National University of Sciences and Technology (NUST)
School of Electrical Engineering and Computer Science

EE-232 Signals & Systems

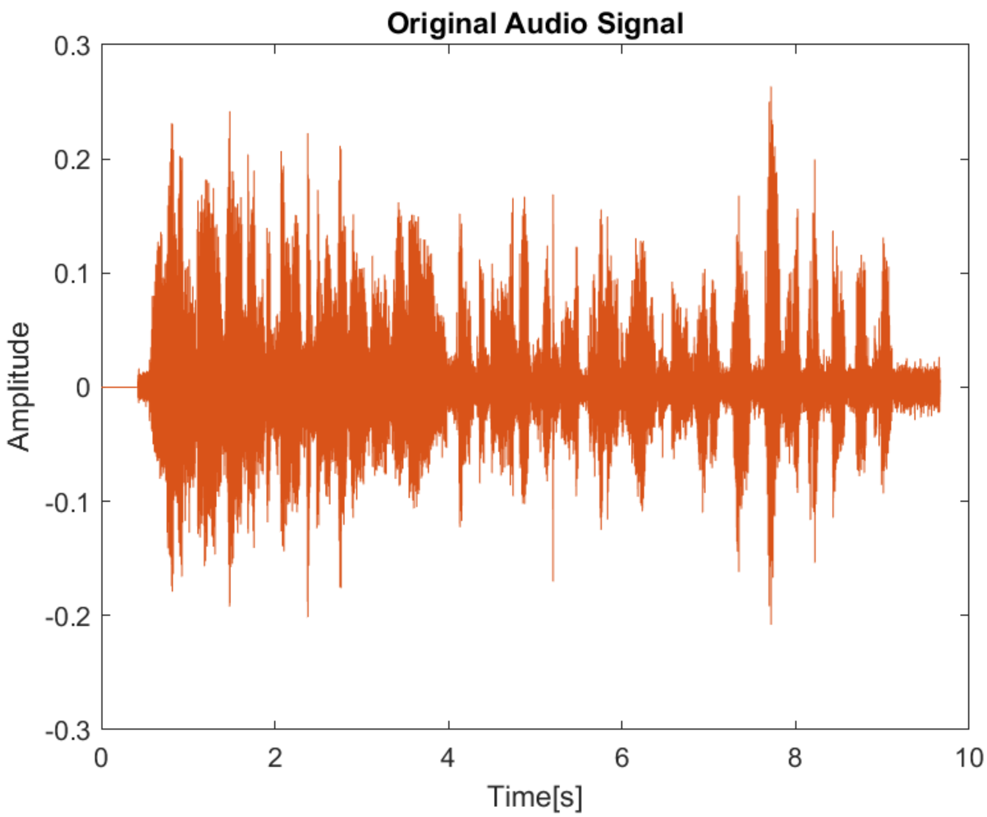
Task 2 Report

Group Members

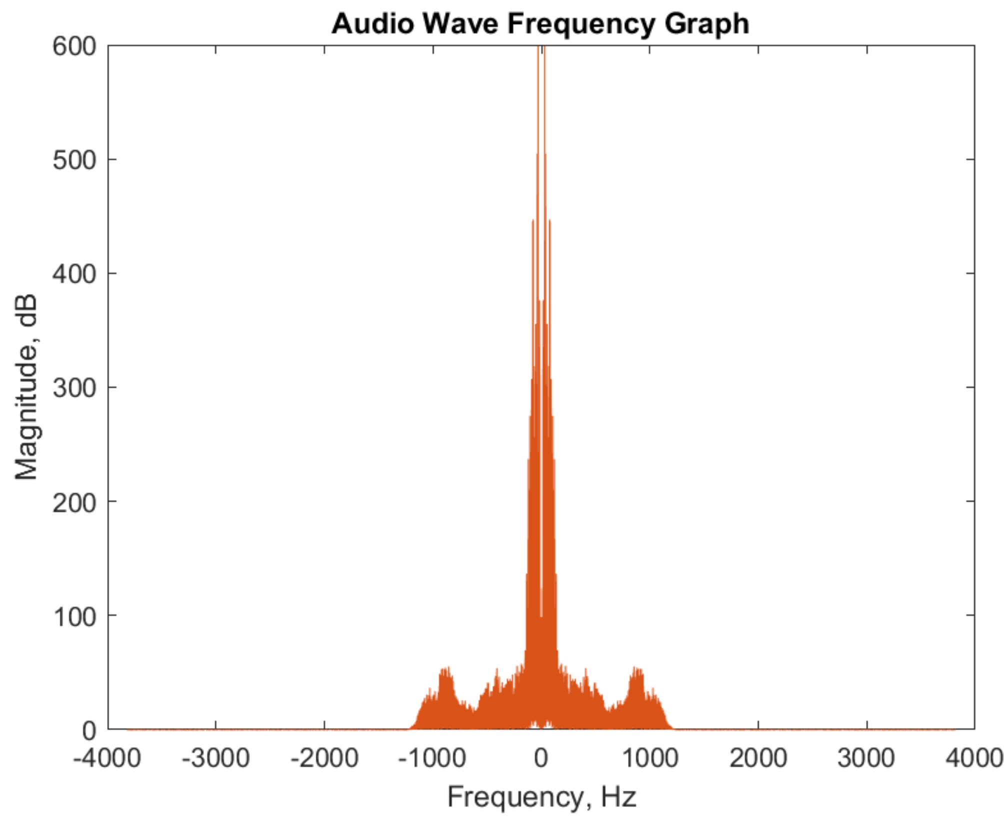
