SIGNAL PROCESSING AND COMMUNICATION

MCTE 4338

Mini Project 2

Students Names:

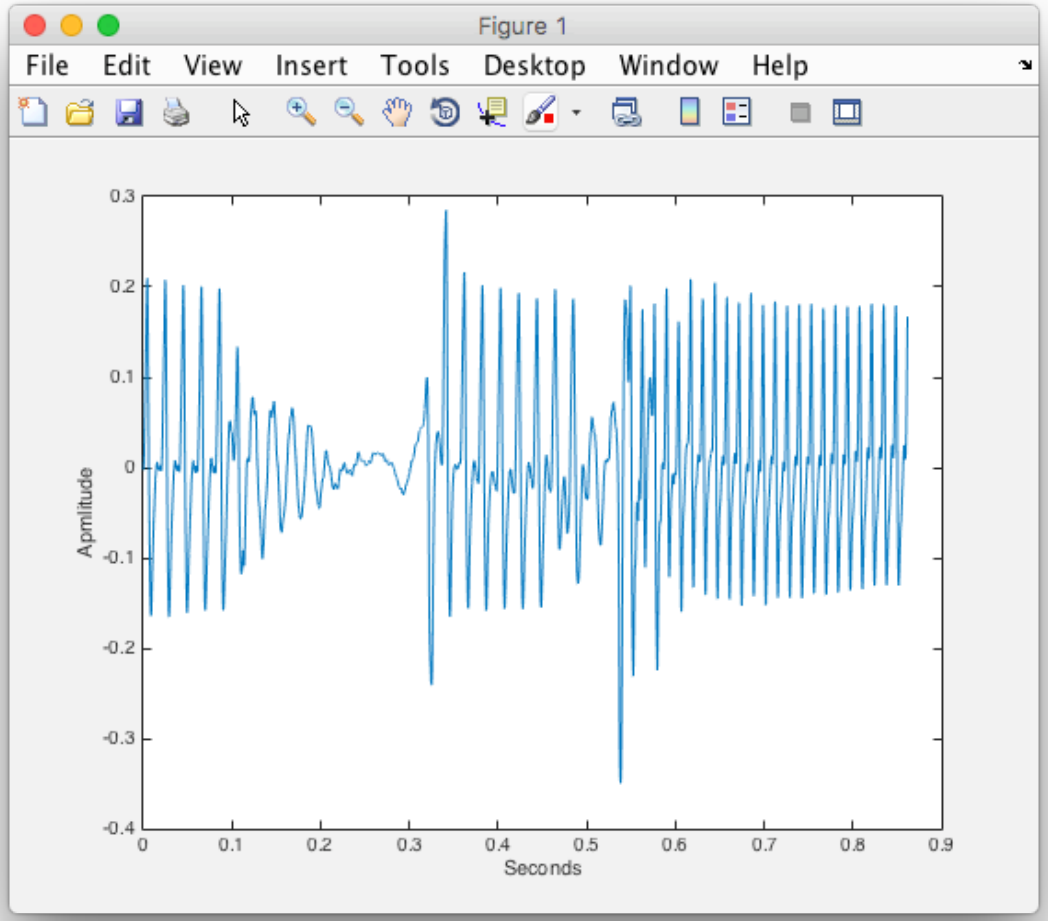
1. Saleem Hadad (1337479)
2. Ahed AlQaud (1410721)

