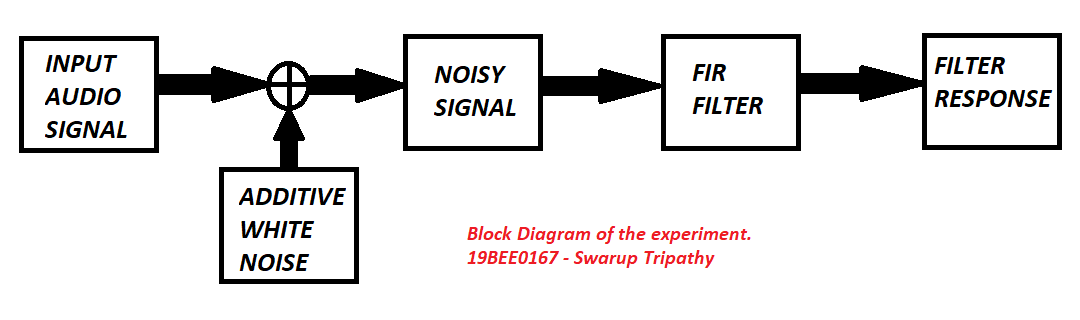
**THEORY:**

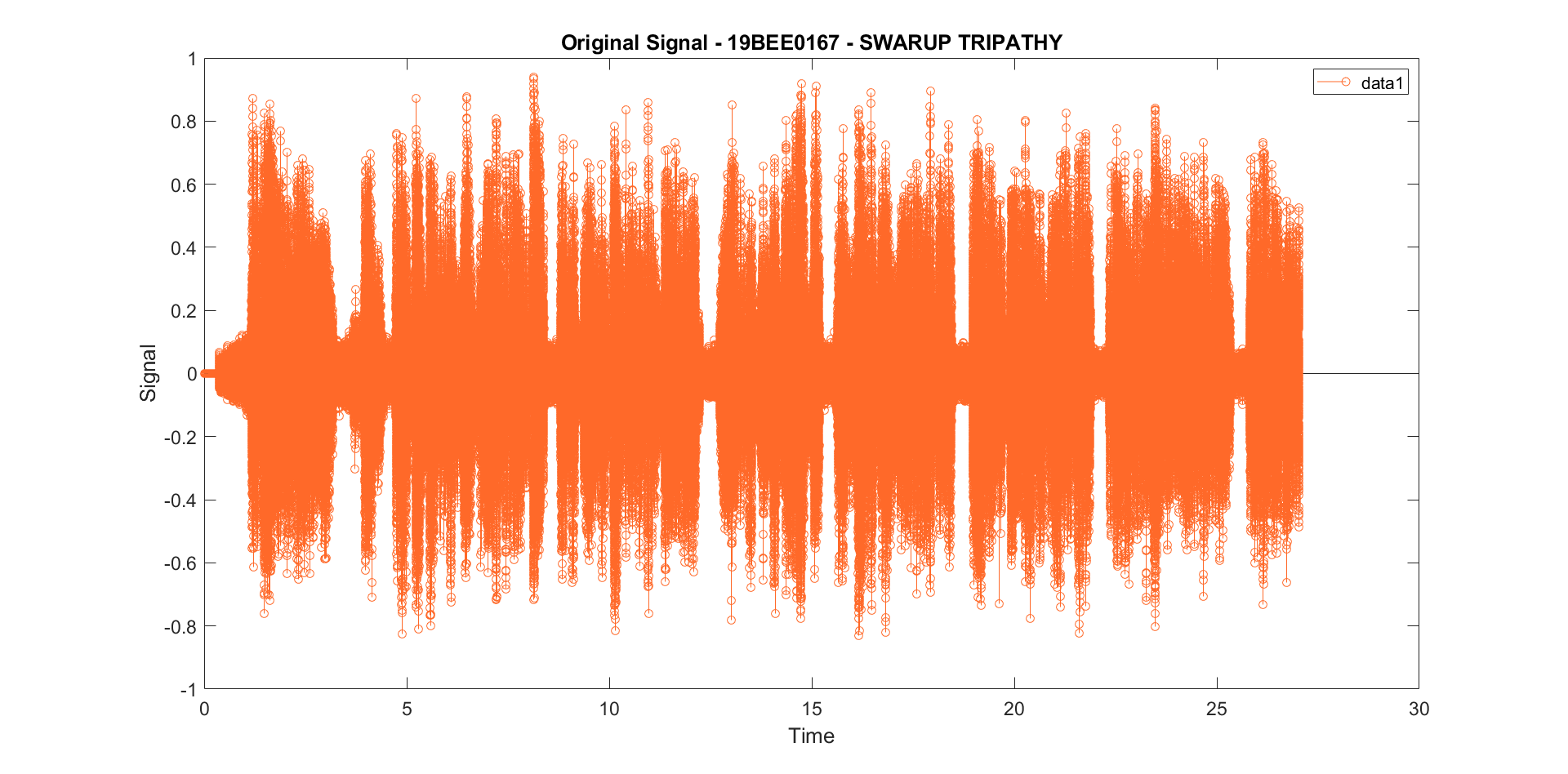
Importance of noise reduction in real time audio signals is said to be having high significance in communication. Noise weakens the signal quality. A well-established method is filtering of the signal in the frequency domain or in the simplest way which is analysing the signal using different filter techniques like low pass, high pass and bandpass filters.

