

Faculty of Engineering

Course Code: ECE466

Course Name: Digital Signal Processing

Simple Voice Biometric

(speech recognition)



Supervisor: Dr. Waleed El Nahal

Eng. Ahmad Hatem

Prepared by: Mahmoud Saied Mahmoud Dahroug

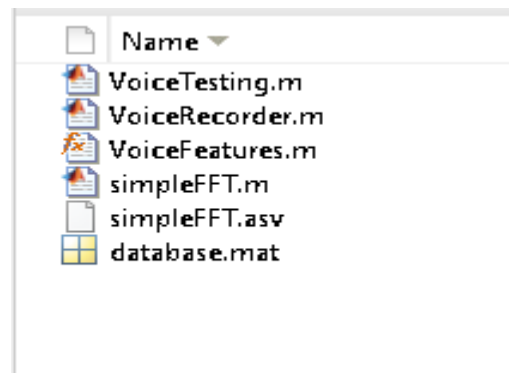
ID:154947

I-INTRODUCTION:

Modern-day security systems are wide-ranging and usually have multiple layers to get through before they can be properly cracked. Aside from the standard locks and deadbolts and alarm systems, there are very complex methods of protecting important material. Many of these are methods that can allow or disallow a specific individual to access material – a computer system has to be able to successfully detect a fingerprint, read an individual's eye patterns, or determine the true identity of a speaker. This last point is the focus of the project – speech recognition.

II-PROJECT DECRPTION:

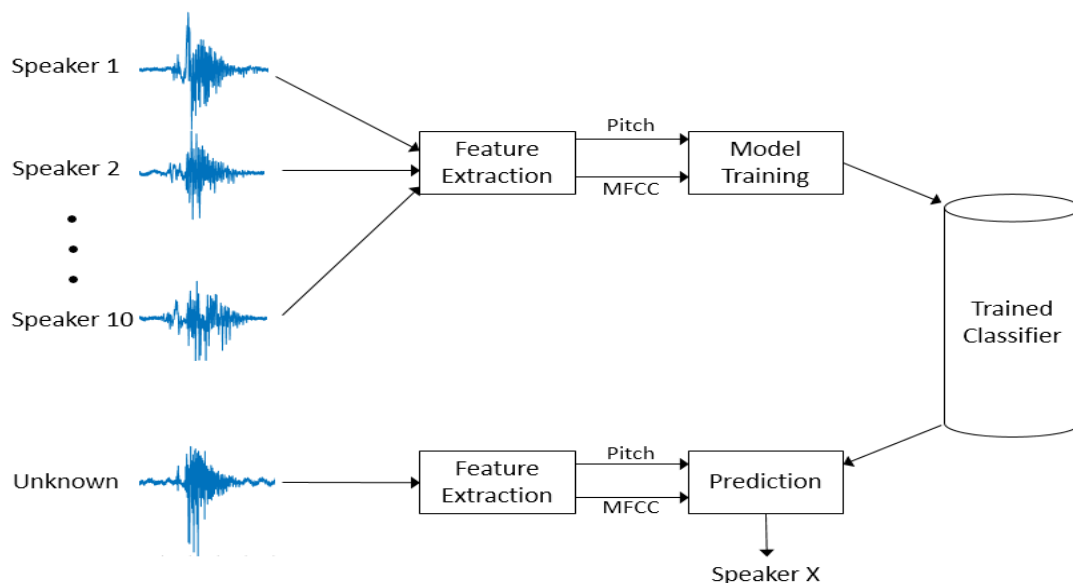
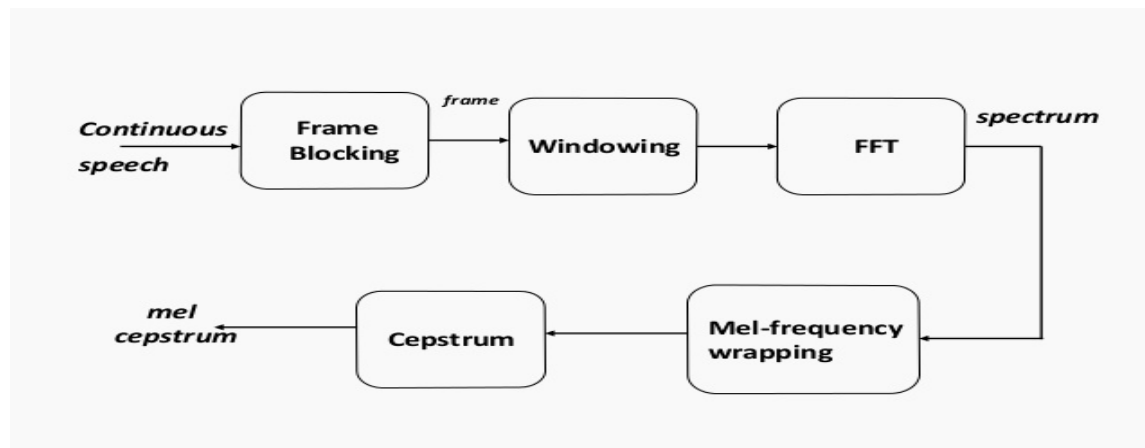
First of all, this project is divided into separate files just to illustrate the job of each file in the main code. The main code itself consists of two files linked to each other (which are “VoiceRecorder.m” & “VoiceTesting.m”) depending on each other as they include the same variables stored in the matlab memory and as they are contained in the same folder.