> **Question 1**

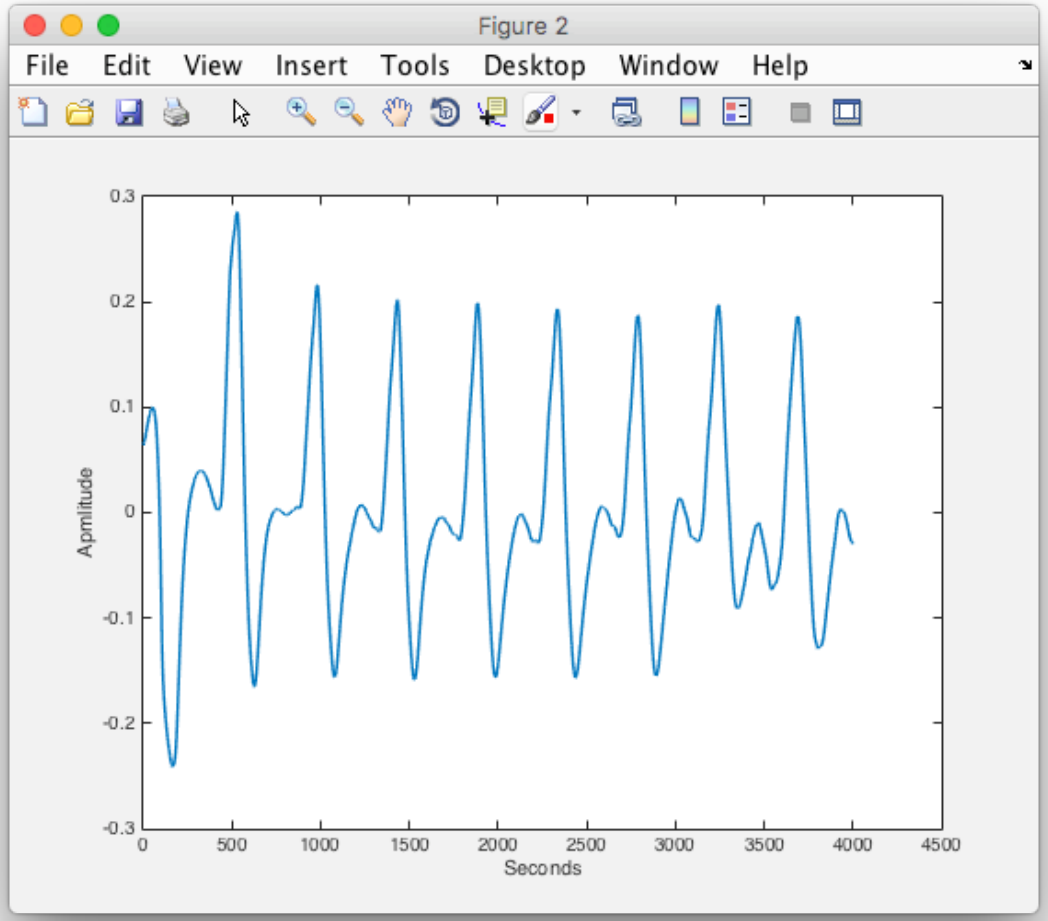
Load 'bass.wav' using [x,fs]=audioread('bass.wav'). Obtain and plot the signal for a segment of between x(670000:689000). You should be able to distinguish about 2 notes in the segments. Plot and determine the frequency of each notes (you should process the signal separately).

Solution

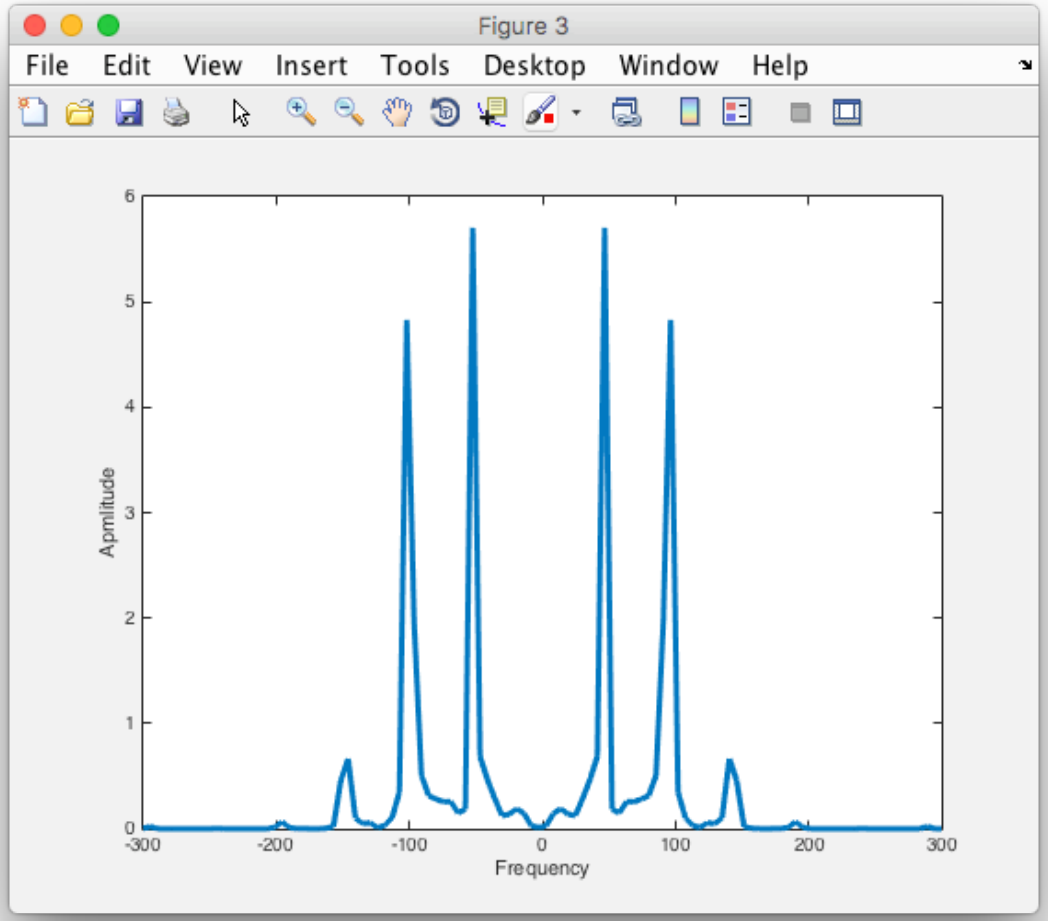
```
% base audio
[xn, fs]=audioread('audio.wav');
x1 = xn(670000:689000);
dt = 1/fs;
t=0:dt:(length(x1)*dt)-dt;
plot(t, x1)
xlabel('Seconds');
ylabel('Apmlitude');
```