1. Amur Saqib Pal	288334
2. Junaid Ali	290927
3. Faiez Kazi	295848
4. Sannan Abbasi	293786

Steps Performed in Task 2

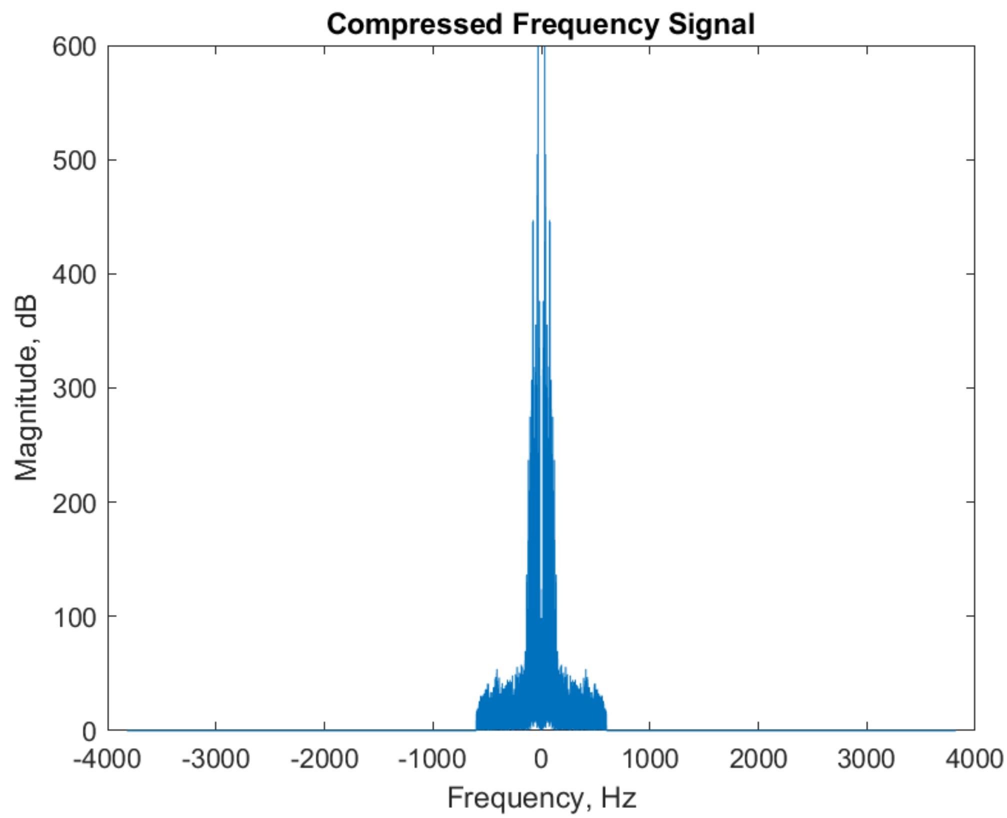
1. Reading of Audio Signal finding its length and plotting it in MATLAB:



