Assignment 3 Matlab Codes (Q1 and Q2) Report

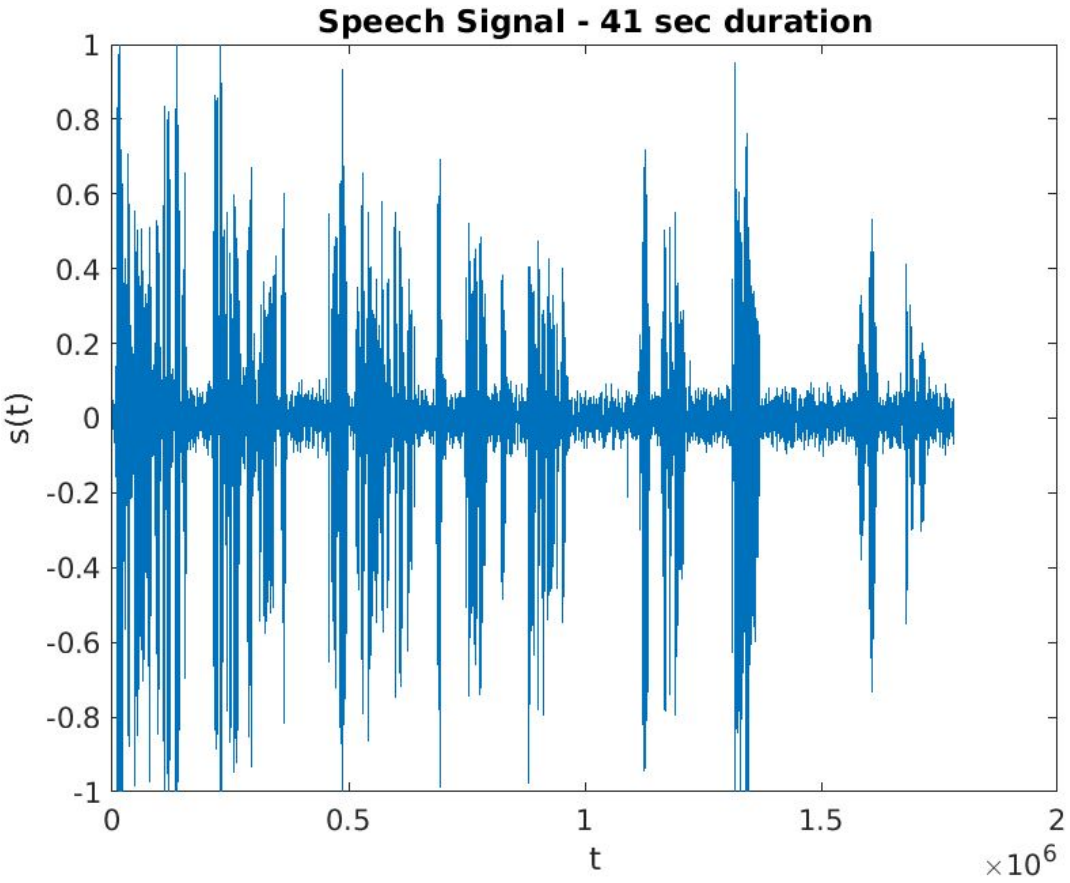
Shivani Chepuri
2018122004

Q1

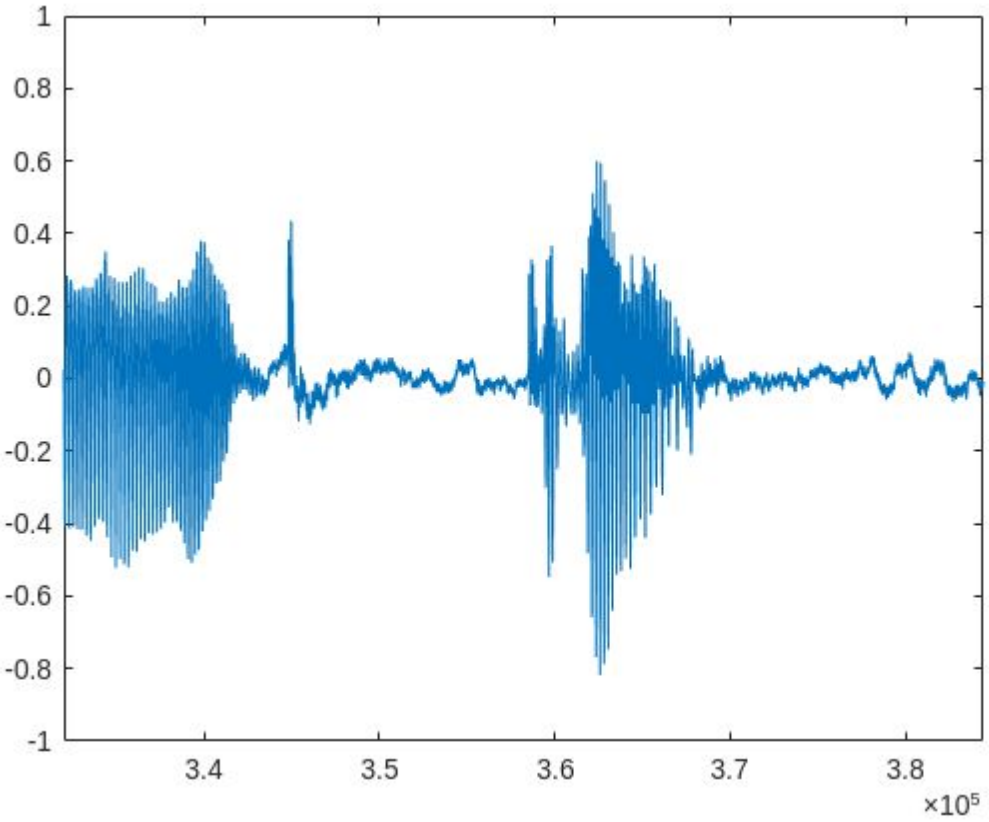
Varying Frame size, keeping Frameshift size same