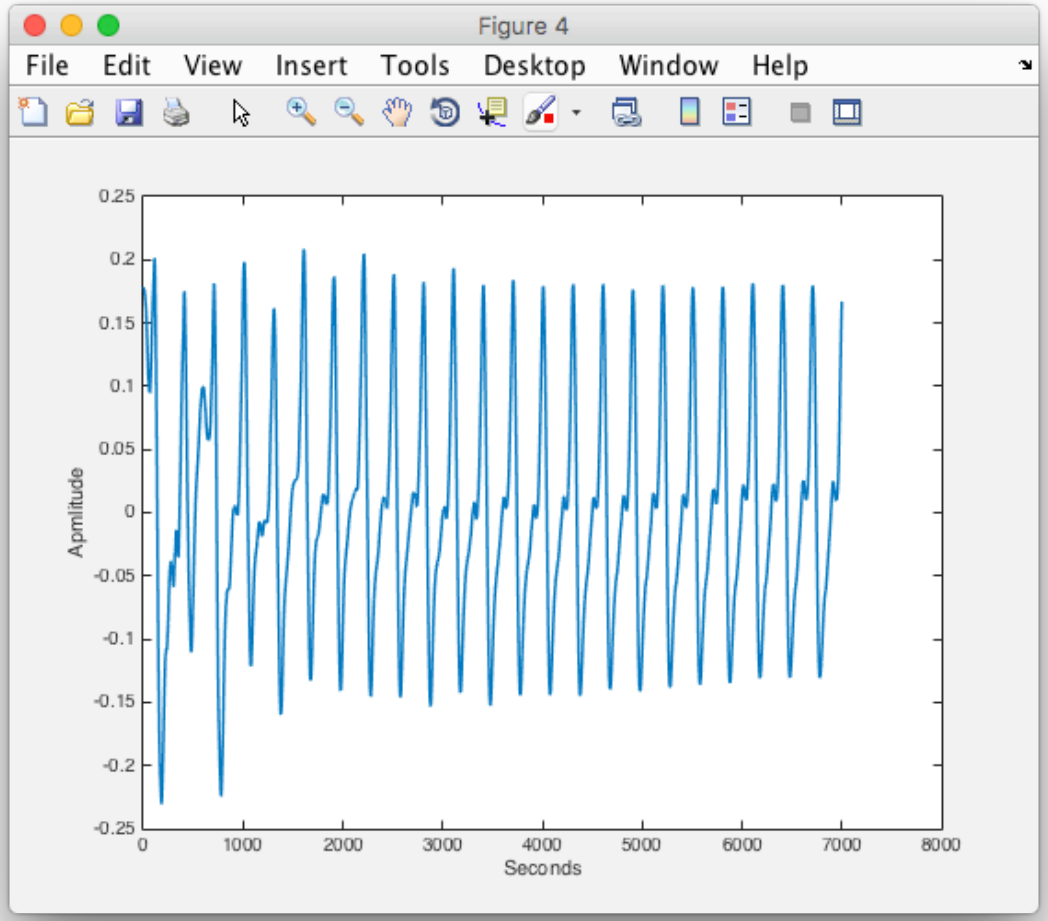

```
% note 1
figure;
N1 = xn(677000:681000);
t=1:1:length(N1);
plot(t, N1, 'lineWidth', 1.5);
xlabel('Seconds');
ylabel('Apmlitude');
```



