Name: Himanshu Singhal

UID: 115891944

Course: CMSC818w

Assignment# : 1

TASK 1: Warming-up

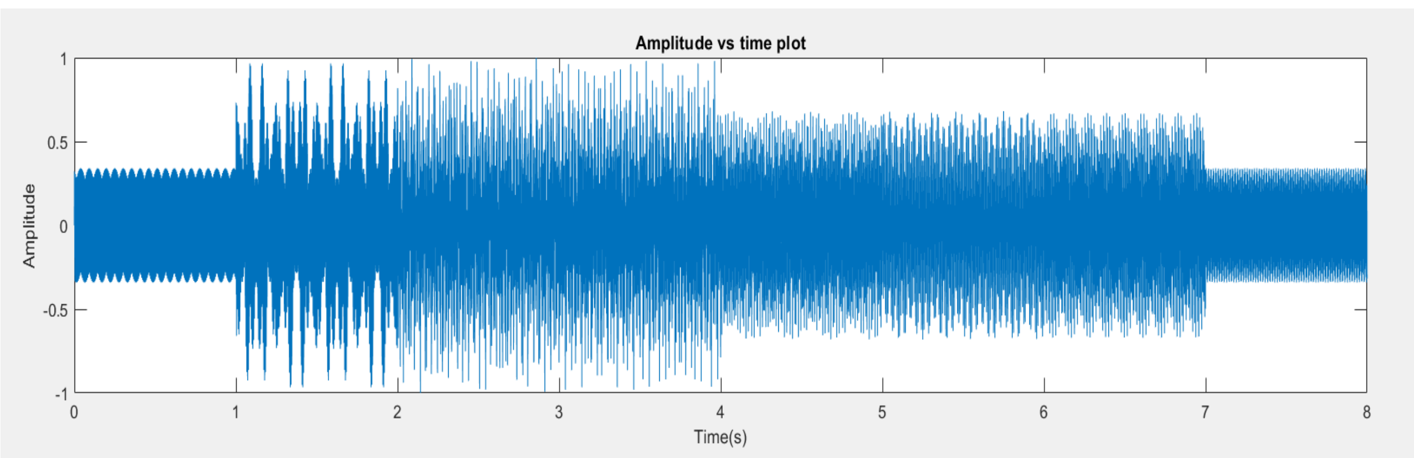
Info - Read the sound file and plot the signal

a) The file with name '4.wav' was uploaded and from the given data set folder
b) The audio file was read and stored in a variable. The sample rate and the sound data were obtained using the inbuilt functions and was stored in a variable

```
[sound, samplingfreq] = audioread('4.wav');
```

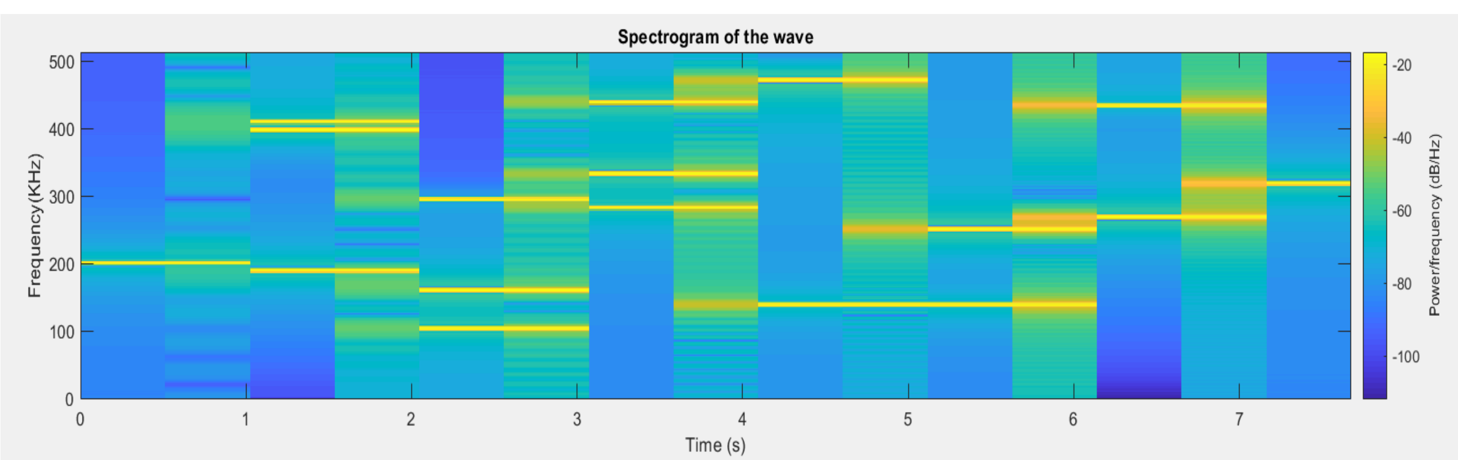
c) The time domain signal was plotted with x axis as time and y axis as amplitude.

```
plot(time, sound);
title('Amplitude vs time plot')
```