Speech Signal Time domain plot - own voice recorded
(we cannot apply FFT directly on this signal, therefore not plotting frequency domain signal, we need to do STFT with specified parameters).

Speech signal Plot - Quasi Stationary Signal

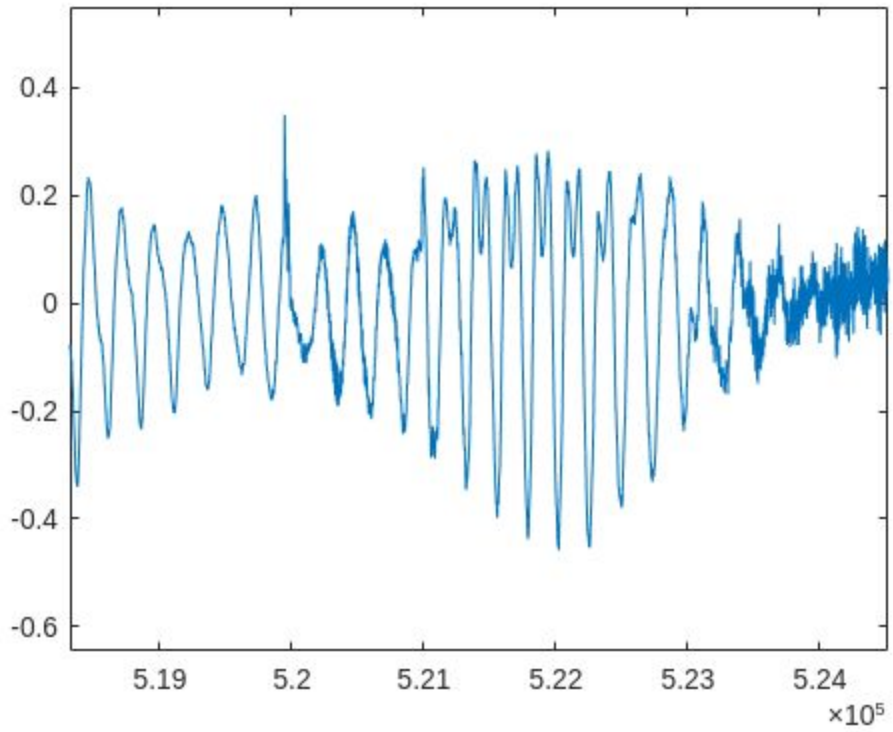


Zoomed in Signal



Here, we can see the stationary (2 lobes of pitch oscillations) and non-stationary parts (due to the gaps created while speaking, and variations of pitch scale).

Pitch Oscillations in the stationary part of the signal (Voiced part)



Stationary part where the signal looks osicallatory, and makes local application of FT possible.

For this question, FFT is taken available default that is, max(256,2^p) points,
where p = [log_2nsc]. Nsc is the number of samples in the time domain signal

Window is also taken as default, that is Hanning Window.
Window length (number of samples in window) and Frameshift size are given as parameters to the function 'tfa1.m'.