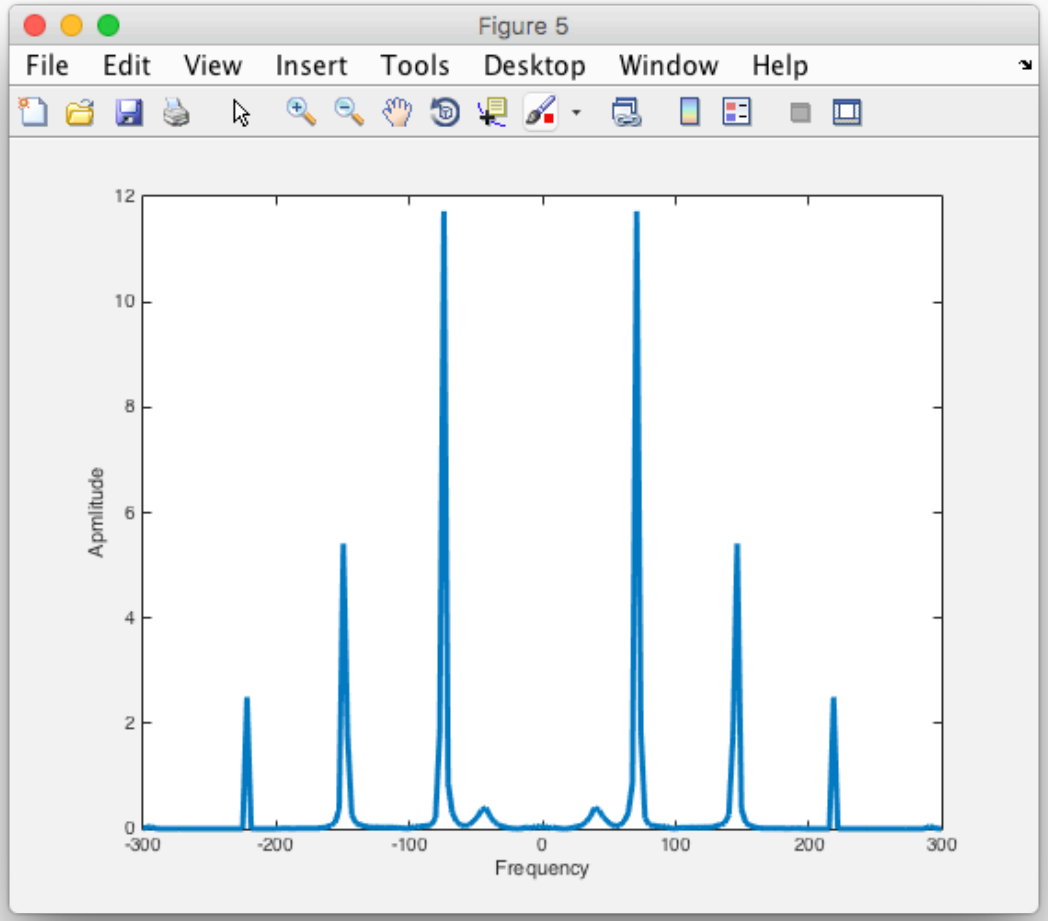

```
% note 1: spectrum analysis
figure;
N_1 = length(N1);
x_1 = fftshift(fft(N1));
fshift = (-N_1/2:N_1/2-1) * (fs/N_1);
powershift=abs(x_1).^2/N_1;
plot(fshift, powershift,'linewidth',3)
xlim([-300, 300]);
xlabel('Frequency');
ylabel('Apmlitude');
```




```
% note 2
figure;
N2 = xn(682000:689000);
t=1:1:length(N2);
plot(t, N2, 'lineWidth', 1.5);
xlabel('Seconds');
ylabel('Apmlitude');
```




```
% note 2: spectrum analysis
figure;
N_2 = length(N2);
x_2 = fftshift(fft(N2));
fshift = (-N_2/2:N_2/2-1) * (fs/N_2);
powershift=abs(x_2).^2/N_2;
plot(fshift, powershift,'linewidth',3)
xlim([-300, 300]);
xlabel('Frequency');
ylabel('Apmlitude');
```



> **Question 2**

Considering there are 10 workers in the office. You are required to design a biometric security system that will only allow the 10 workers to enter the office. Write a code that could recognize who is speaking using the method of Euclidean distance of their speech magnitude spectrum.

Dataset

We have collected 6 (the last column used for testing) audio files for 5 persons as folow:

Person	audio 1	audio 2	audio 3	audio 4	audio 5	audio 6
Saleem	✅	✅	✅	✅	✅	✅
Tariq	✅	✅	✅	✅	✅	✅
Khaled	✅	✅	✅	✅	✅	✅
Muhammed	✅	✅	✅	✅	✅	✅
Google	✅	✅	✅	✅	✅	✅

Python language used to do this project.