d) The spectrogram of the signal was plotted using the inbuilt MATLAB function

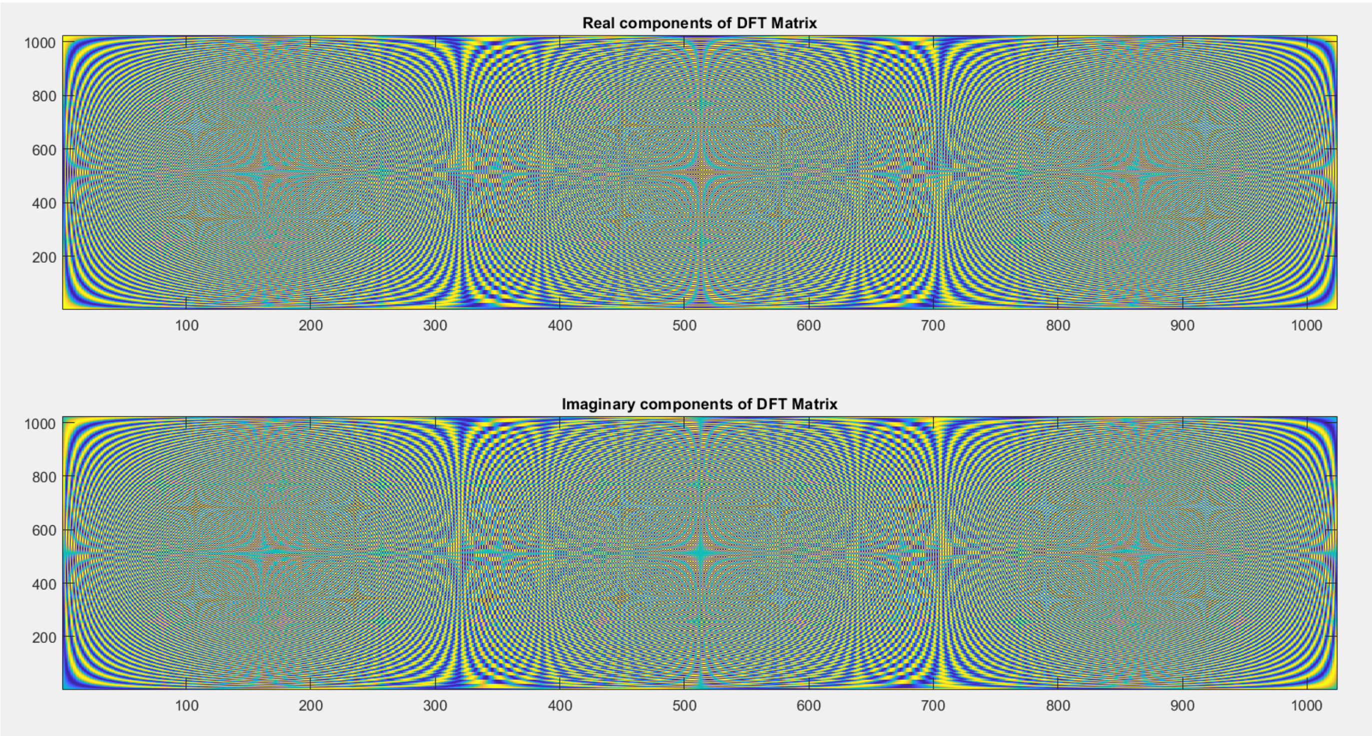
```
spectrogram(sound, window, noverlap, nfft, samplingfreq, 'yaxis');
title('Spectrogram of the wave')
```



