**EE556 Digital Signal Processing**

**Project 2: Cleaning up an Audio File**

**Yusuf Shaikh (816921177)**

**Benjamin Hunt (815841735)**

**12/23/2016**

**Introduction:**

In this project of digital signal processing we explored Matlab’s capabilities of reading, writing and cleaning audio signals. Sometimes recorded audio ends up with unintentional and unpleasant residual low-level sound. These sounds are what is known as “audio noise” or “noise” and can be: a hiss, rumble, crackle and a hum in the background of the audio playback. These noises usually occur due to background noise during recording, or sometimes even when converting between analog and digital. Luckily with post-processing development we are now able to remove most noise and clean up the audio signals. Unfortunately, however, it is impossible to completely remove noise due to the existence of random signals, but still obtain a satisfactory quality.

An audio signal is a range of electrical voltage between 0 and VDD, which make up frequencies. To clean the audio signal we have design and use FIR filters removing certain unwanted frequencies. The four types of Linear-Phase FIR filters that Matlab is capable of implementing are: low-pass filter, high-pass filter, band-pass filter, band-stop filter. In this report we will explain each filter, it’s use and how we used certain filters to remove the noise from the given audio signal.

**Results:**

We started the project by analyzing the original audio signal, seen in Figure 1, the original signal was intended on being a normal speech signal, whose peaks should be sharp and uniform. However, as we can see the signal was filled with unintentional noise, overlapping the genuine speech signal.  Our goal is to remove the noise, which are two sinusoids and some high frequency colored noise.