Code

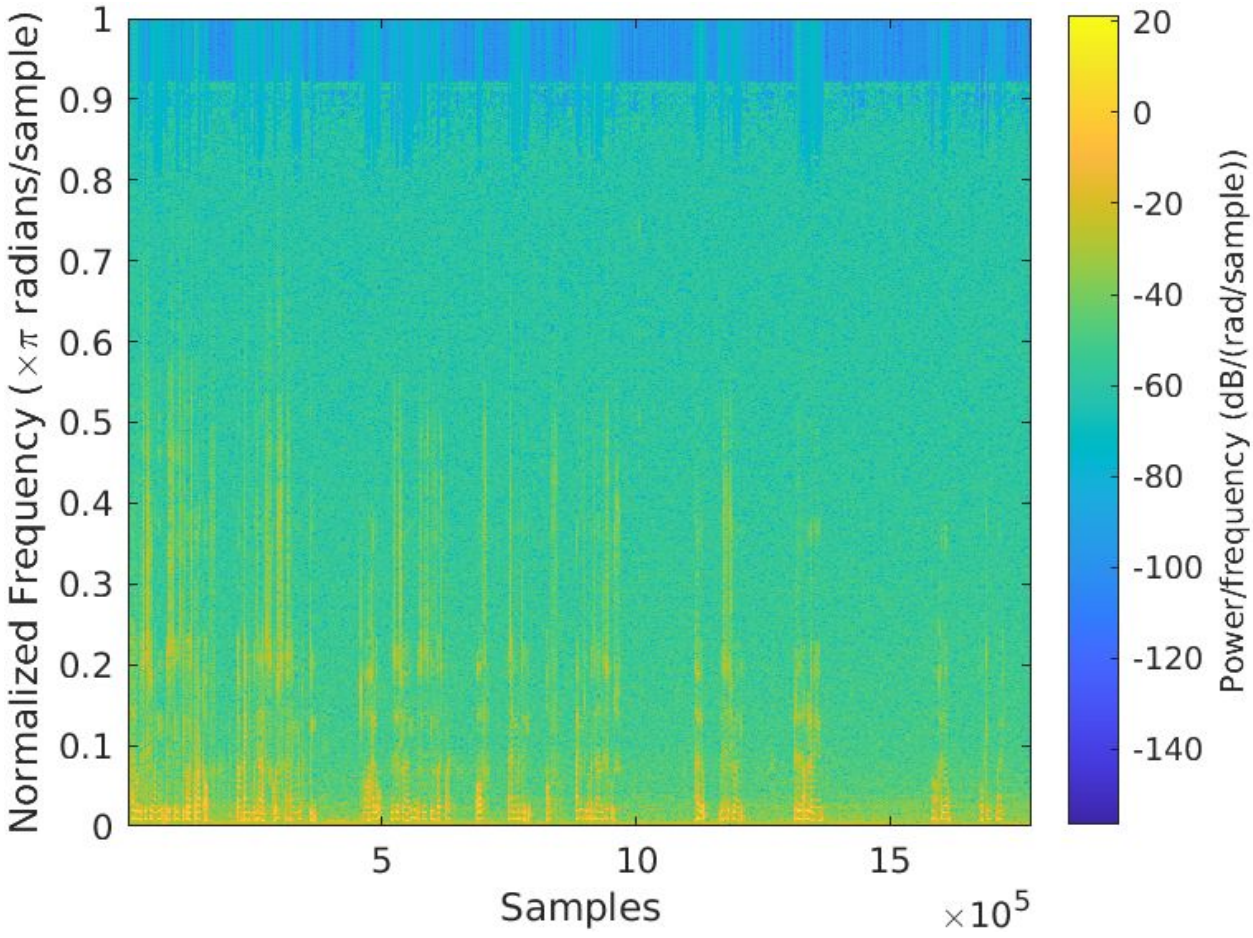
Function - Q1

```
function tfa1(window_duration, frame_shift)
%%

% x is the sample amplitude of the signal
% fs is the sampling frequency of the signal
% spectrogram is the magnitude of STFT of the signal
% for fft purpose, it better to have window length in powers of 2

[x, fs] = audioread('tfa_assg3.wav');
x = x(:, 1); % first channel
% sound(x,fs)
% figure(1);
```