TASK 2: HOME-GROWN STFT

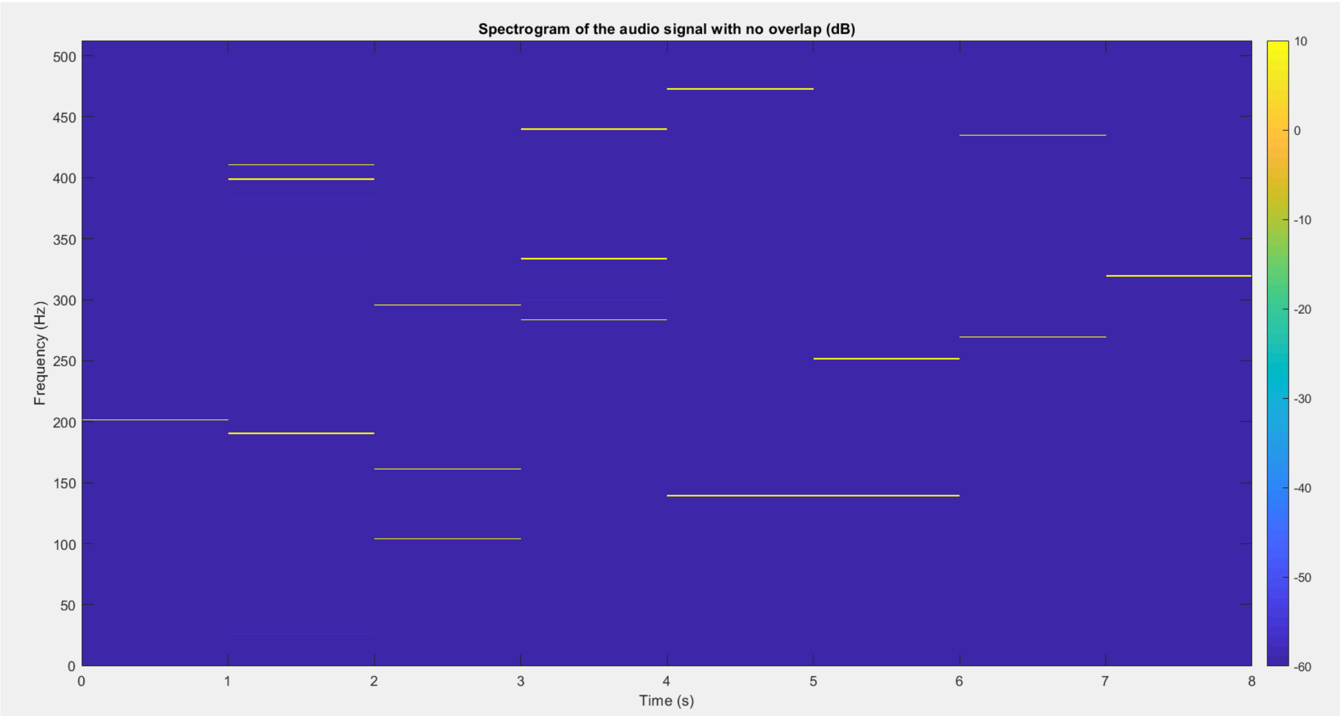
Info - Plot the STFT of the signal using our own DFT matrix

a) A 1024-point DFT matrix was created using the knowledge from the audio signal. Since, it is a complex matrix, the real and imaginary parts were plotted separately using the inbuilt function. As and when the figure size was changed, the plots would change and therefore a constant figure size was used to plot the below graphs.