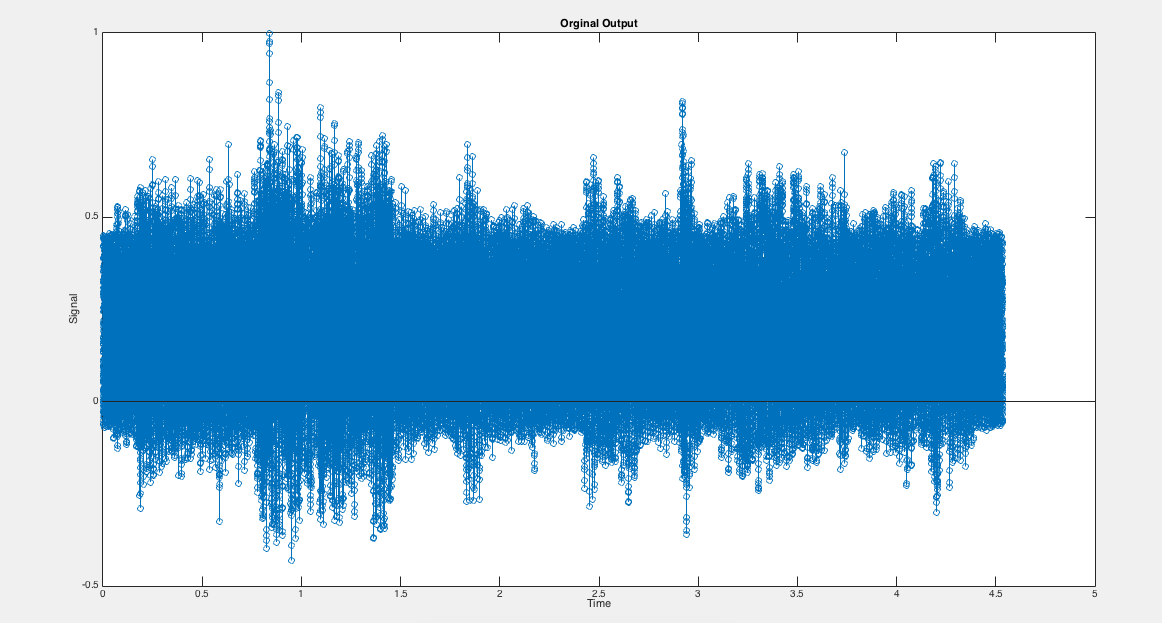


Figure 1

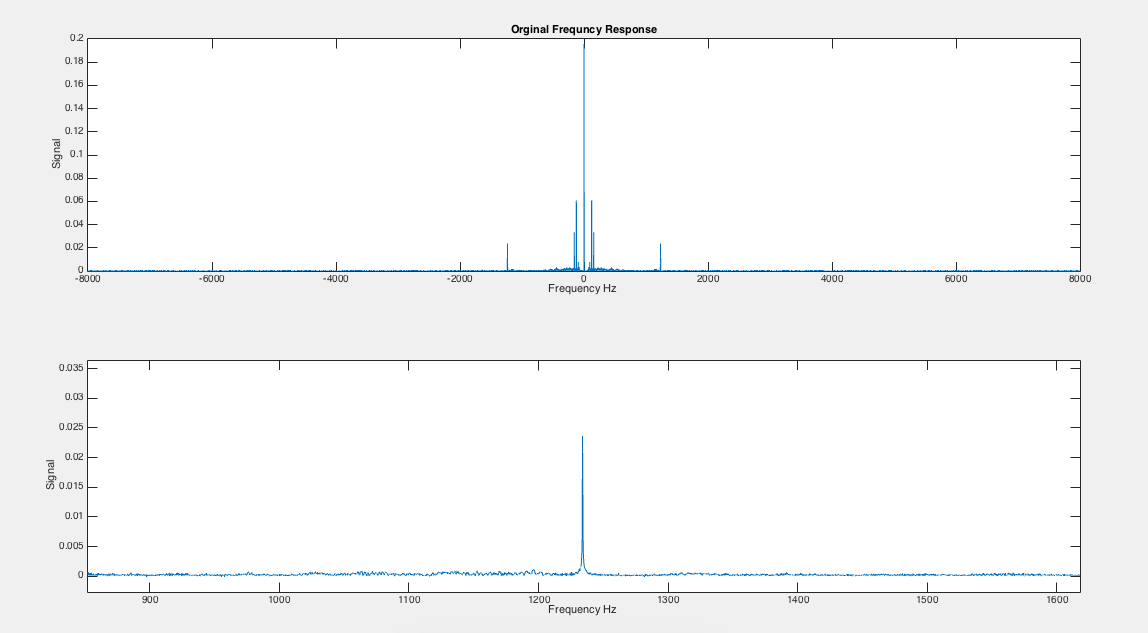
Our first step in the cleanup process was to take the *Fourier Transform*, or in Matlab the *Fast Fourier Transform*(‘fft’), of the signal. This puts the signal into the Frequency Domain and allows us to find the areas of the plot where the frequency spikes. These Spikes represent various noise that are not apart of the original signal. Figure 2 shows the FFT of the signal.

Figure 2 – a. Original FFT b. Target frequency to remove

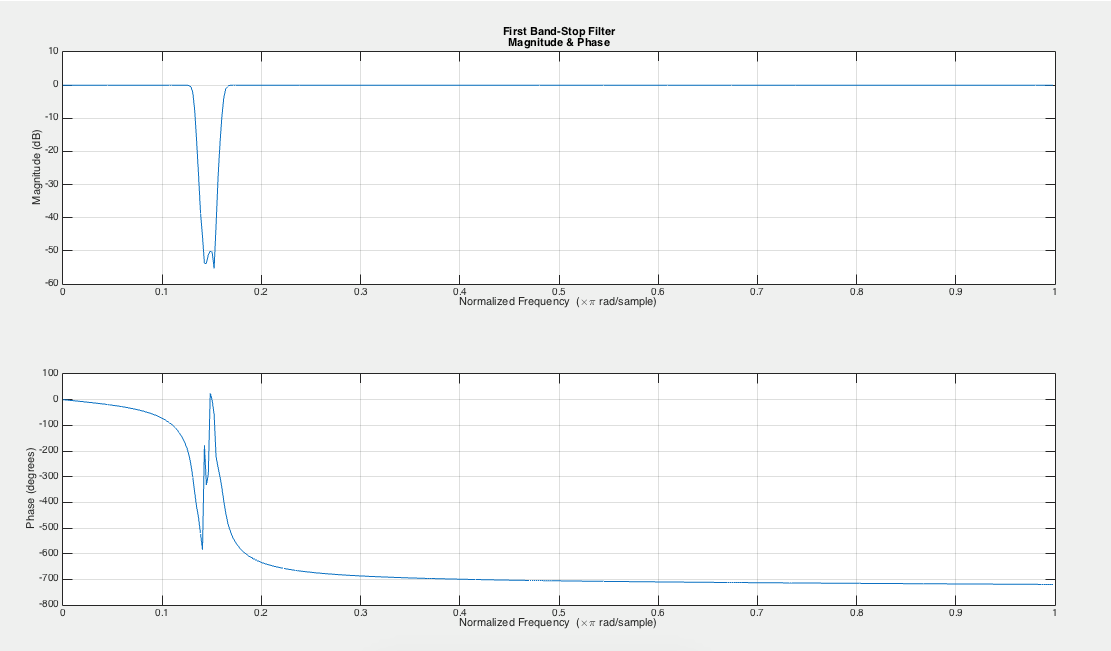
 By looking at the Fourier Transform, we can see some obvious spikes. We will have to use various filters to remove all the spikes, as each spike requires a different cleanup process, without taking out too much of the original signal. First, we will take care of the noise produced by one of the sinusoids. We can see from the plot, that the noise is produced somewhere between 1000 and 1300 Hz. To remove only those frequencies, we used a band-stop filter, which removes all frequencies between two given cut-off frequencies. Below is the impulse response of the band-stop filter.