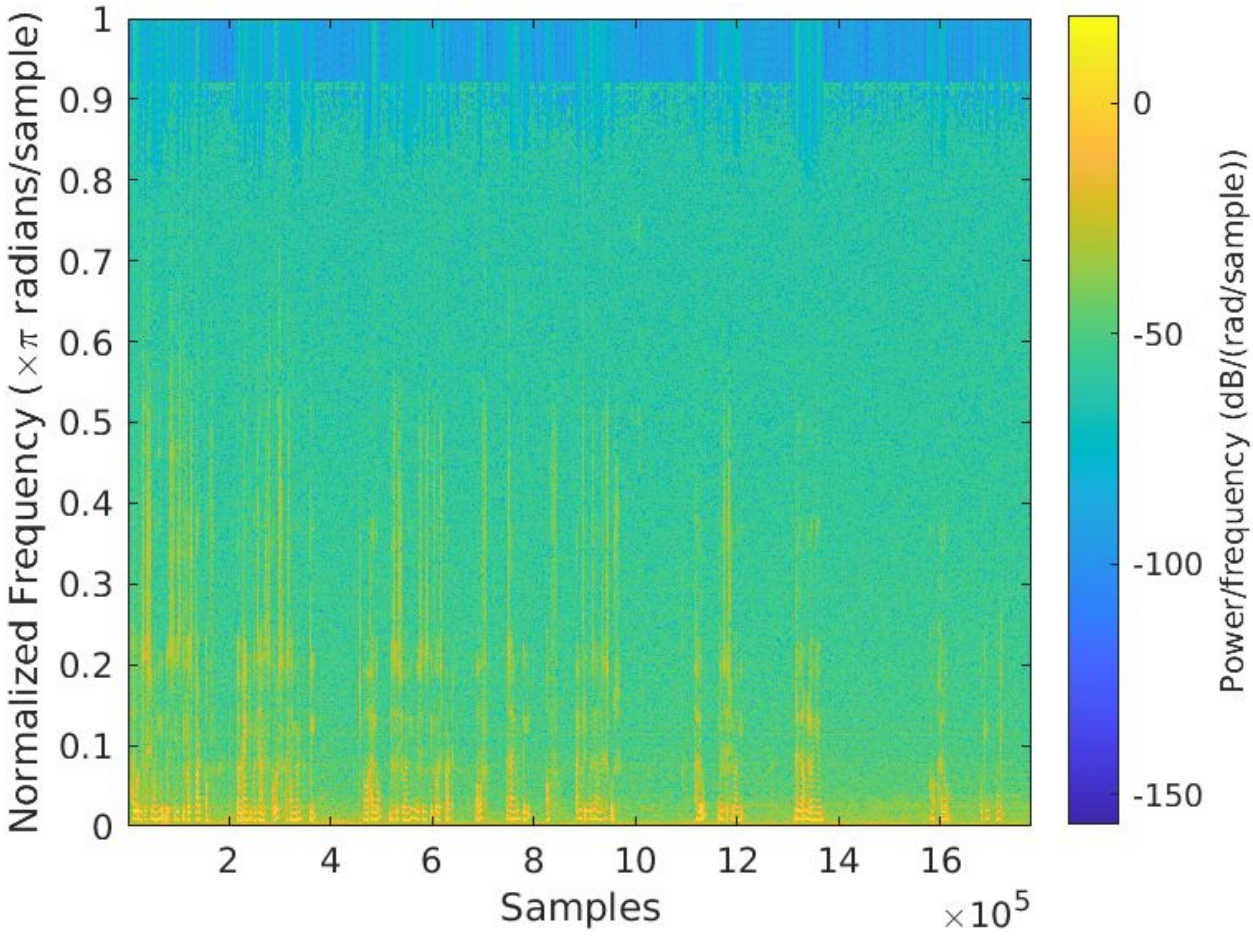
```
% plot(x);

%fs = fs*0.01;
Ts = 1/fs;
len_x = length(x); % number of samples in x
duration_x = len_x * (1/fs); % 1/fs is the time period

% window_duration = 0.1; % frame size (in sec)
% frame_shift = 0.01; % given (in sec)

frame_shift_samples = floor(frame_shift * fs);
frame_size_samples = floor(window_duration * fs);
%num_overlapping_samples = floor(frame_size_samples - frame_shift_samples);
%num_overlapping_samples = max(256, 2^nextpow2(frame_size_samples));

% using default hamming window, default nfft,
spectrogram(x, frame_size_samples, frame_shift_samples, 'yaxis');
```

Spectrogram (Magnitude of STFT) Plots

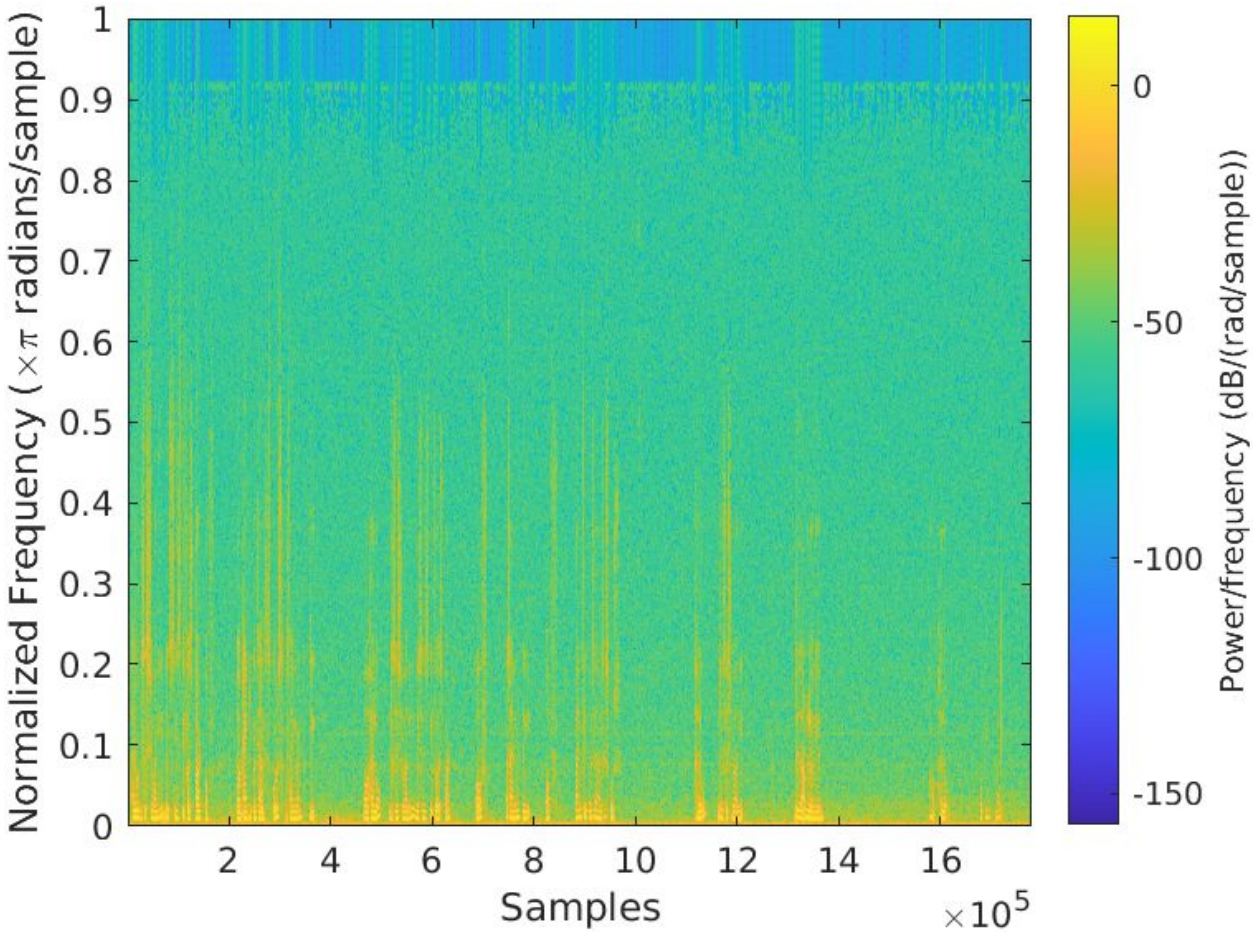
Case 1 - window_duration1 = 100e-3; frame_shift1 = 10e-3;



