”, “df” is the sampling rate, “w” is the frequency domain, “y” is the FFT of “fi”. “y2” is FFT shifted to zero. The absolute of “y2” is plotted against “w” to show the frequencies of the signal, and the angle of “y2” is plotted against “w1” to show the phase of the signal.

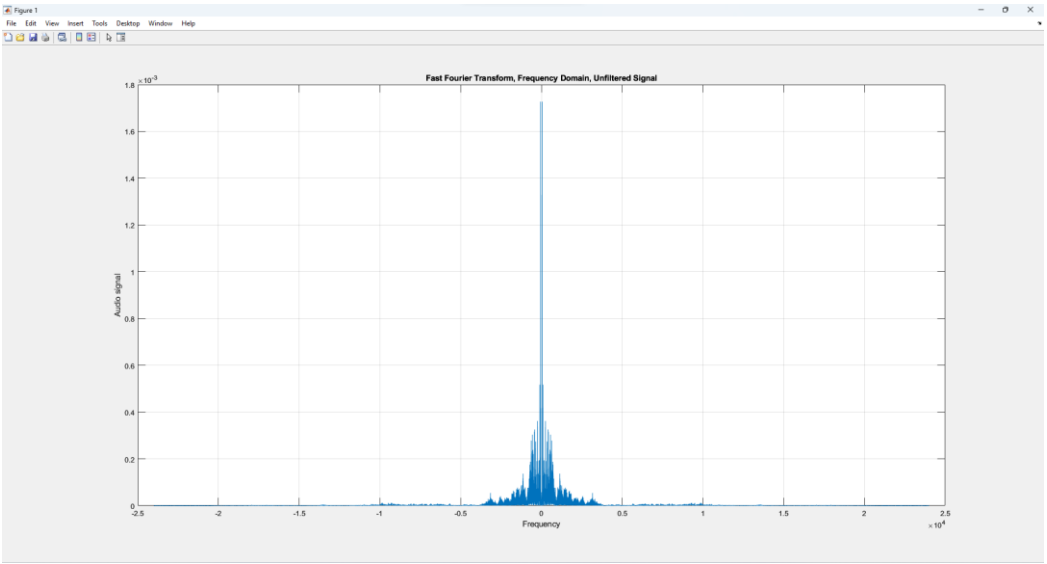

FIGURE 14: FREQUENCY DOMAIN OF UNFILTERED AUDIO SIGNAL

This graph shows the unfiltered signal plotted against the frequency domain by the Fast Fourier Transform, to show the frequencies of the signal.

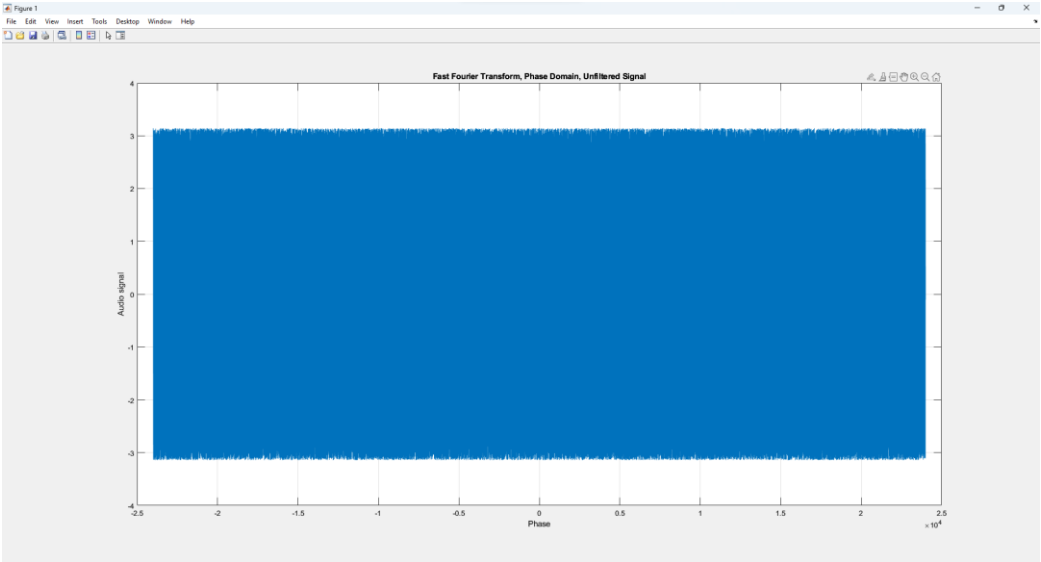

FIGURE 15: PHASE DOMAIN OF UNFILTERED AUDIO SIGNAL

This graph shows the unfiltered signal plotted against the phase domain by the Fast Fourier Transform, to show the phases of the signal.