b) The DFT matrix (1024*1024) was then used to find the STFT of the signal. The input sound signal X (8192,1) was divided into 8 equal parts (representing 8 equal seconds) and each part was multiplied with the conjugate transpose of the DFT matrix to get the STFT matrix Z. Then, all the 8 Z matrices were concatenated column wise to obtain a final Z matrix (1024*8) that represents the signal split into 8 seconds. Now, since the signal is symmetric, only the first half of the signal (512*8) was used to plot the signal. Also, since the signal is complex, its absolute value was calculated before plotting the spectrogram. Lastly, 20*log(final Z) was taken to plot the power spectrum of the signal as shown .

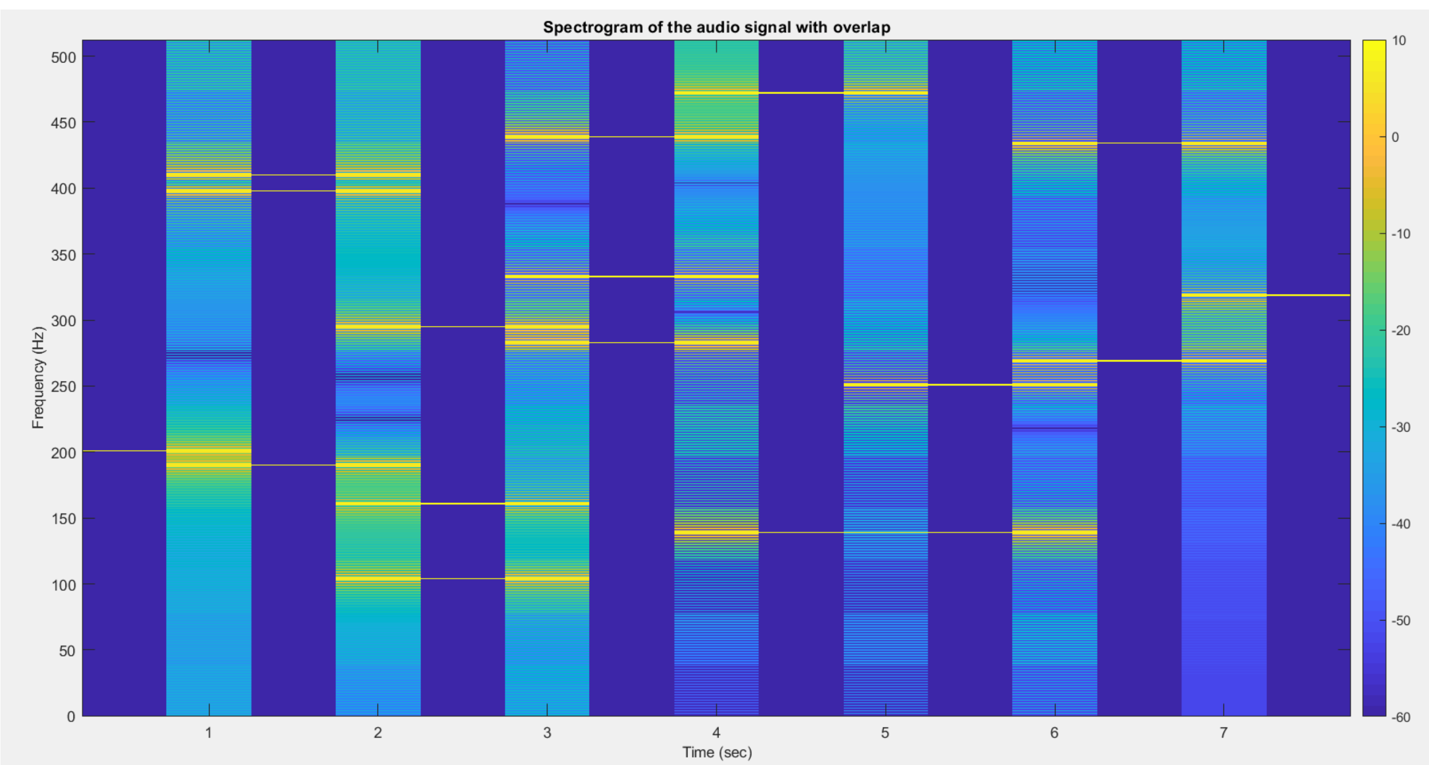
```
x1 = reshape(sound(1:1024),[1024,1]);

F_C = ctranspose(F);

Z1 = F_C*x1;

Z_final = cat(2,Z1,Z2,Z3,Z4,Z5,Z6,Z7,Z8);

Z_final = abs(Z_final);

imagesc(t2,f2, 20*log10(Z_final));
```