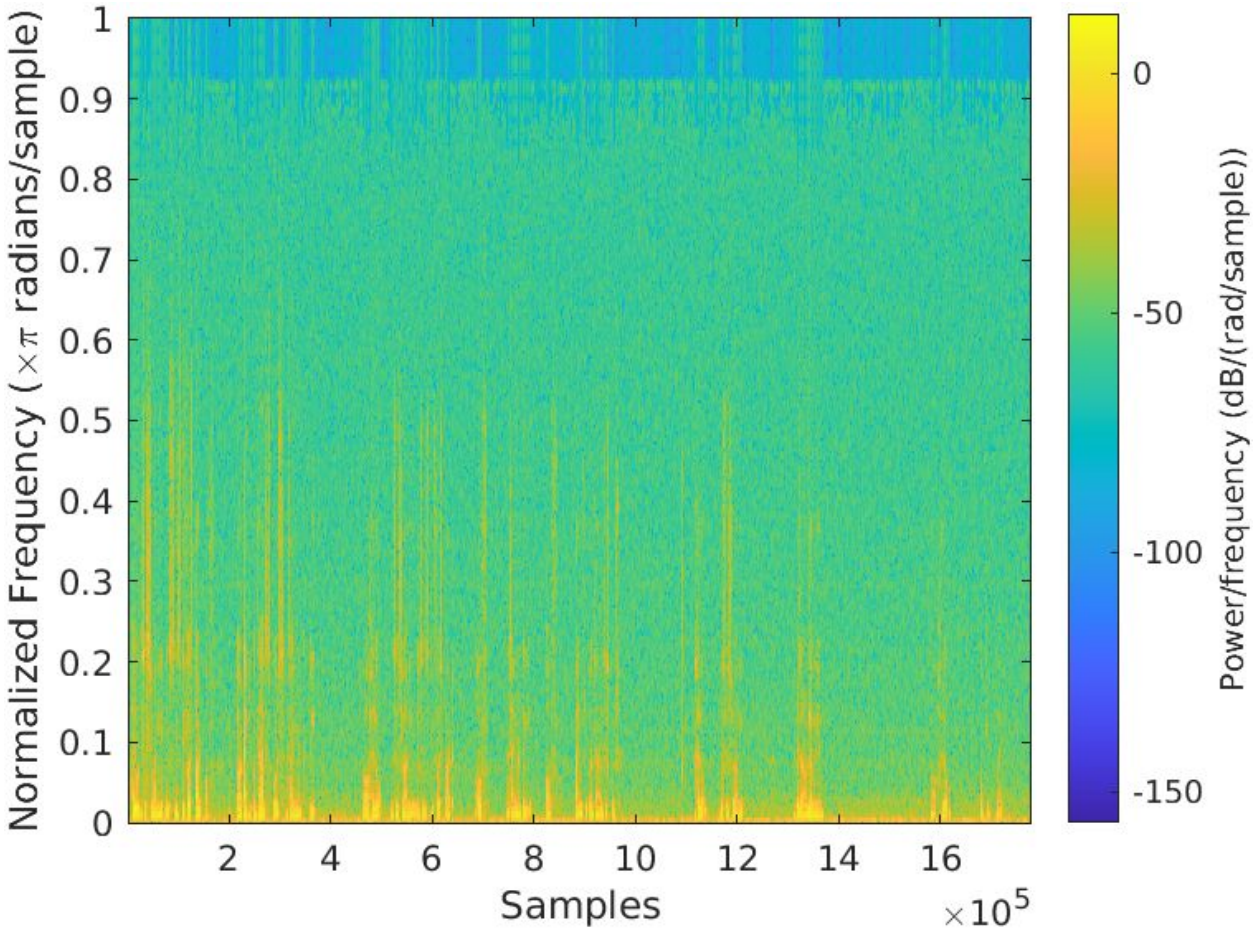
Case 2 - window_duration2 = 50e-3; frame_shift2 = 10e-3;