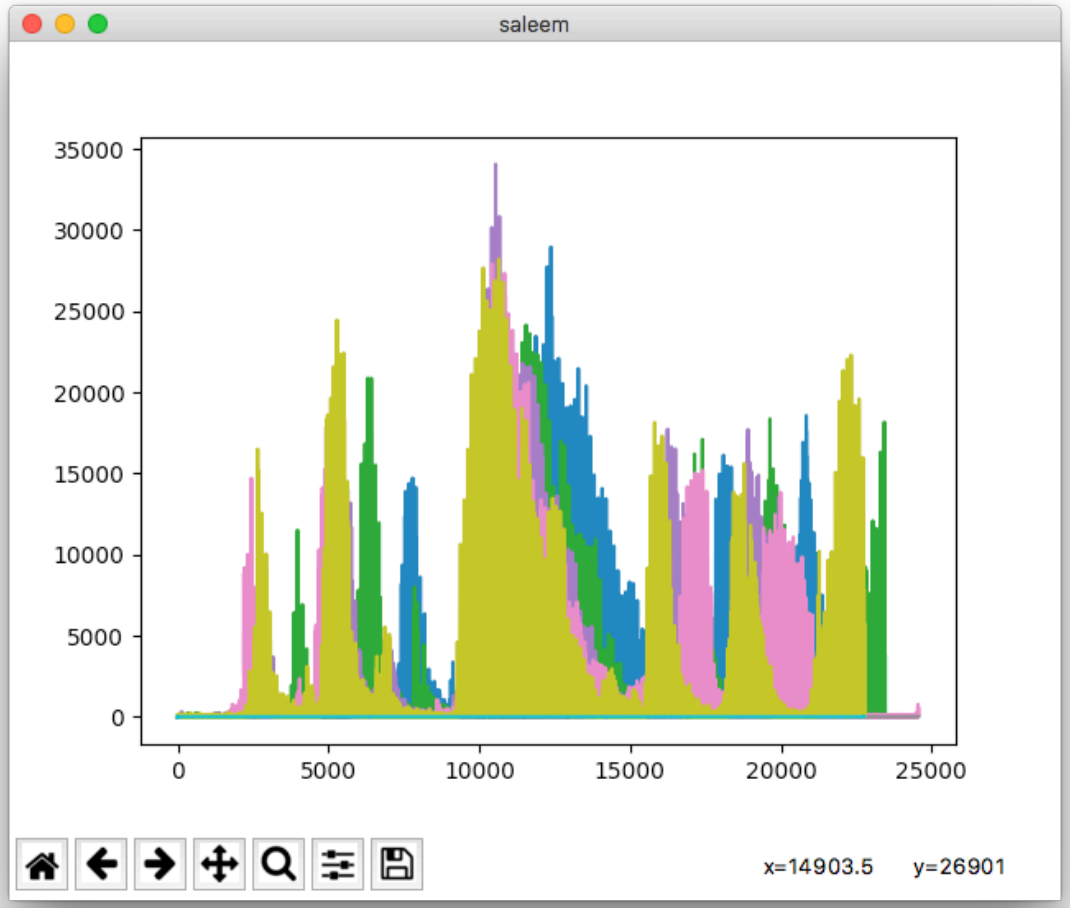
Code

```python
# import useful librarys and tools
import numpy as np
from scipy.fftpack import fft
import matplotlib.pyplot as plt
from scipy.io import wavfile as wav

NTTF = 1000

# stored names in the dataset.
names = [
    'saleem',
    'tariq',
    'khaled',
    'muhammed',
    'google'
]

# calculate the FFT for a given audio file name.
def calc_fft(audio):
    # read and process the audio file
    fs, data = wav.read(audio)
    fft_out = fft(data)
    k = np.arange(len(data))
    T = len(data)/fs
    frqLabel = k/T
    fft_out = fft_out[:len(fft_out)//2-1]
    frqLabel= frqLabel[:len(frqLabel)//2-1]

    # plot the spectrum
    plt.plot(frqLabel, np.abs(fft_out))
    return np.abs(fft_out)

# helper method used to calculate Euclidean distance between two variables.
def calc_dist(s1, s2):
    return np.linalg.norm(s1 - s2)

# used to compute the average FFT real values.
def calc_avg_fft(s):
    return np.sum(s/len(s))

# used to calcualte the average of all resulted fft for a given person name.
def avg_fft(name):
    avg_ffts = []
    plt.figure(name)
    for i in range(5):
        avg_ffts.append(
```