TASK 3: HOME-GROWN STFT WITH OVERLAPS

Info - Plot the STFT of the signal using our own DFT matrix, with overlaps

a) The same process as above was used here but overlap was defined separately to plot the spectrogram

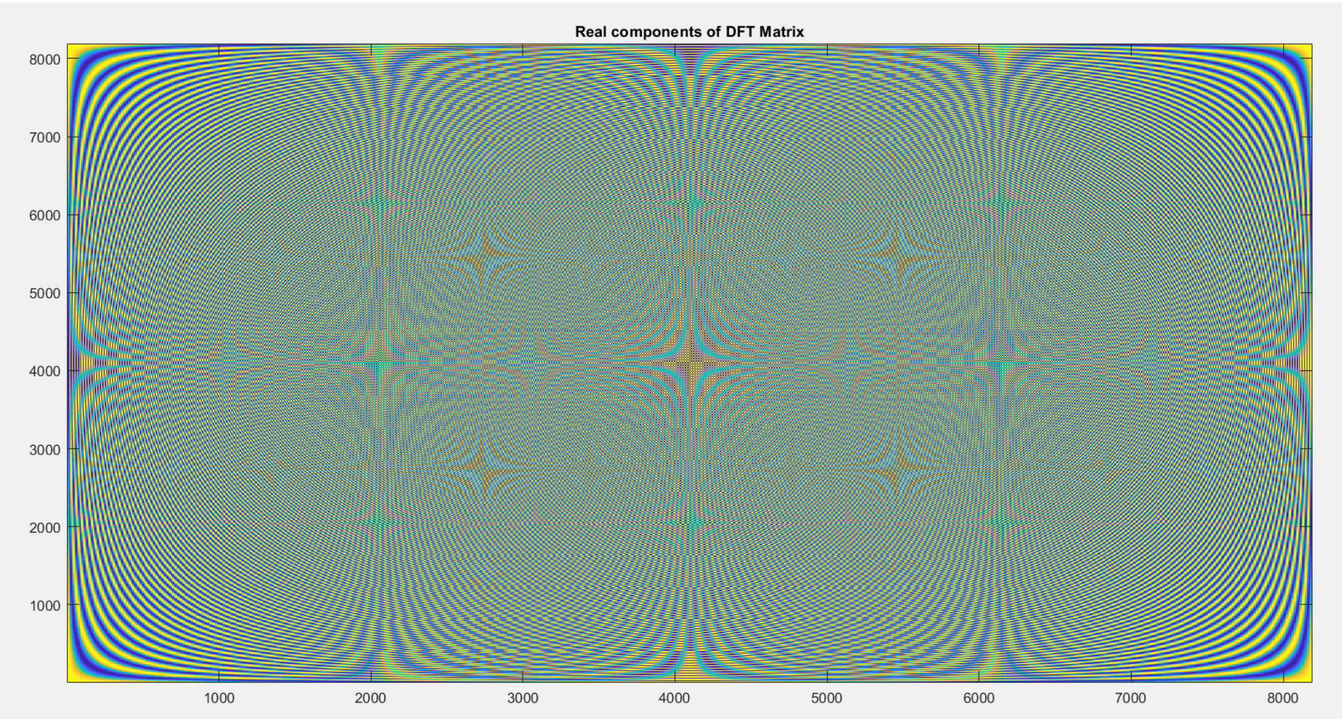


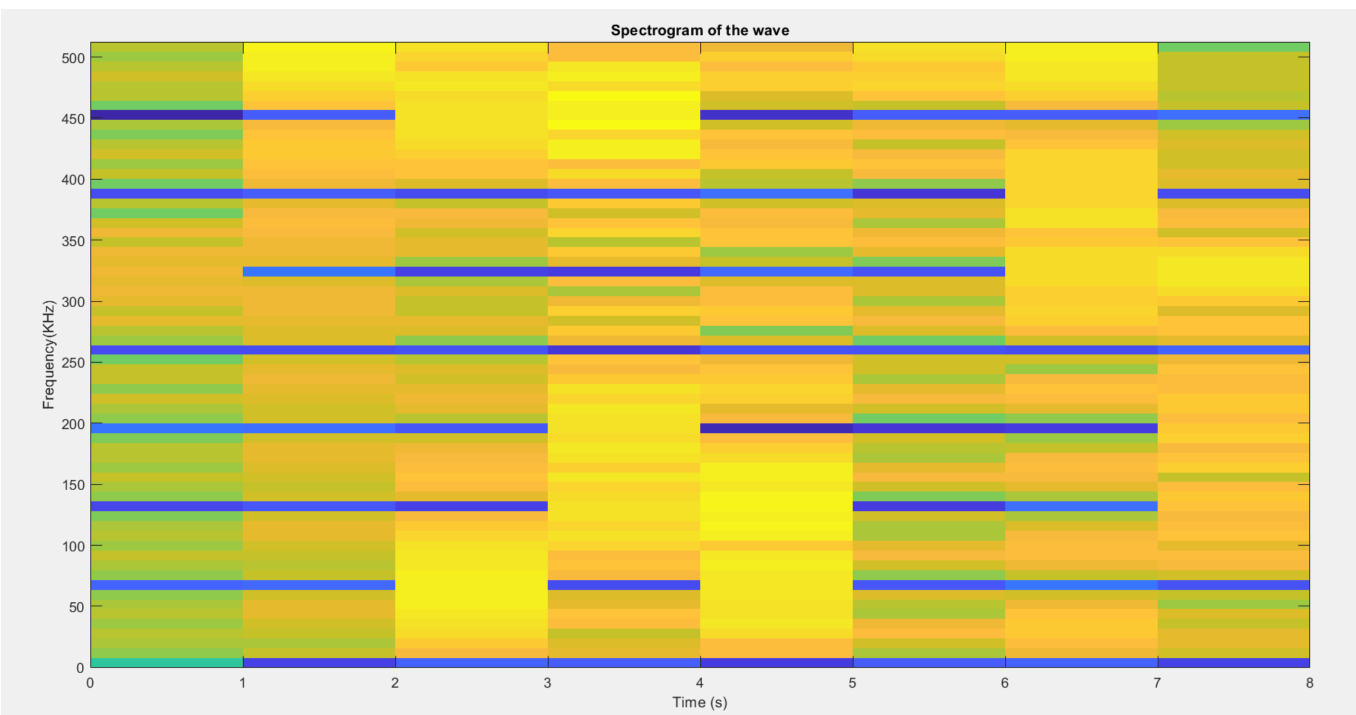
BONUS TASK

Info – obtain the STFT signal using just one multiplication

a) The original DFT matrix was scaled to a higher dimension; from 1024 to 8192 using the inbuilt MATLAB function. Then the resulting DFT matrix was multiplied by the original sound signal and the obtained Z matrix was plotted for the STFT signal.

```
Big_F = kron(F,ones(8));
```