Case 3 - window_duration3 = 20e-3; frame_shift3 = 10e-3;



Case 4 - window_duration4 = 10e-3; frame_shift4 = 5e-3;



Q2

```
window_duration = 20e-3; % frame size (in sec) - fixed
frame_shift = 10e-3; % given (in sec) - fixed
```

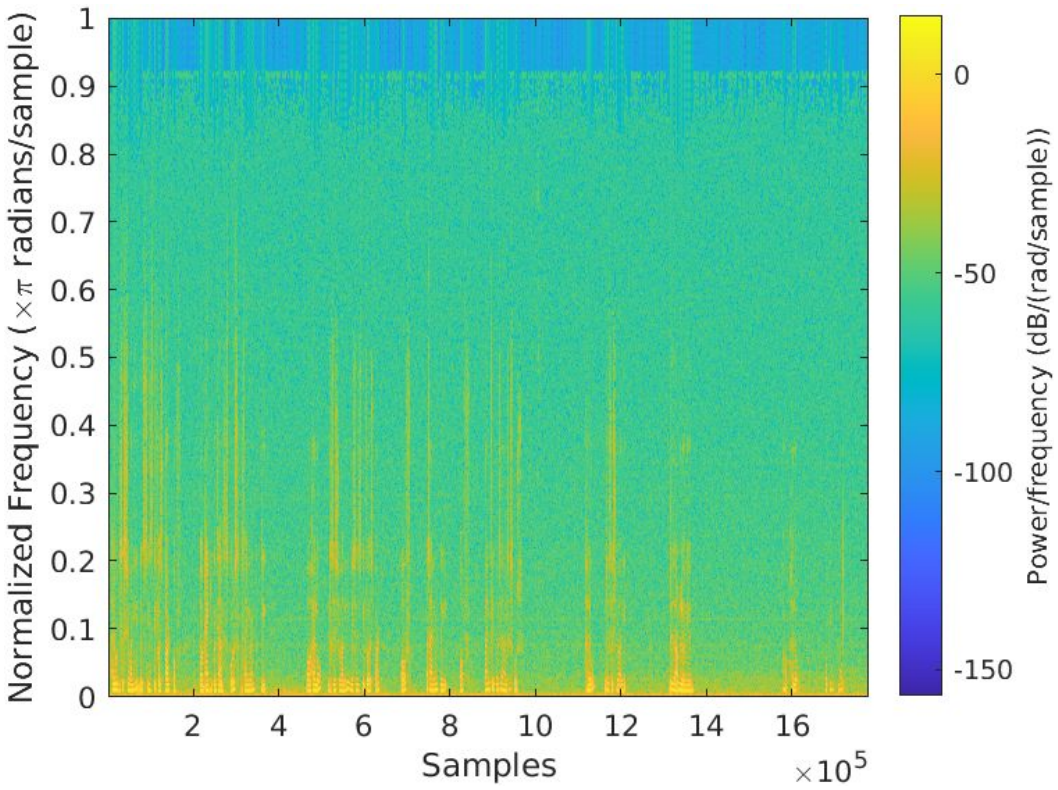
f(ixed)