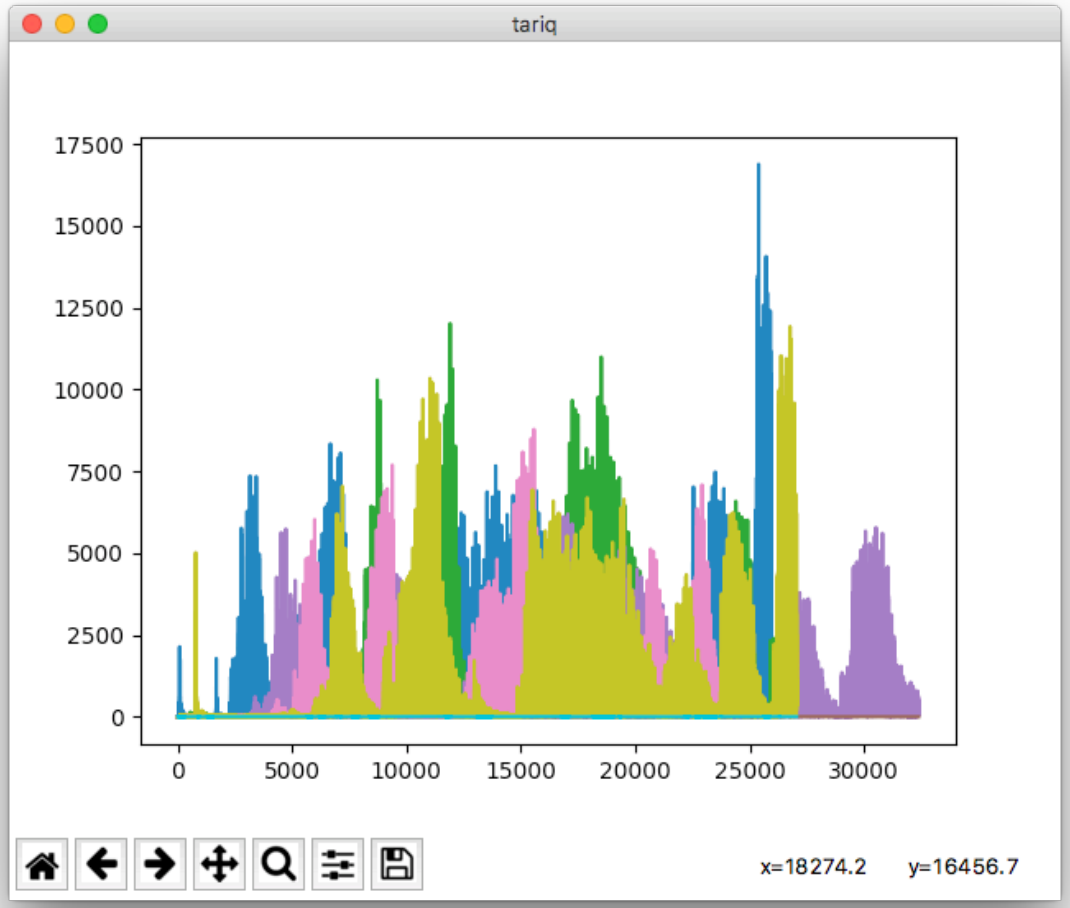
```
            calc_avg_fft(
                calc_fft('./dataset/' + name + str(i+1) + '.wav')
            )
        )
    return np.average(avg_ffts)

# normilize output result
def normalized(a, axis=-1, order=2):
    l2 = np.atleast_1d(np.linalg.norm(a, order, axis))
    l2[l2==0] = 1
    return a / np.expand_dims(l2, axis)

# calculate the percentage of match between each unknown audio file and known dataset.
def process_unknowns(known):
    n = len(known)
    result = []
    for i in range(n):
        dists = []
        avg_fft = calc_avg_fft(calc_fft('./dataset/unknown' + str(i+1) + '.wav'))
        for j in range(n):
            dists.append(calc_dist(known[j], avg_fft))
        result.append((1-normalized(dists))*100)
    return result

result = process_unknowns([
    avg_fft(names[0]),
    avg_fft(names[1]),
    avg_fft(names[2]),
    avg_fft(names[3]),
    avg_fft(names[4]),
])

# print results
for i in range(len(result)):
    print("unknown " + str(i+1) + ", matched "),
    for j in range(len(result[i][0])):
        print(str(int(result[i][0][j])) + "% with " + names[j] + " "),
    print

# show plotted graphs
plt.show()
```

