Gunjan K Khut
UID–115813083

Assignment 1

Task 1:

Read the sound file and Plot the signal

1. Selected file **'3.wav'** from the data folder. And used this for the entire assignment.
2. Used **'audio read function'** to read the sound signal and store it in a variable. Then found out the sample rate of the recording. Also, Played the file using the **'soundsc function'**.
3. Then plotted the time domain signal using '**plot function**' with x-axis as time & the y-axis as amplitude.
4. Then plotted the spectrogram of the signal using **'MATLAB's spectrogram function'** with x-axis as time & the y-axis as frequency.

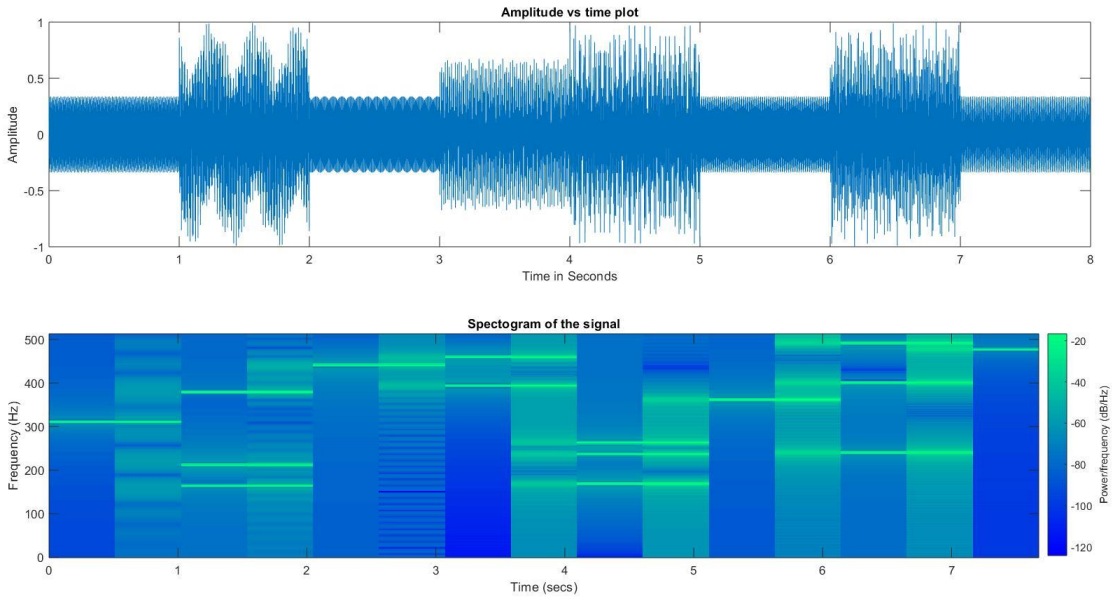

Figure 1: Plots for Task 1

Task 2:

Home-grown STFT

1. Created a 1024-point DFT matrix using the information obtained from the audio signal. It was a complex matrix of dimension (1024 x 1024). Since, it was a complex matrix, splitted the real and imaginary parts and then plotted separately using **'MATLAB's real function'** and **'imag function'**. Now here the problem was, when the figure size was changes, the plots also changes and therefore used a constant figure size to plot both the graphs.
2. Then used my DFT matrix with dimension (1024 x 1024) to find the Short-time Fourier Transform (STFT) of the signal. For this, 8 chunks of the input wave signal were multiplied

Gunjan K Khut
UID–115813083

with the conjugate transpose of the DFT matrix to obtain the 8 STFT matrices (i.e. the **'Z matrix'**). Then, all these 8 - Z matrices were concatenated column wise to obtain a final Z matrix (1024 x 8) that represented the signal which was splitted into 8 seconds.

3. Now, the obtained signal was symmetric, so splitted into two parts and used just the first half of the signal (512 x 8) to plot the final signal.
4. The signal was complex, so its absolute value was used for plotting the spectrogram. Used **'20*log(Z_final)'** to plot the power spectrum of the signal.