Function - Q2

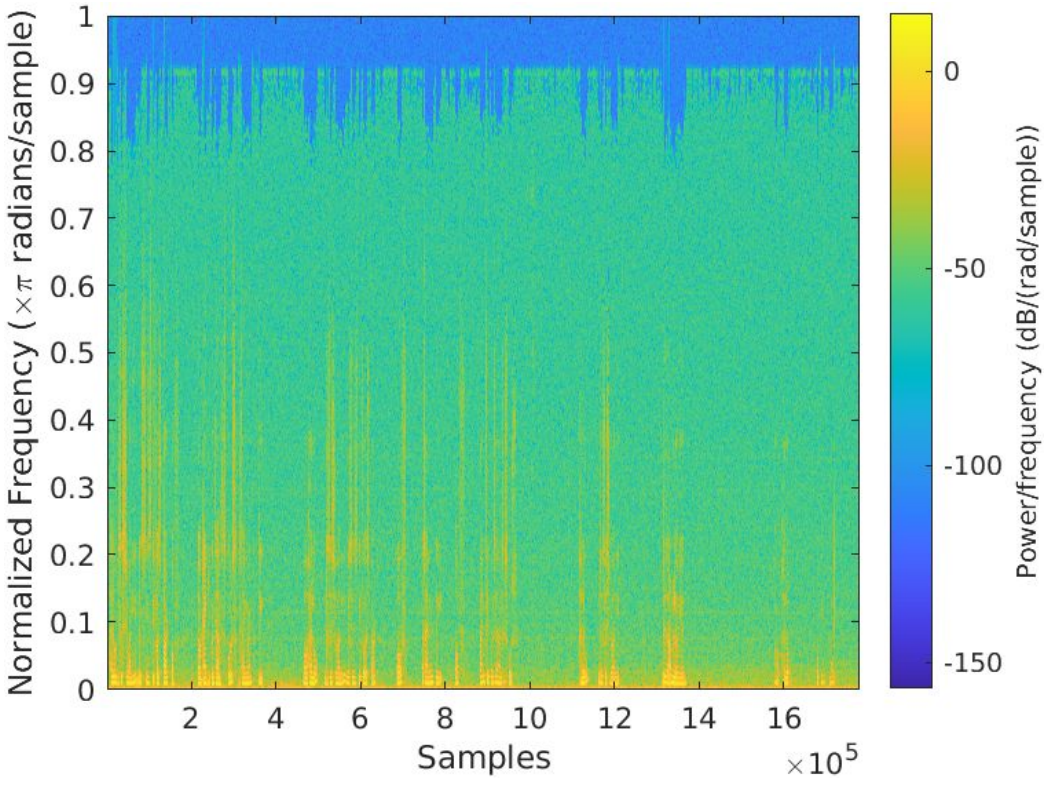
```
function tfa2(window)
% x is the sample amplitude of the signal
% fs is the sampling frequency of the signal
% spectrogram is the magnitude of STFT of the signal
% for fft purpose, it better to have window length in powers of 2

[x, fs] = audioread('tfa_assg3.wav');
x = x(:, 1); % first channel

% sound(x,fs)
% figure(1);
% plot(x);

Ts = 1/fs;
```

```
len_x = length(x); % number of samples in x
duration_x = len_x * (1/fs); % 1/fs is the time period

window_duration = 20e-3; % frame size (in sec) - fixed
frame_shift = 10e-3; % given (in sec) - fixed

frame_shift_samples = frame_shift * fs;
frame_size_samples = window_duration * fs;
num_overlapping_samples = floor(frame_size_samples - frame_shift_samples);

% Using default nfft

% window is 1 implies use hamming window
if (window == 1)
spectrogram(x,hamming(frame_size_samples),frame_shift_samples,[],'yaxis');
end

% window is 2 implies use hanning window
if (window == 2)
spectrogram(x,hann(frame_size_samples),frame_shift_samples,[],'yaxis');
end

% window is 3 implies use rectangular window
if (window == 3)
spectrogram(x,rectwin(frame_size_samples),frame_shift_samples,[],'yaxis');
end

% window is 4 implies use hamming window
if (window == 4)
spectrogram(x,triang(frame_size_samples),frame_shift_samples,[],'yaxis');
end

end
```

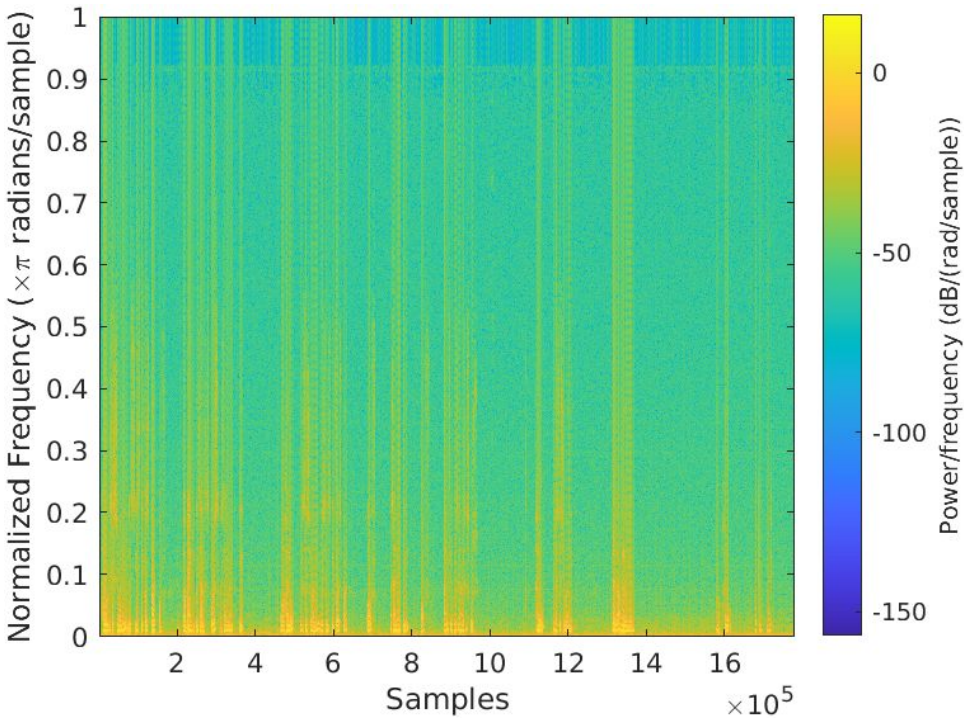
Spectrogram (Magnitude of STFT) Plots
Case 1 - hamming