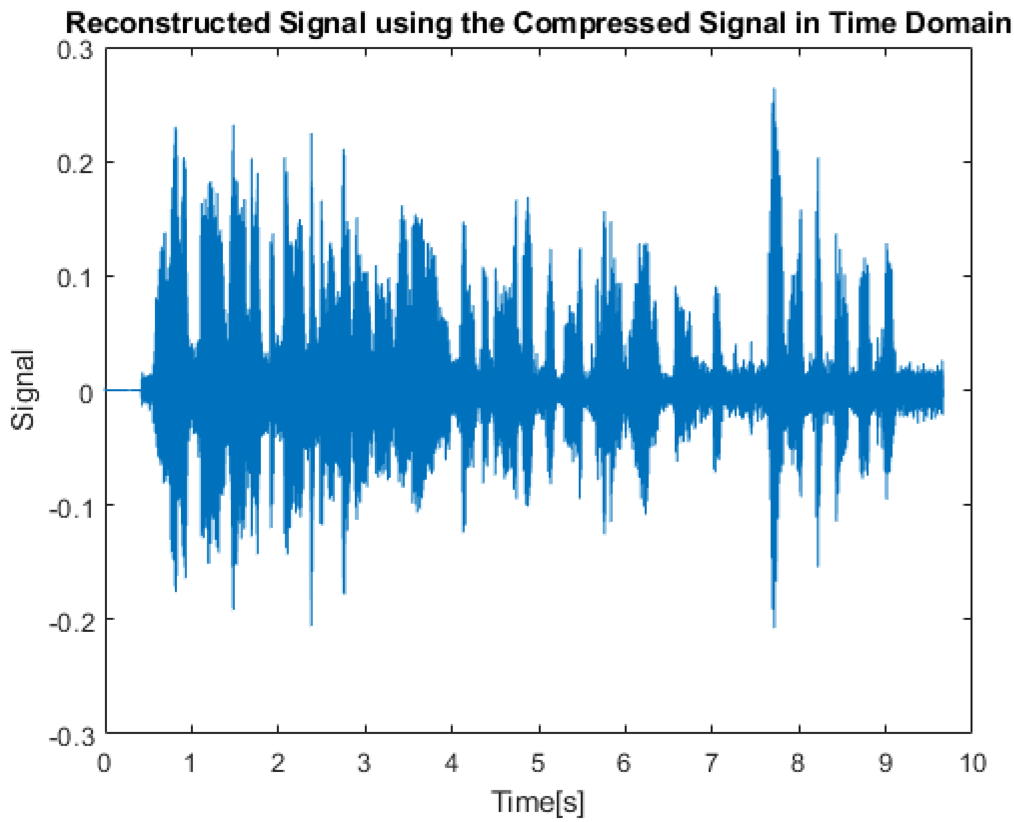
2. Finding the Frequency Spectrum of the signal and plotting it:



3. Finding and Plotting the Compressed Signal:



4. Plotted the Reconstructed Signal using Compressed Signal in Time Domain:



5. Calculated the Compressed Ratio and Distortions in the signals:

Compression rate for L=600:

Compression ratio = $100-(length\ of\ significant\ frequency\ components/length\ of\ original\ signal)(100)$

```
Audio signal was compressed
Chosen L for significant frequency components was  600
Compressed Ratio is : 8.429179e+01%
>>
```