Audio Analytics

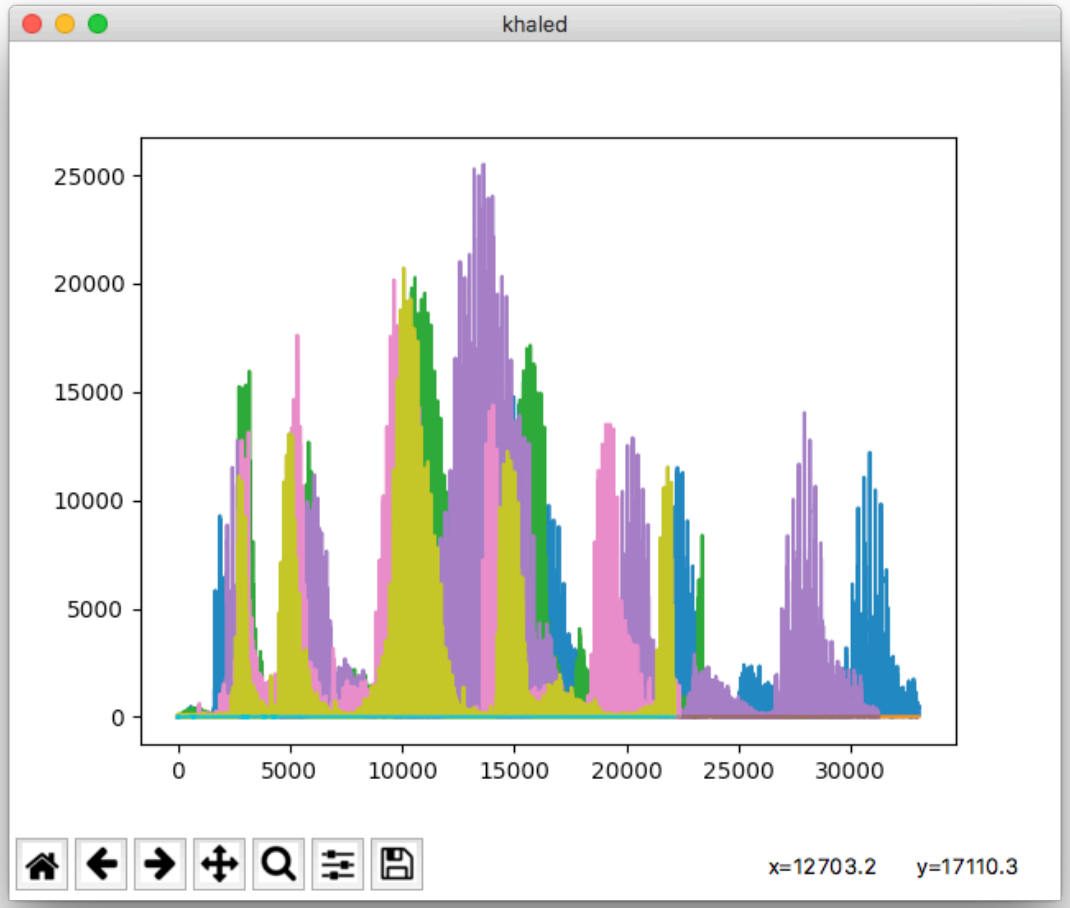
Saleem


saleem
35000
30000
25000
20000
15000
10000
5000
0
0
5000
10000
15000
20000
25000
x=14903.5
y=26901

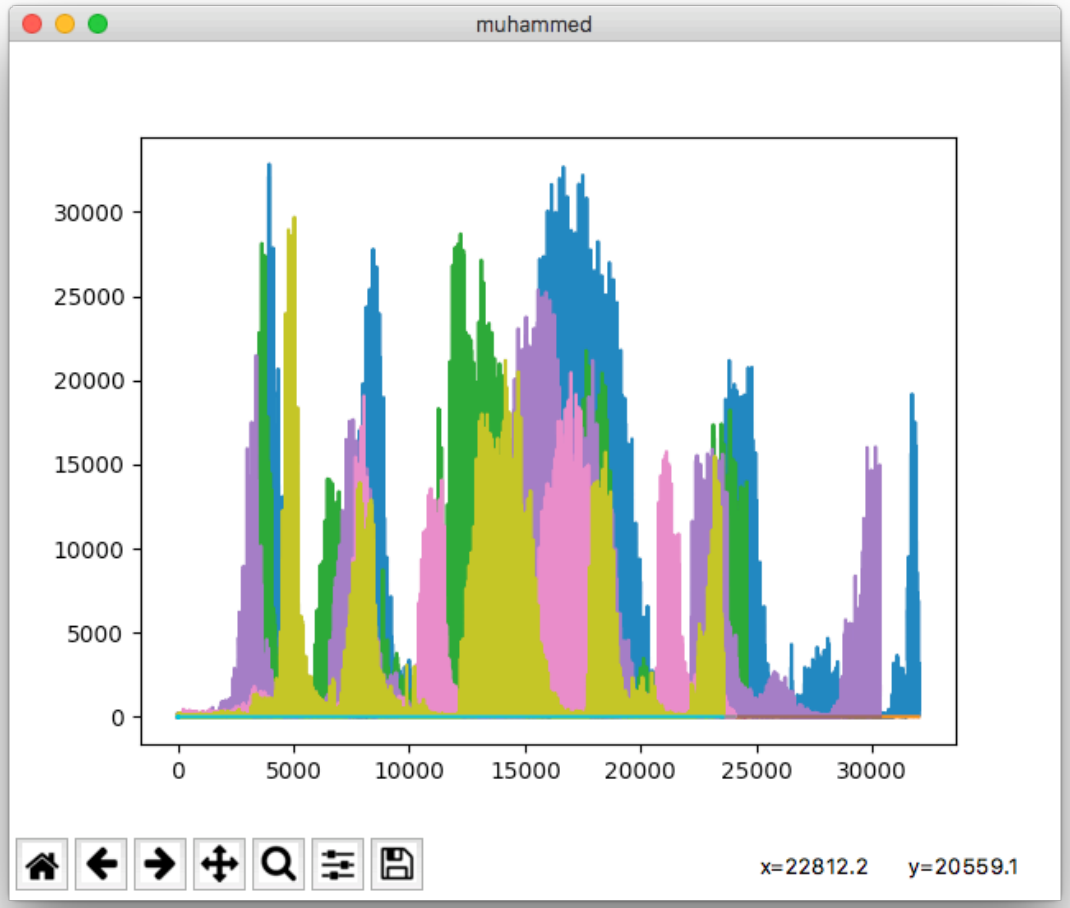
Tariq


tariq
17500
15000
12500
10000
7500
5000
2500
0
0
5000
10000
15000
20000
25000
30000
x=18274.2
y=16456.7