8. Fourier Transform Filtered

```
91   %% Fast Fourier Transform Filtered Signal
92   A1 = size(fi,1);
93   df1 = fs / A1;
94   w1 = (-(A1/2):(A1/2)-1)*df1;
95   y1 = fft(fi(:,1), A) / A;
96   y3 = fftshift(y1);
97   subplot(10,1,9)
98   plot(w1,abs(y3))                                    %% Plot the spectrum
99   title('Fast Fourier Transform, Frequency Domain, Filtered Signal')
100  xlabel 'Frequency'
101  ylabel 'Audio signal'
102  grid on
103  subplot(10,1,10)
104  plot(w1,angle(y3))
105  title('Fast Fourier Transform, Phase Domain, Filtered Signal')
106  xlabel 'Phase'
107  ylabel 'Audio signal'
108  grid on
```

FIGURE 16: FILTERED FAST FOURIER TRANSFORM CODE

In this section, “A1” is the size of the filtered signal “fi”, “df1” is the sampling rate, “w1” is the frequency domain, “y1” is the FFT of “fi”. “y3” is FFT shifted to zero. The absolute of “y3” is plotted against “w1” to show the frequencies of the signal, and the angle of “y3” is plotted against “w1” to show the phase of the signal.

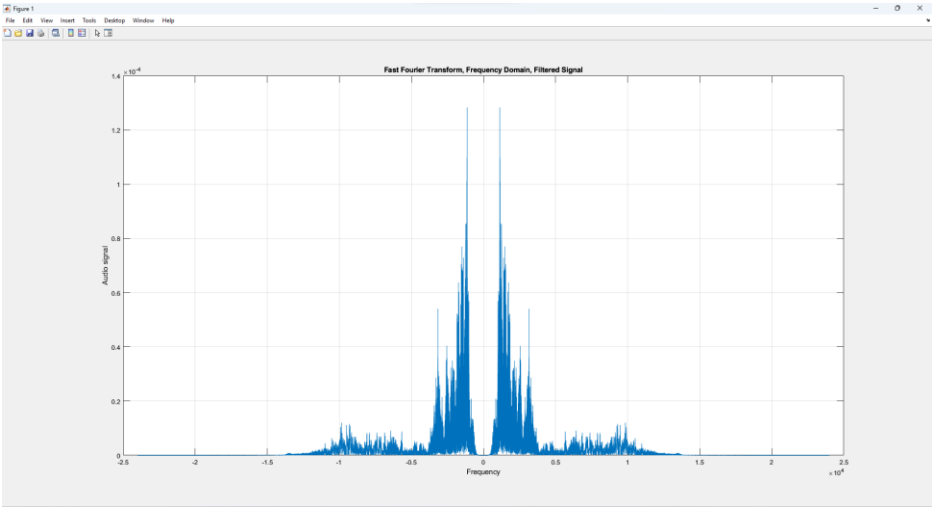

FIGURE 17: FREQUENCY DOMAIN OF FILTERED AUDIO SIGNAL

This graph shows the filtered signal plotted against the frequency domain by the Fast Fourier Transform, to show the frequencies of the signal.