Figure - Impulse Response of the first Band-Stop Filter

After the band-stop filter removes the first sinusoid, we re-plot the FFT (Figure 4) and look for other discrepancies. We can see some noise being produced somewhere between 50 and 250 Hz. Again, we use a band-stop filter to remove those specific frequencies, with cut-off frequencies of 50 and 300 Hz. Below is the impulse response of the filter (Figure 5).

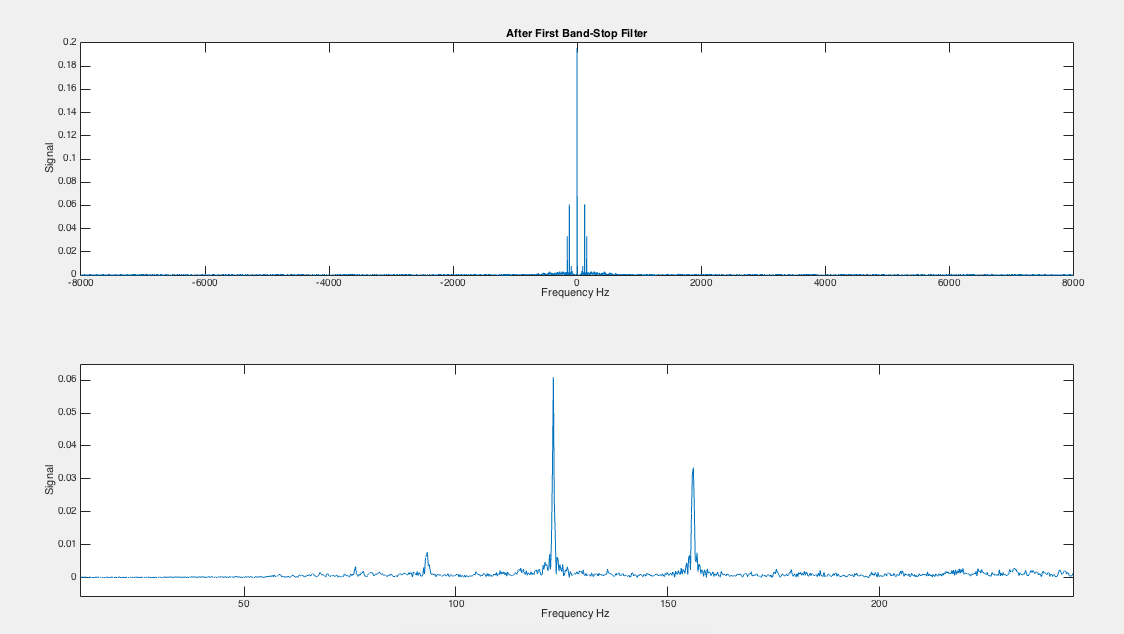


Figure - FFT after the first Band-Stop filter

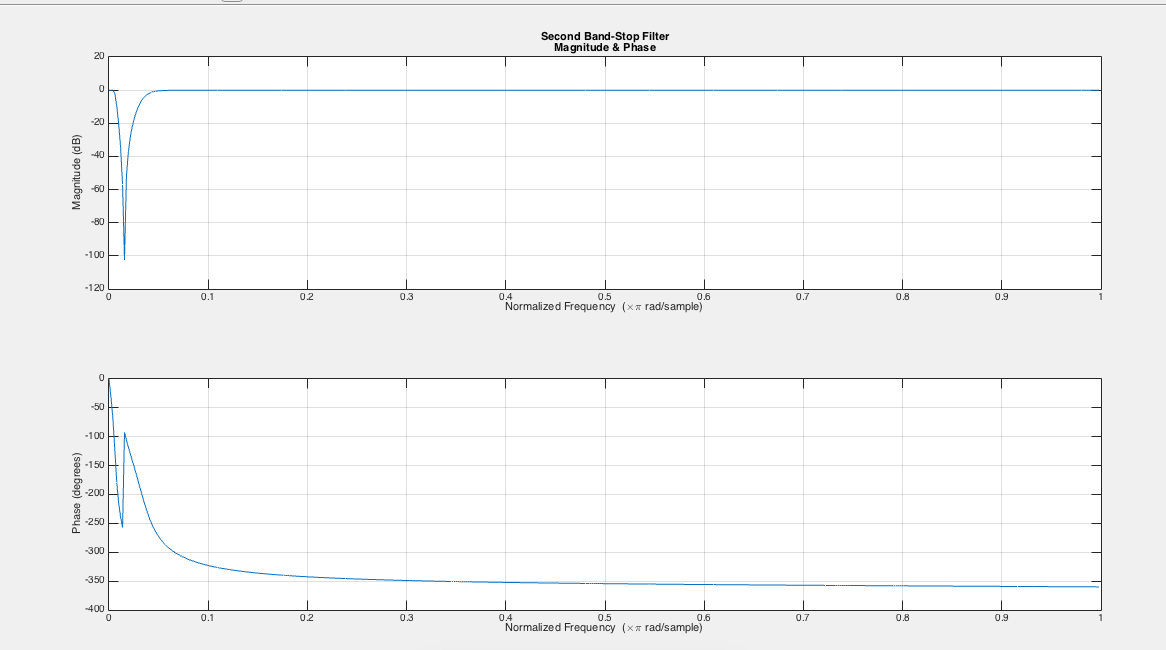