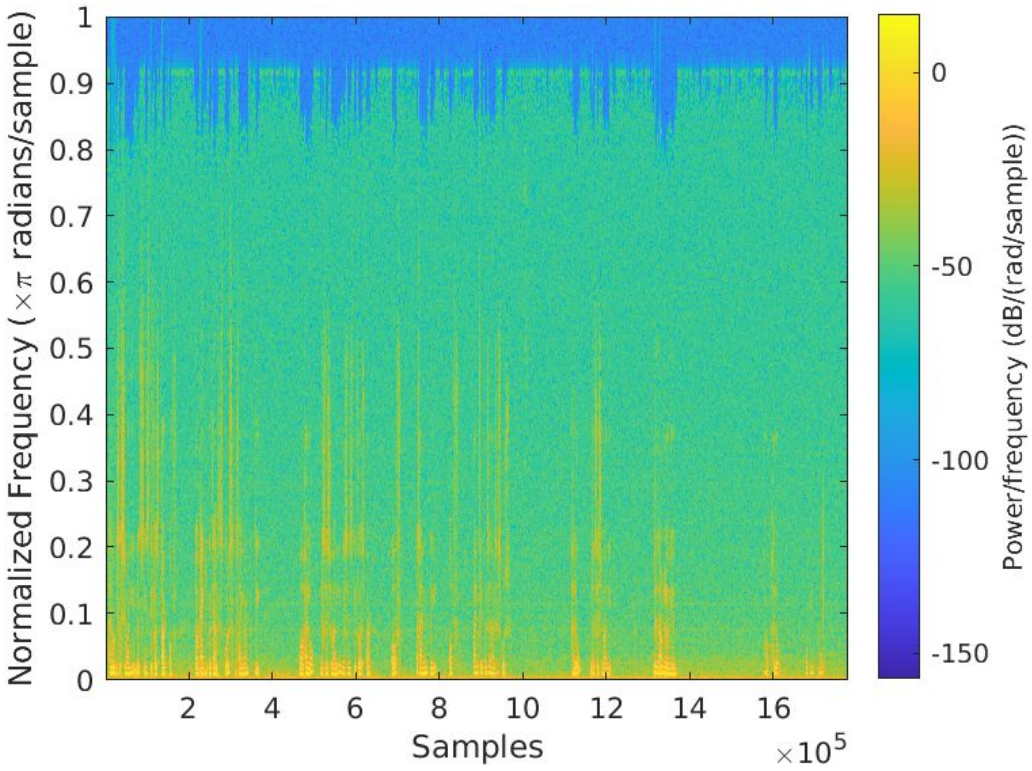
Case 2 -hanning



Case 3 -rectangular



Case 4 -triangular



Driver code for Q1 and Q2

```
% window_duratoin is the frame size, all in seconds

clc;close all;clear all;
window_duration1 = 100e-3; frame_shift1 = 10e-3;
tfa1(window_duration1, frame_shift1);

%%
clc;close all;clear all;
window_duration2 = 50e-3; frame_shift2 = 10e-3;
tfa1(window_duration2, frame_shift2);
%%
clc;close all;clear all;
window_duration3 = 20e-3; frame_shift3 = 10e-3;
tfa1(window_duration3, frame_shift3);

%%
clc;close all;clear all;
window_duration4 = 10e-3; frame_shift4 = 5e-3;
tfa1(window_duration4, frame_shift4);

%%
clc;close all;clear all;
tfa2(1); % hamming window
%%
clc;close all;clear all;
tfa2(2); % hanning window
%%
clc;close all;clear all;
tfa2(3); % rectandular window
%%
clc;close all;clear all;
tfa2(4); % triangular window
```

Observations:

In the second question,
Varying Windows tunes energy of the signal. They effect the time freq resolution.
A rectangular window allows more frequency components - hence making the signal more spread in the domain. The spectrograms varied a little bit, sharply cutting windows like rect and triag are not very fine.

Different windows varied spectrogram only a little bit. In the length of the time frequency bins (that is in first question, as we changed the frame size and frame shift). But over all all of them

look similar - time freq resol changed a lot. Where as in q2, time freq resol changed less as frame shift ans size are constant, but windows changed in shape.
There window shape and size effect tf resolution.

Narrow wind - good time res, bad req res and vice versa