A Fourier transform of a signal gives us the frequency composition of the audio signal. Here, through this challenging experiment I’ll be laying out the procedure I followed to denoise an audio signal which is a commentary during a football match having noise in the background.



**MATLAB program and their corresponding Graphs:** Given Below

|  |
| --- |
| clc; clear all;  close all;  [x,fs]=audioread('C:\Users\iwill\Downloads\sample-cut.wav'); %reading the audio signal    t = [1/fs:1/fs:length(x)/fs];  figure(1)  stem(t,x)  title('Original Signal');  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);  plot(w,abs(x2));  title('Original Frequency Response');  ylabel('Signal');  xlabel('Frequency Hz ');    figure(10);  freqz(x,fs);  title({'Original','Magnitude & Phase'})    %-------------------------------------------------------------------------%    %StopBand1%  n = 1;%7  beginFreq = 1600 / (fs/2);  endFreq = 3500 / (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(x2));  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'})    %StopBand2%  n2 = 4;%3  beginFreq2 = 600 / (fs/2);%600  endFreq2 = 1000 / (fs/2);%900  [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(x4));  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%    Fe2 = fs;  F = [0.0 0.28 0.3 0.5 0.52 1];  M = [0 0 1 1 0 0];  B2 = fir2(30, F,M);  [H2,W2] = freqz(B2,1,100,Fe2);    n3 = 2;  beginFreq3 = 250 / (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'})      t=[1/fs:1/fs:length(fOut3)/fs];  figure(9);  subplot(2,1,1);  stem(t,x)  title('Original 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); |