SOURCE CODE/SCRIPT

```
%Himanshu Singhal
%UID  115891944
%Course - CMSC818w
%Assignment 1
%% TASK 1 - WARMING UP
% read the sound file and plot the signal
clc;
clear all;
close all;

% Read the audio file and get sound data and the sampling frequency
[sound, samplingfreq] = audioread('4.wav');

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

% Plotting the amplitude vs time plot
figure(1);
time=0:1/samplingfreq:(length(sound)-1)/samplingfreq;
subplot (2,1,1);
plot(time, sound);
title('Amplitude vs time plot')
xlabel('Time(s)');
ylabel('Amplitude');

% Compute and plot spectrogram (frequency vs time plot)
 fr_overlap = 0; % Initially we want zero overlap
 fr_len  = 512;
 window        = 'hamming'; %15 windows

 nfft = round(fr_len  * samplingfreq / 1000);
 noverlap = round(fr_overlap * samplingfreq / 1000);
 window   = eval(sprintf('%s(nfft)', window));

% Obtain all the required parameters and plot it
 [S, F, T, P] = spectrogram(sound, window, noverlap, nfft, samplingfreq);
 subplot (2,1,2);
 spectrogram(sound, window, noverlap, nfft, samplingfreq, 'yaxis'); % plot
 title('Spectrogram of the wave')
 xlabel('Time (s)')
 ylabel('Frequency(KHz)')
 colormap default;

%% TASK 2 - HOME GROWN STFT
% plot the STFT signal using you own DFT matrix

% use the data from the already loaded audio file
sound = sound(:, 1);     % get all the rows and the first column

% Compute DFT Matrix
% The DFT matrix should be 1024 * 1024 as obtained by the sampling
% frequency from the audio wave
V = 1024;
for m=0:V-1
    for n=0:V-1
        F(m+1,n+1)=cos((2*pi*m*n)/V)-1i*sin((2*pi*m*n)/V);
    end
end
dftmatrix = F; % w is 1024 x 1024 matrix

%obtain the real and imaginary part of the DFT matrix
real_F = real(F);
```

```
imag_F = imag(F);

figure(2);
subplot (2,1,1);
imagesc(real_F);
axis tight;
set(gca,'YDir','normal')
title('Real components of DFT Matrix');
subplot (2,1,2);
imagesc(imag_F);
axis tight;
set(gca,'YDir','normal')
title('Imaginary components of DFT Matrix');
% subplot (3,1,3);
% imagesc(imag_F);
% axis tight;
% title('Imaginary component of the DFT matrix')

% Reshaping the audio signal X (N) into 8 equal chunks (N/8) (i.e. X - Matrix)
% divide the signal (8192) into 8 equal chunks (size-1024) so that we can
% multiply it with the DFT matrix (size-1024)
x1 = reshape(sound(1:1024),[1024,1]);
x2 = reshape(sound(1025:2048),[1024,1]);
x3 = reshape(sound(2049:3072),[1024,1]);
x4 = reshape(sound(3073:4096),[1024,1]);
x5 = reshape(sound(4097:5120),[1024,1]);
x6 = reshape(sound(5121:6144),[1024,1]);
x7 = reshape(sound(6145:7168),[1024,1]);
x8 = reshape(sound(7169:8192),[1024,1]);

% Taking Conjugate transpose of DFT Matrix (i.e. F matrix)
F_C = ctranspose(F);

% Now computing Z matrix (F_H * X matrix)