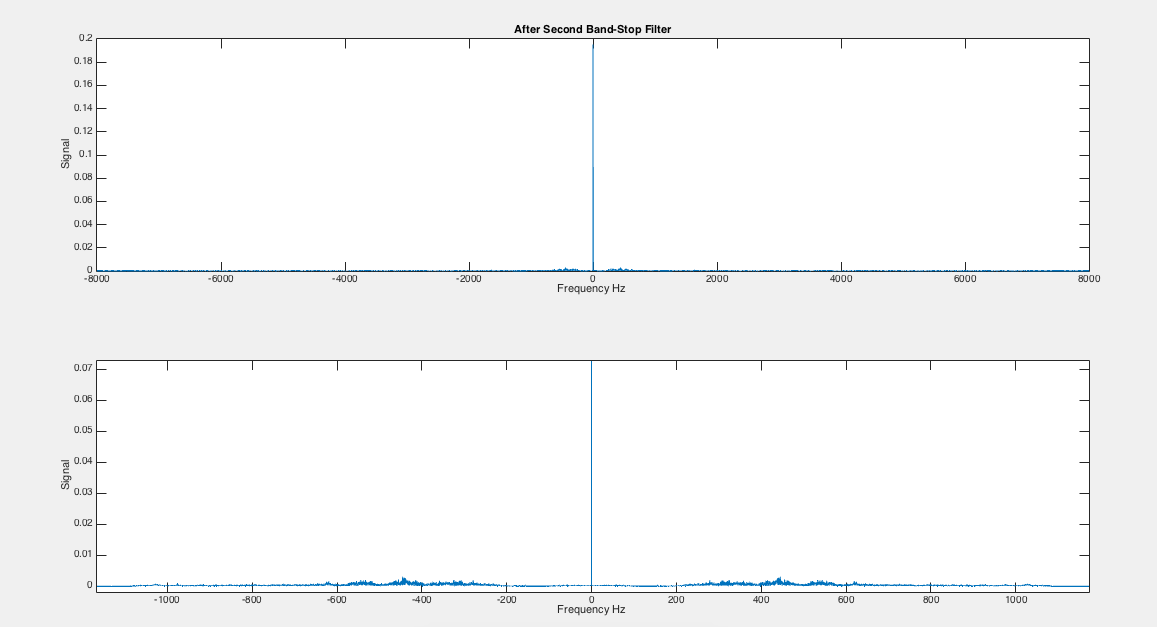


Figure - Impulse Response of the second Band-Stop filter

After removing the second sinusoid, we once again re-plot the updated FFT (Figure 6). There is a very sharp spike right in the middle of the plot, between the 0 and 5 Hz. In order to remove that and leave the rest of the signal unchanged, we use a high-pass filter. This filter will remove all the frequencies below a given cut-off. After trial and error, we are using a cut-off frequency of 100 Hz. Below is the impulse response (Figure 7).