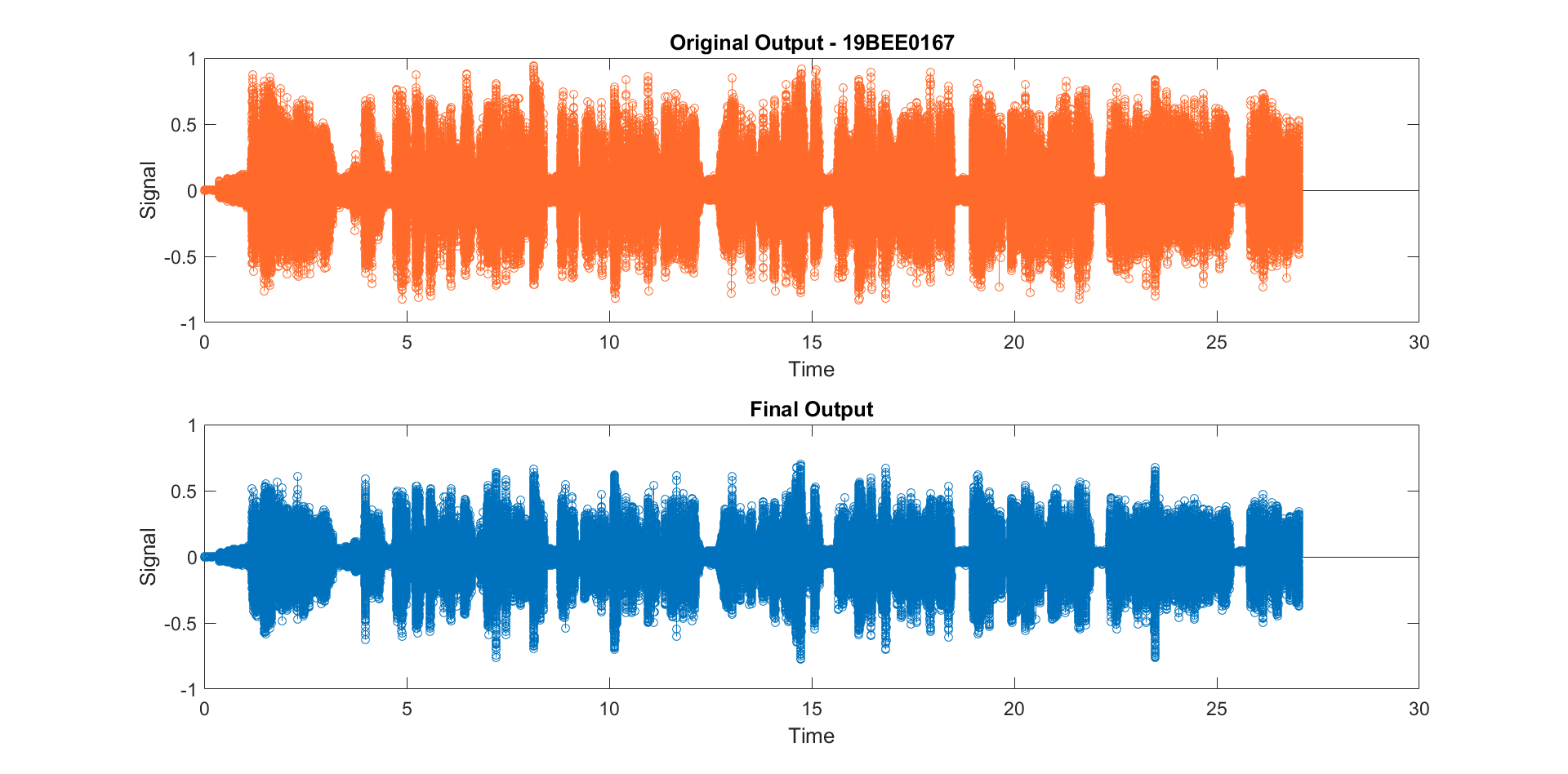










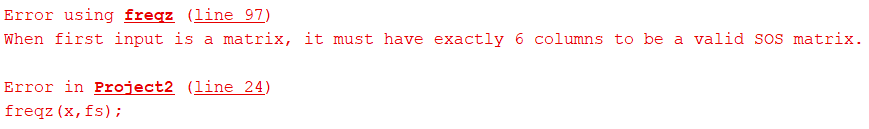


**INFERENCE:**

As was observed from this experiment we were able to achieve quite significant amount of difference with the original audio signal and the output signal. Even though using FIR we were still able to get some amount of noise in the output audio signal. If it had been a single frequency noise over the whole duration of the audio, we might have achieved some good results.

**Problems Faced:**

* Even after using stop band and high pass filter, I was still getting some amount of noise in my output signal
* It was really easy yet difficult at the same time to know which range of frequency to be altered. Though I still think if the frequencies are still altered, I could achieve much better results.
* The selection of audio signal played a vital role since initially I went for my own audio signal but due to some reason, I wasn’t able to rectify the following error



**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**