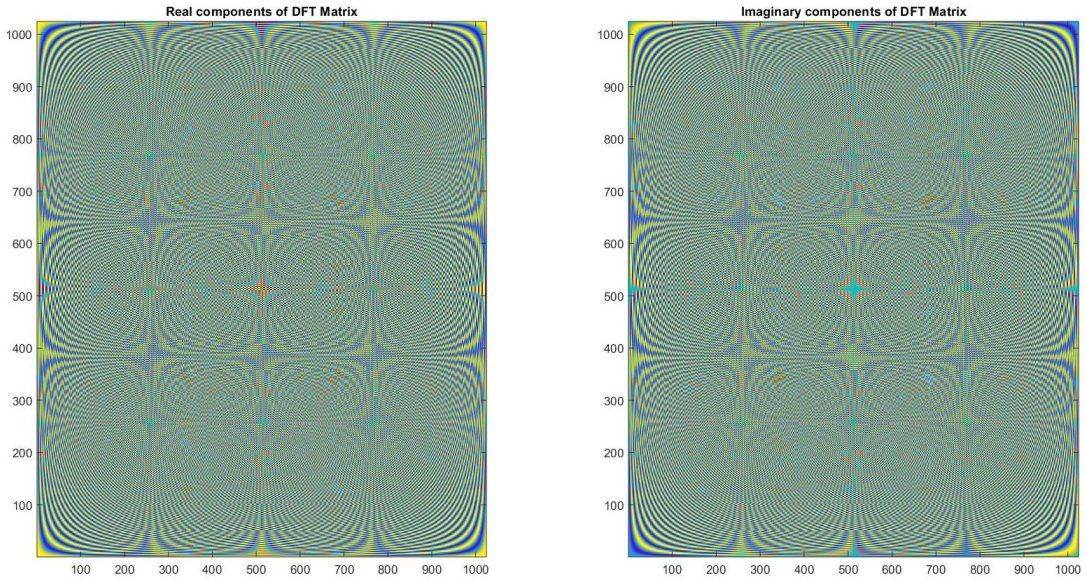

Figure 2. Plot of Real and Imaginary components of DFT Matrix

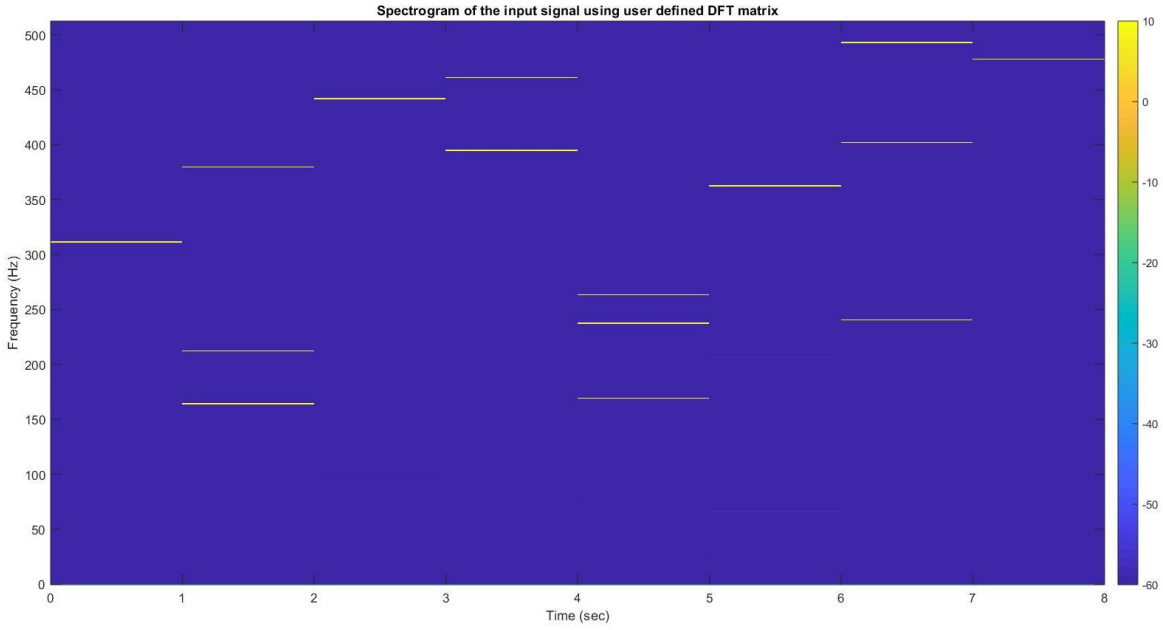

Figure 3. Spectrogram plot of the input signal using user defined DFT matrix

Gunjan K Khut
UID–115813083

Task 3:

Home grown STFT with Overlaps

1. The process was same as Task 2, but here defined analysis parameters for a given overlapping time-windows for the STFT and then plotted the spectrogram.