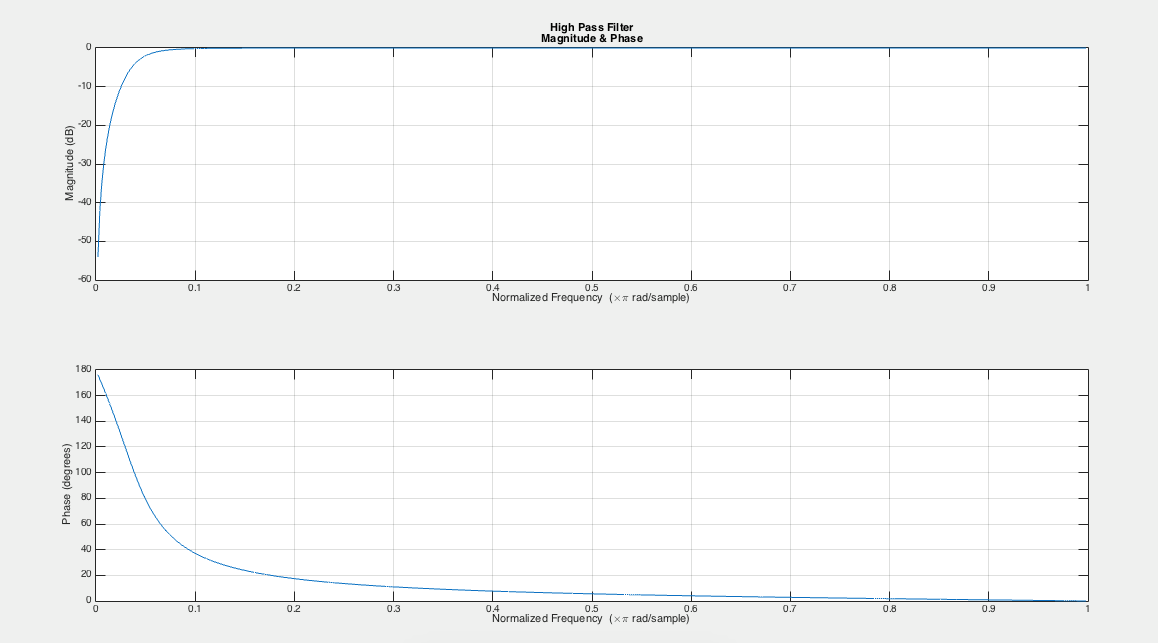


Figure 7 – Impulse Response of High-Pass filter

Figure - FFT after second Band-Stop filter

**Conclusion:**

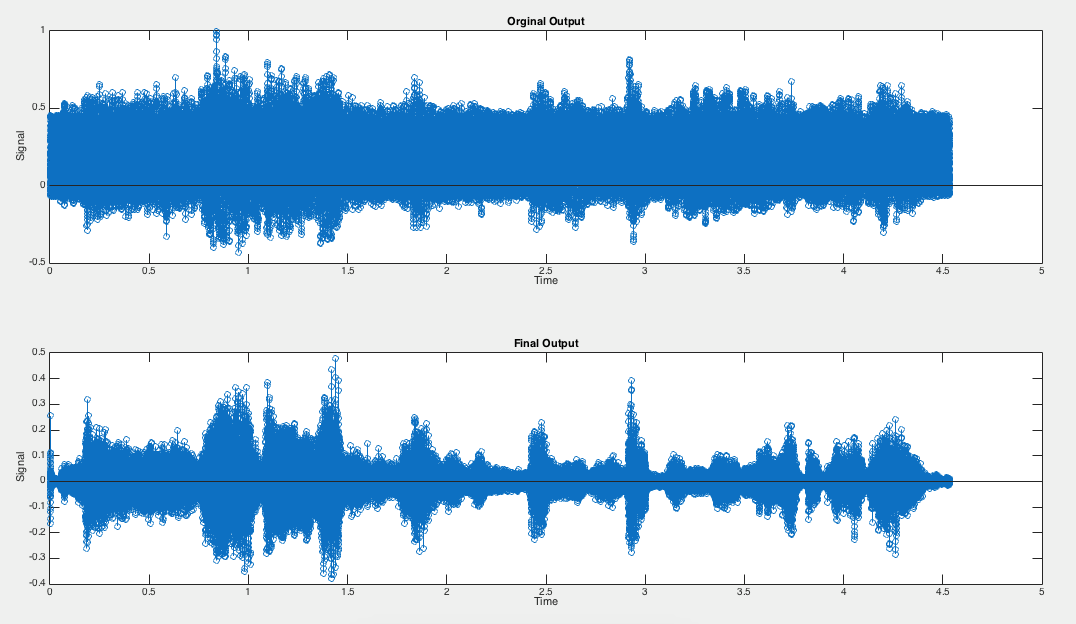
 After passing the signal through the three filters, we plot the output signal (Figure 8). By comparing the two signals, you can see the output is far less corrupt and looks more like a speech signal. The spikes are far more sharp and uniform, showing that the cleanup process was successful.