Compression rate for L=100:

Compression ratio = $100-(length\ of\ significant\ frequency\ components/length\ of\ original\ signal)(100)$

```
Audio signal was compressed
Chosen L for significant frequency components was 100
Compressed Ratio is : 9.738161e+01%
>>
```

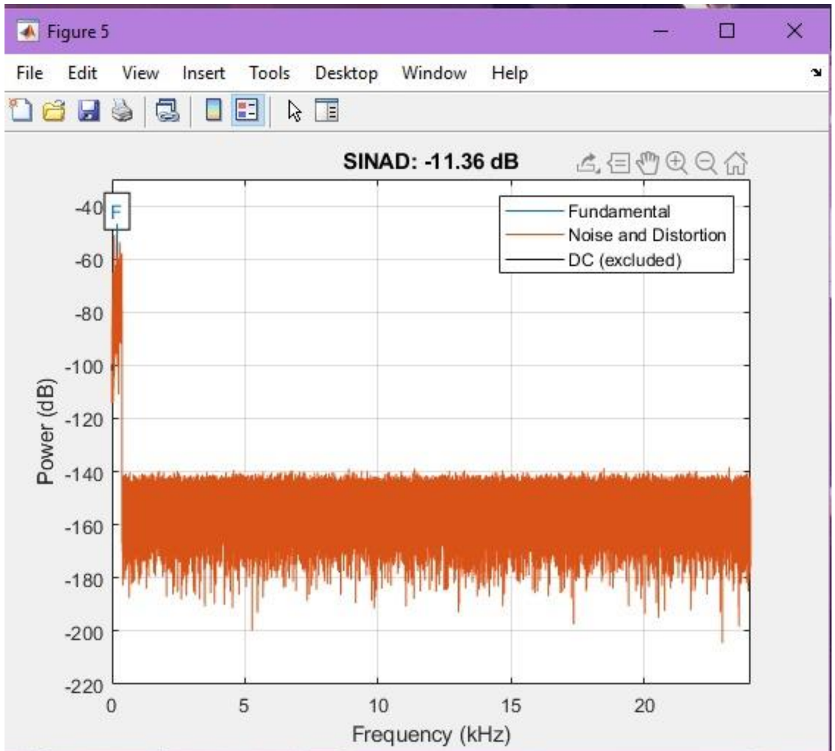
To quantify signal distortion in our reconstructed signal, we used **SINAD,** a function that returns signal to noise distortion ratio.

General syntax is:

$$\text{Sinad}(x, F_s)$$

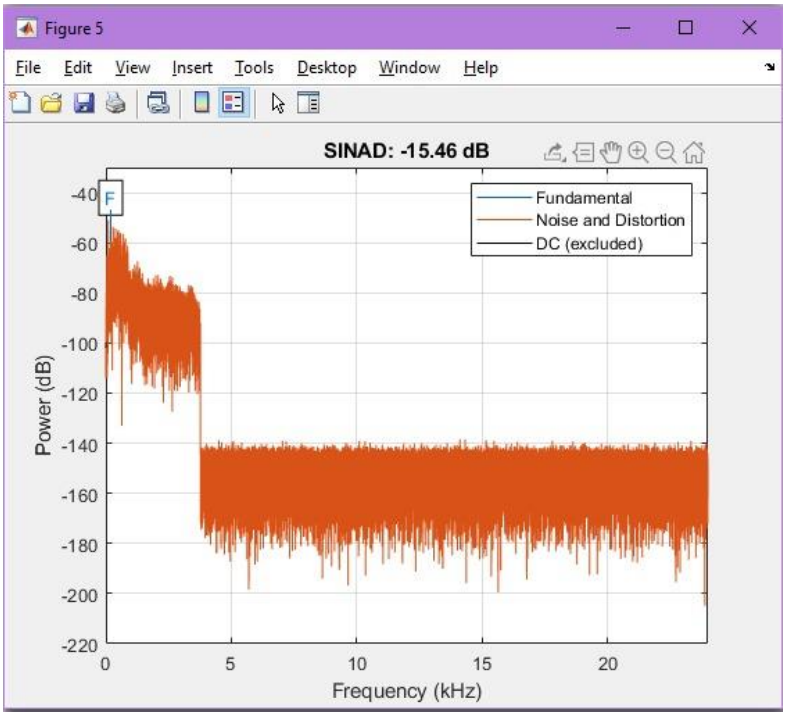
Where x is the input signal(reconstructed audio) and F_s is the sampling rate of the reconstructed signal. The graphs have been plotted and their trend observed in the next part where the DC level and fundamental are excluded as no noise computation is required.