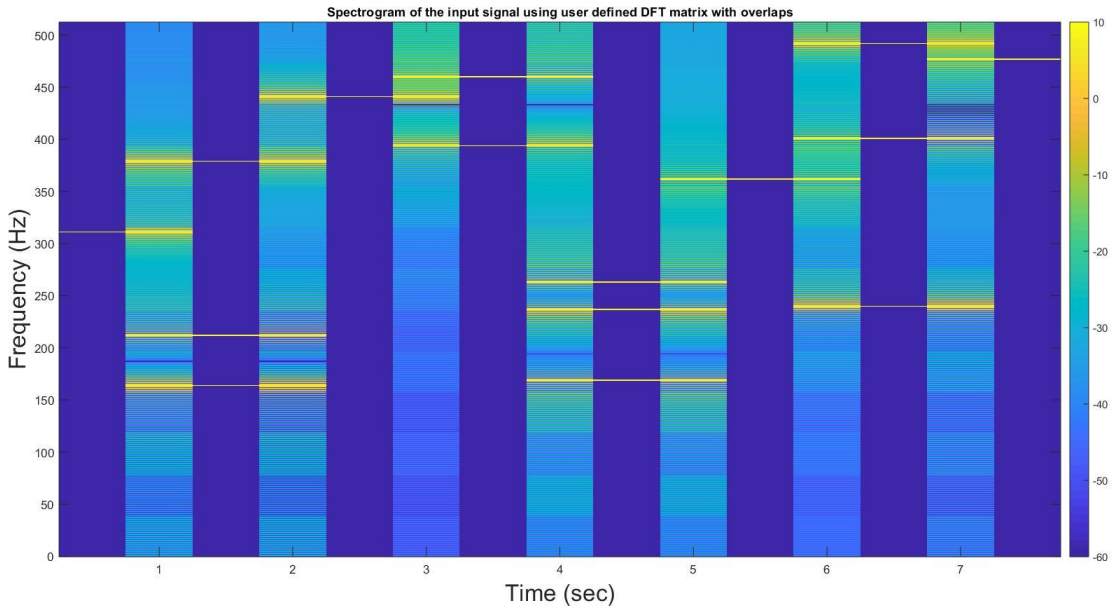

Figure 4. Spectrogram plot of the input signal using user defined DFT matrix with overlaps

Gunjan K Khut
UID–115813083

MATLAB CODE:

Task 1:

```
clear all;
close all;
clc;

%% Load the audio file
[wave, fs] = audioread('3.wav');

%% Play the sound
% wave contains the samples of the audio signal and fs is the sampling
frequency.
% soundsc(wave,fs); %play to check the correct fs

%% Plotting the amplitude vs time plot

figure(1);
t=0:1/fs:(length(wave)-1)/fs;
subplot (2,1,1);
plot(t, wave); title('Amplitude vs time plot');
xlabel('Time in Seconds'); ylabel('Amplitude');

%% Plotting the amplitude vs frequency plot

% n=length(wave)-1;
% f=0:fs/n:fs;
% wavefft=abs(fft(wave));
% subplot(3,1,2);
% plot(f,wavefft); xlabel('frequency in kHz'); ylabel('Amplitude');

%% Compute and plot spectrogram (frequency vs time plot)

 %[x, fs] = audioread('3.wav');
 frame_overlap = 0;
 frame_length  = 512;
 window        = 'hamming'; %15 windows for task 3

 nfft = round(frame_length  * fs / 1000); % convert ms to points
 noverlap = round(frame_overlap * fs / 1000); % convert ms to points
 window   = eval(sprintf('%s(nfft)', window)); % e.g., hamming(nfft)

 [S, F, T, P] = spectrogram(wave, window, noverlap, nfft, fs);
 subplot (2,1,2);
 spectrogram(wave, window, noverlap, nfft, fs, 'yaxis'); % plot
 title('Spectogram of the signal');
 colormap winter;
```