Figure 8 – Comparison of original signal vs. cleaned up signal

**Appendix:**

clearvars;

close all;

[x,fs]=audioread('Audio\_File\_Fall\_2016.wav');

t = [1/fs:1/fs:length(x)/fs];

figure(1)

stem(t,x)

title('Orginal Output');

ylabel('Signal');

xlabel('Time');

OrigSize = size(x,1);

df = fs / OrigSize;

w = (-(OrigSize/2):(OrigSize/2)-1)\*df;

x1 = fft(x(:,1), OrigSize) / OrigSize; % For normalizing, but not needed for our analysis

x2 = fftshift(x1);

figure(2);

subplot(2,1,1);

plot(w,abs(x2));

title('Orginal Frequncy Response');

ylabel('Signal');

xlabel('Frequency Hz ');

subplot(2,1,2);

plot(w,abs(x2));

ylabel('Signal');

xlabel('Frequency Hz ');

figure(10);

freqz(x,fs);

title({'Orginal','Magnitude & Phase'})

%-------------------------------------------------------------------------%

%BandStop1%

n = 7;

beginFreq = 1050 / (fs/2);

endFreq = 1300 / (fs/2);

[b a] = butter(n, [beginFreq, endFreq], 'stop');

fOut = filter(b, a, x);

x3 = fft(fOut(:,1), OrigSize) / OrigSize;

x4 = fftshift(x3);

figure(3);

subplot(2,1,1);

plot(w,abs(x4));

title('After First Band-Stop Filter');

ylabel('Signal');

xlabel('Frequency Hz ');

subplot(2,1,2);

plot(w,abs(x4));

ylabel('Signal');

xlabel('Frequency Hz ');

figure(4);

freqz(b,a);

title({'First Band-Stop Filter','Magnitude & Phase'})

%BandStop2%

n2 = 3;

beginFreq2 = 50 / (fs/2);

endFreq2 = 300 / (fs/2);

[b2 a2] = butter(n2, [beginFreq2, endFreq2], 'stop');

fOut2 = filter(b2, a2, fOut);

x5 = fft(fOut2(:,1), OrigSize) / OrigSize;

x6 = fftshift(x5);

figure(5);

subplot(2,1,1);

plot(w,abs(x6));

title('After Second Band-Stop Filter');

ylabel('Signal');

xlabel('Frequency Hz ');

subplot(2,1,2);

plot(w,abs(x6))

ylabel('Signal');

xlabel('Frequency Hz ');

figure(6);

freqz(b2,a2)

title({'Second Band-Stop Filter','Magnitude & Phase'})

%HighPass Filter%

n3 = 2;

beginFreq3 = 350 / (fs/2);

[b3 a3] = butter(n3, beginFreq3, 'high');

fOut3 = filter(b3, a3, fOut2);

x7 = fft(fOut3(:,1)\*100, OrigSize) / OrigSize;

x8 = fftshift(x7);

figure(7);

plot(w,abs(x8));

title('After High Pass Filter');

ylabel('Signal');

xlabel('Frequency Hz ');

figure(8);

freqz(b3,a3)

title({'High Pass Filter','Magnitude & Phase'})

%Final Signal%

t=[1/fs:1/fs:length(fOut3)/fs];

figure(9);

subplot(2,1,1);

stem(t,x)

title('Orginal Output');

ylabel('Signal');

xlabel('Time');

subplot(2,1,2);

stem(t,fOut3)

title('Final Output');

ylabel('Signal');

xlabel('Time');

p = audioplayer(fOut3, fs);

play(p);