As the value of L is increased, we observe the distortion becomes lesser and vice versa



Here as we can see, for L=60 it is visible from the graph that there is a large amount of distortion. To reduce this distortion, we increase the value of L (our significant frequency components).

Let’s say we increase it up to 600. The resulting distortion plot is shown below:



As we can see, the signal has less distortion and is more understandable.

Value of SINAD is lowered, it is visible from the graph that distortion has surely decreased.

We can increase L, up to a limit where most of its significant components are included in the signal, at this point we get a clear audio, and above that point won't make a very noticeable difference.

Division of Work:

Task 2 of the project was completed by Junaid and Faiez. Both worked together in compiling and formatting the code. They both were responsible for completing the Task 2 report as well.

Code

```
clc;
close all;
clear all;

% *************************Reading of Audio
Signal************************************************
[y,Fs]=audioread('samplewave.wav');

% *************************Finding the length of Signal and Plotting
it*************************
sl = length(y); % length of the wave
Slength = sl/Fs; % time for input signal
t = linspace(0, Slength, sl);
figure;
plot(t,y); %graphing the signal
xlabel('Time[s]')
ylabel('Amplitude')
title('Original Audio Signal')

% *************************Finding and Plotting the Frequency Spectrum of
the Audio Signal*********************
N = length(y);              % Length of vector y, number of samples
Y = fft(y,N);               % Fourier transform of y
Yn= fftshift(Y);
w = ((-N/2:N/2-1)*(Fs/N));  % Frequency vector
F = w/(2*pi);               % Angular frequency vector
Fn = Fs/2;                  % Nyquist Frequency
magnitudeY = abs(Yn);       % Magnitude of the FFT
phaseY = unwrap(angle(Yn)); % Phase of the FFT
figure;
plot(F, magnitudeY);
ylabel('Magnitude, dB');
```

```
xlabel('Frequency, Hz');
title('Audio Wave Frequency Graph');

% ***************************Finding and Plotting the Compressed
Signal******************************************
L=600;                      %Significant Frequency Component Value taken by
US!
start = find(ceil(F) == -L);%this would return a matrix
stop = find(floor(F) == L);
lowindex = start(length(start)); %last value is closest to required frequency
upindex = stop(1); %first value is closest to required values
compressed_y=zeros(1,N);
compressed_y(lowindex:upindex)=Yn(lowindex:upindex);
figure;
plot(F,abs(compressed_y));
ylabel('Magnitude, dB');
xlabel('Frequency, Hz');
title('Compressed Frequency Signal')

% ***************************Plotting the Reconstructred Signal using
Comressed Signal in Time Domain************************
figure;
convert = real(ifft(fftshift(compressed_y)));
plot(t,convert);
xlabel('Time[s]')
ylabel('Signal')
title('Reconstructed Signal using the Compressed Signal in Time Domain')

% ***************************Calculating the Compressed
Ratio*************************************
compressionratio = 100 - ((upindex-lowindex)/N)*100;
fprintf('Audio signal was compressed');
fprintf('\nChosen L for significant frequency components was %d',L);
```

```
fprintf('\nCompressed Ratio is : %d%%\n',compressionratio);

audiowrite('compressedsample.wav',convert,Fs);
[newy,newFs]=audioread('compressedsample.wav');
sound(newy,newFs);
figure;
sinad(newy,newFs)
```