Gunjan K Khut
UID–115813083

Task 2 and Task 3

```
clear all;
close all;
clc;

%% load an audio file
[wave, fs] = audioread('3.wav');    % load an audio file
wave = wave(:, 1);                  % get the first channel

%% Define analysis parameters
windowlength = 1024;

%% Compute the DFT Matrix

N = 1024;
for k=0:N-1
    for l=0:N-1
        w(k+1,l+1)=cos((2*pi*k*l)/N)-1i*sin((2*pi*k*l)/N);
    end
end
dftmatrix = w; % size of DFT matrix(w) is 1024 x 1024

real_w = real(w);
imag_w = imag(w);

%% Plotting the Real and Imaginary components of the DFT Matrix

figure(1);
subplot (1,2,1);
imagesc(real_w);  axis tight;
set(gca,'YDir','normal')
title('Real components of DFT Matrix');
subplot (1,2,2);
imagesc(imag_w), axis tight;
set(gca,'YDir','normal')
title('Imaginary components of DFT Matrix');

%% Reshaping the signal X (N) into 8 equal chunks (N/8) (i.e. X - Matrix)

data1 = reshape(wave(1:1024),[1024,1]);
data2 = reshape(wave(1025:2048),[1024,1]);
data3 = reshape(wave(2049:3072),[1024,1]);
data4 = reshape(wave(3073:4096),[1024,1]);
data5 = reshape(wave(4097:5120),[1024,1]);
data6 = reshape(wave(5121:6144),[1024,1]);
data7 = reshape(wave(6145:7168),[1024,1]);
data8 = reshape(wave(7169:8192),[1024,1]);

%% Taking Conjugate transpose of DFT Matrix (i.e. F matrix)
```

```
F_H = ctranspose(w);

%% Now computing Z matrix (F_H * X matrix) for each chunk, Total of 8.

Z1 = F_H*data1;
Z2 = F_H*data2;
Z3 = F_H*data3;
Z4 = F_H*data4;
Z5 = F_H*data5;
Z6 = F_H*data6;
Z7 = F_H*data7;
Z8 = F_H*data8;

%% Concatenate all the Z matrices into a final Z matrix

%concatenate matrices column wise to get the size of Z_final as 1024*8
Z_final = cat(2,Z1,Z2,Z3,Z4,Z5,Z6,Z7,Z8);

%% Split the Z matrix into to halves.

Z_final_half1 = Z_final(1 : end/2,:);
Z_final_half2 = Z_final(end/2+1 : end,:);

%% Using one half to calculate the Z plot, and using absolute value for
plotting

%Z_final_plot = abs(20*log10(Z_final_half2));
Z_final_plot = abs(Z_final_half1);

t2 = linspace(0.5,7.5,8);
f2 = transpose(0:windowlength/fs:N/2);
% plot(Z_final_plot);axis tight;view(0,90);
% imagesc(abs(Z_final_half1));
% imagesc(t2,f2, 20*log10(Z_final_plot));
% ax=gca;
% ax.YDir='normal';

%% Plot the Spectogram for Task 2

figure(2);
imagesc(t2,f2, 20*log10(Z_final_plot)); colorbar; axis xy;caxis([-60 10]);
F_size = 20;
ax.FontSize = F_size;
title('Spectrogram of the input signal using user defined DFT matrix')
xlabel('Time (sec)', 'FontSize', F_size);
ylabel('Frequency (Hz)', 'FontSize', F_size);
set(gca,'FontSize',10)

%% Task 3

%% Define analysis parameters for a given overlap
```

```
overlap = 512;
windowlength = 1024;
wave_size = 8192;
numberOfWindows = floor(1 + (wave_size - windowlength) / (windowlength -
overlap));

%% Adusting the time and frequency axis as required

t3 = linspace(0.5,7.5,numberOfWindows);
f3 = transpose(0:windowlength/fs:N/2);

%% Computing the time matrix with overlap

X = zeros(N,numberOfWindows);
for j = 1:N
    X(j,1) = wave(j);
end
for i = 2:numberOfWindows
    for j = 1:N
    X(j,i) = wave((i-1)*(N-overlap)+j);
    end
end
%% Compute STFT
Z_final_3 = 1/sqrt(N)*F_H*X;
Z3_H = abs(Z_final_3(1:N/2+1,:));

%% Plot the Spectogram for Task 3
figure(3); imagesc(t3, f3, 20*log10(Z3_H)); colorbar; axis xy;caxis([-60
10]);
F_size = 20;
ax.FontSize = F_size;
title('Spectrogram of the input signal using user defined DFT matrix with
overlaps')
xlabel('Time (sec)', 'FontSize', F_size);
ylabel('Frequency (Hz)', 'FontSize', F_size);
```