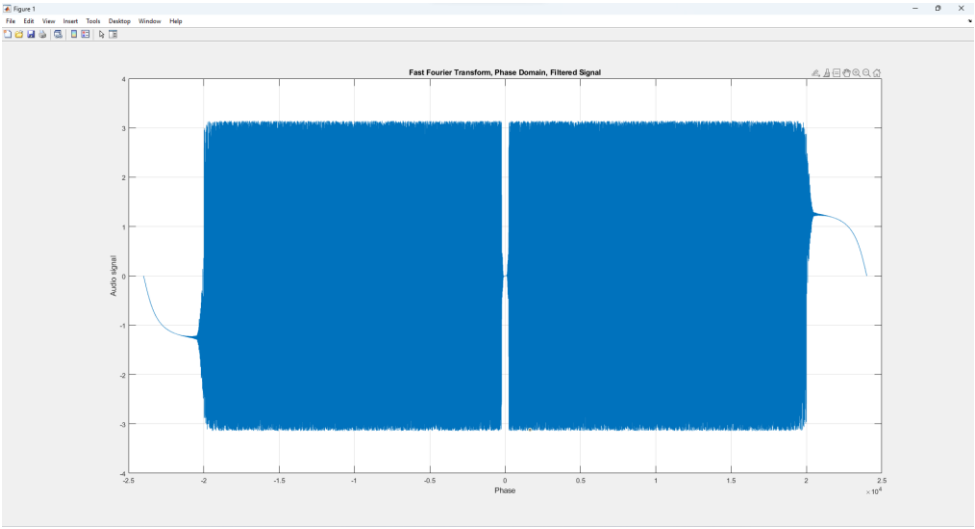

FIGURE 18: PHASE DOMAIN OF FILTERED AUDIO SIGNAL

This graph shows the filtered signal plotted against the phase domain by the Fast Fourier Transform, to show the phases of the signal.

3. Conclusion

Using Matlab, signal processing becomes an easy task, all the formulas used describe many real life processes such as amplitude scaling (boosting), time limiting and shifting (editing and recording audio), FFT (used to determine frequencies in a signal), and filters (noise cancelation).

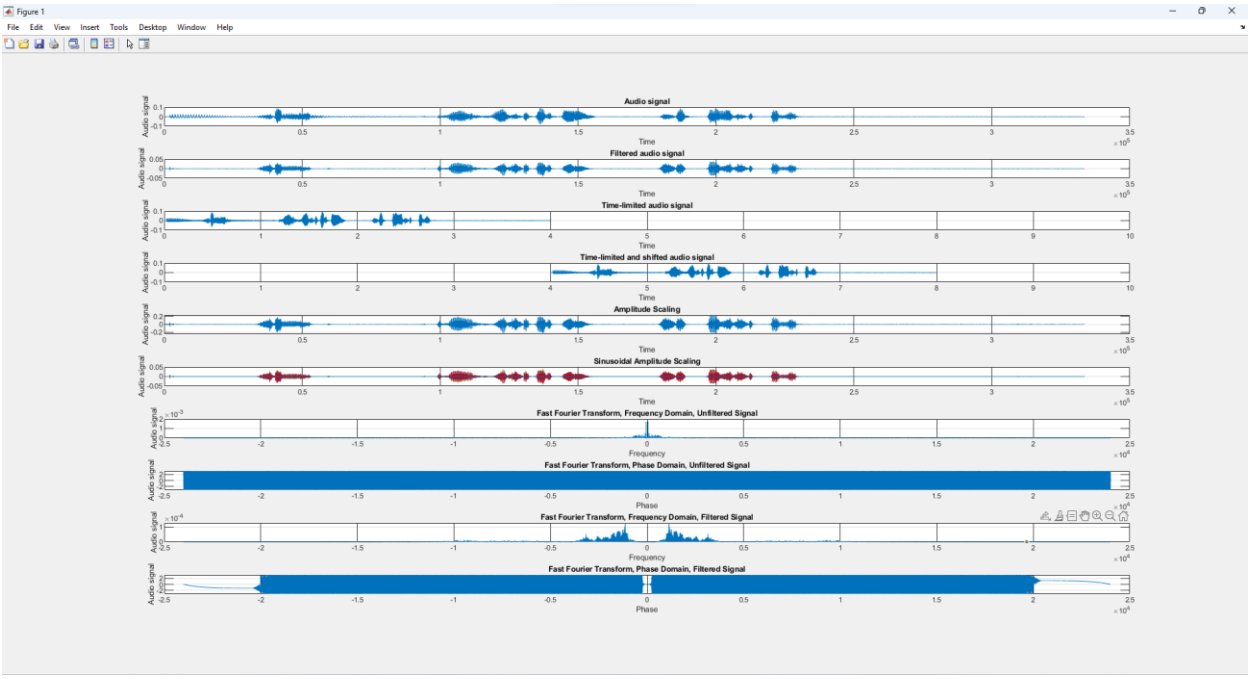

FIGURE 19: ALL RESULTS