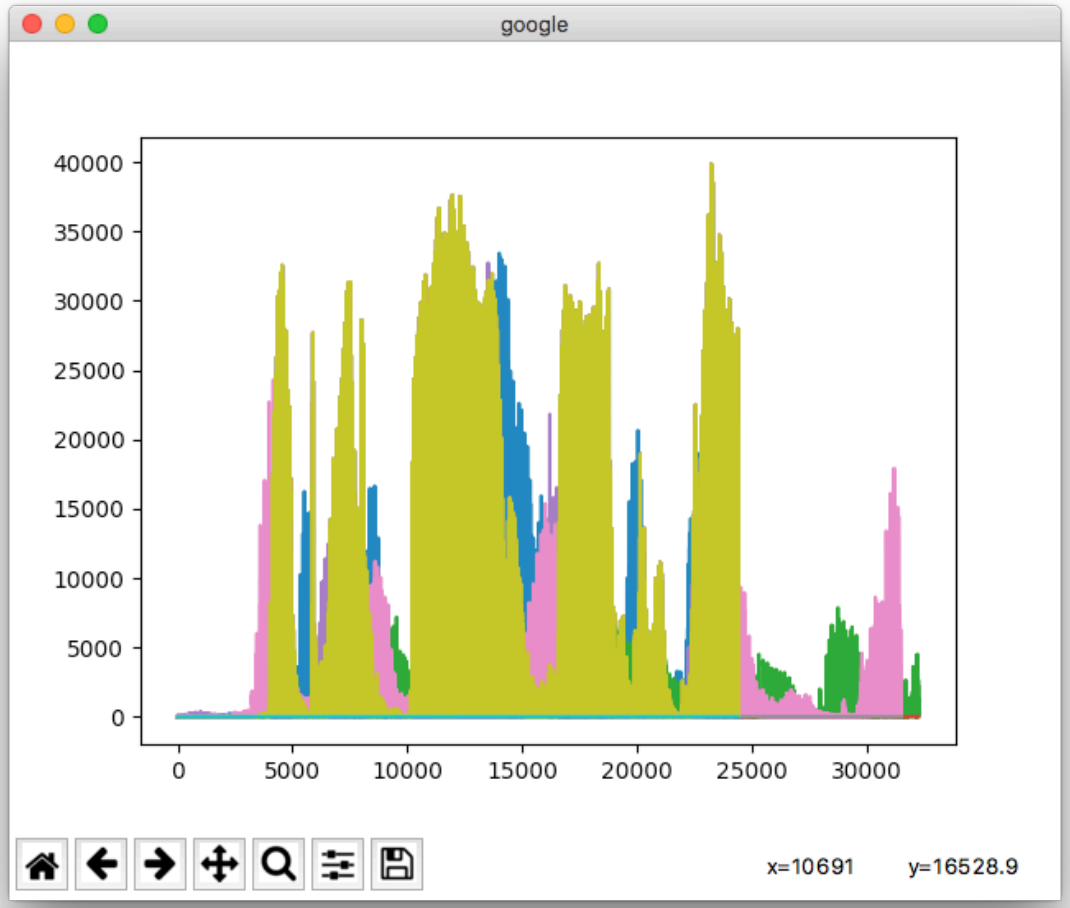
Khaled


khaled
25000
20000
15000
10000
5000
0
0
5000
10000
15000
20000
25000
30000
x=12703.2
y=17110.3

Muhammed


muhammed
30000
25000
20000
15000
10000
5000
0
0
5000
10000
15000
20000
25000
30000
x=22812.2
y=20559.1

Google



Results

Based on the result we obtain, the performance of the system perform very well in identifying the correct person. However, there are several limitations to this implementation such as the process time and the achieved accuracy, we can improve the speed performance by implementing it on DSP based system in real-time using C or C++ language which is way fast than Python or Matlab. In addition to that, we can increase the dataset size and implement machine learning algorithm with can beat the implemented handcraft feature extraction.

Unkown	Saleem	Tariq	Khaled	Muhammed	Google
1	**97%**	68%	81%	95%	7%
2	80%	**97%**	92%	74%	5%
3	89%	85%	**96%**	81%	3%
4	91%	81%	93%	**95%**	3%
5	54%	41%	47%	58%	**100%**