% Z matrix (1024*1) is a product of the DFT matrix and the audio signal
Z1 = F_C*x1;
Z2 = F_C*x2;
Z3 = F_C*x3;
Z4 = F_C*x4;
Z5 = F_C*x5;
Z6 = F_C*x6;
Z7 = F_C*x7;
Z8 = F_C*x8;

% Concatenate all the Z matrices into a final Z matrix
% the final Z matrix used for plotting the spectrogram should be of the
% size (1024*8), where each column represents 1 second window
% adding all the individual matrices column wise
Z_final = cat(2,Z1,Z2,Z3,Z4,Z5,Z6,Z7,Z8);
Z_final = Z_final/sqrt(V);

% divinding the matrix into two equal halves because it is symmetric
Z_final = Z_final(1 : end/2,:);
% Z_final_half2 = Z_final(end/2+1 : end,:);

% taking the absolute value to get scalar values for plotting
Z_final = abs(Z_final);

t2 = linspace(0.5,7.5,8);
f2 = transpose(0:window/samplingfreq:V/2);

% Plot figure
figure(3);
imagesc(t2,f2, 20*log10(Z_final));
```

```
colorbar;
axis xy;
caxis([-60 10]);
F_size = 20;
ax.FontSize = F_size;
title('Spectrogram of the audio signal with no overlap (dB)')
xlabel('Time (s)', 'FontSize', F_size);
ylabel('Frequency (Hz)', 'FontSize', F_size);
set(gca,'FontSize',10)

%% Task 3 - HOME GROWN DFT WITH OVERLAP
% plot the STFT of the signal using your own DFT matrix but with overlap

% Need a number to track windows for a given overlap
overlap = 512;
window = 1024;
len_sign = 8192;
F_s = samplingfreq;
% round the value to the lower integer
N_window = floor(1 + (len_sign - window) / (window - overlap));

% Create the time and frequency axis
t3 = linspace(0.5,7.5,N_window);
f3 = transpose(0:window/F_s:V/2);

% Matrix with overlap
X = zeros(V,N_window);
for mm = 1:V
    X(mm,1) = sound(mm);
end
for nn = 2:N_window
    for mm = 1:V
    X(mm,nn) = sound((nn-1)*(V-overlap)+mm);
    end
end

% Obtain STFT
% Multiply by 1/sqrt (length(1024)) to normalize the matrix
Znew = 1/sqrt(V)*F_C*X;
% Take its absolute value and plot
Znew = abs(Znew(1:V/2+1,:));

% Plot figure
figure(4);
imagesc(t3, f3, 20*log10(Znew));
colorbar;
axis xy;
caxis([-60 10]);
F_size = 20;
ax.FontSize = 20;
xlabel('Time (sec)', 'FontSize', 10);
ylabel('Frequency (Hz)', 'FontSize', 10);
title('Spectrogram of the audio signal with overlap')

%% Bonus task

Big_F = kron(F,ones(8));
real_Big_F = real(Big_F);
img_Big_F = imag(Big_F);
%D = diag(real_Big_F);
figure(6);
imagesc(real_Big_F);
axis tight;
set(gca,'YDir','normal')
```

```
title('Real components of DFT Matrix');

M = 1024;
Big_F_C = ctranspose(Big_F);
Big_Z = Big_F * sound;
% Big_Z = Big_Z/sqrt(M);
Big_Z = reshape(Big_Z, [1024,8]);
Big_Z = Big_Z(1 : end/2,:);
Big_Z = abs(Big_Z);
t4 = linspace(0.5,7.5,8);
f4 = transpose(0:window/F_s:M/2);
figure(7)
imagesc(t4,f4, 20*log10(Big_Z));
title('Spectrogram of the wave')
xlabel('Time (s)');
ylabel('Frequency(KHz)');
colormap default;
set(gca,'YDir','normal')
```