Each file has a unique task to accomplish. The file named “VoiceRecorder.m” is used to run a program coded to record voice signals or to create a record object for a certain time by using the audiorecorder () function which needs three input parameters which are in my code (n, b, chn).Where “n” will be the number of taken samples for my voice to be able to convert it from an analog signal to a digital one to do the processing techniques on it and it will be represented in ”b” bits and ”chn” is the number of channels used and it will store these signals into a database (that will be needed later) by using some of the matlab modules and assign for each voice a specific number that the user will choose by inserting it at the command window which will refer to its specific voice or speech. Then the file names “simpleFFT.m” includes the lines of code that is used to take the input analog signal of the voice and process on it a fast Fourier transform to change it from its time domain into the frequency domain for easier analysis on the signals in their spectrum form. Next it’s the “VoiceTesting.m” file section to once again record a random voice, but then it should compare it with the voices stored in the database and this

can be done only by comparing the voice's features of different signals to each other and this is what the "VoiceFeatures.m" file code lines are charged to, extracting the features of the voices using like the maximum peak for example to be able to compare it with other signals. Then the program should recognize the recorded speech after running the "VoiceTesting.m" file by matching the closest output maximum frequency to the input's one and produce an output number in the command window that refers to a specific user's voice exactly.

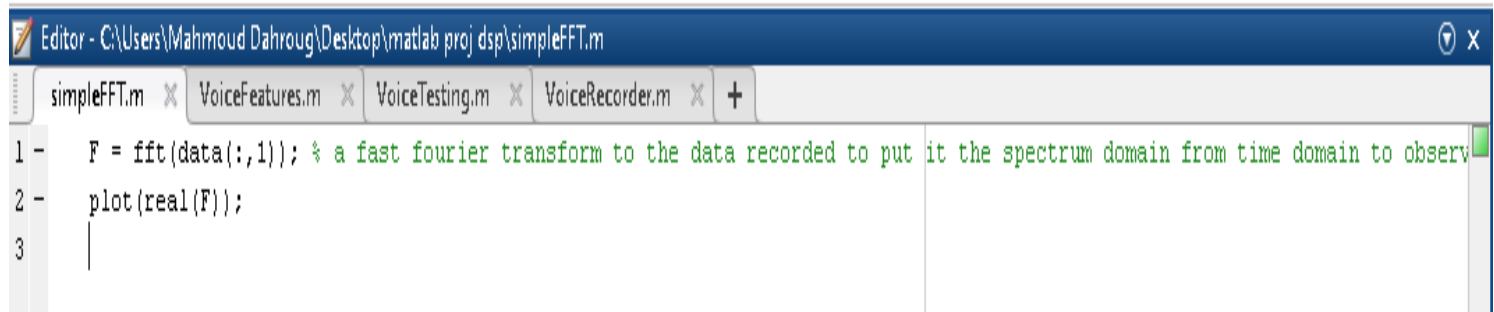
- **block diagrams:**



III-MATLAB CODE:

- **simpleFFT.m**

tlab proj dsp

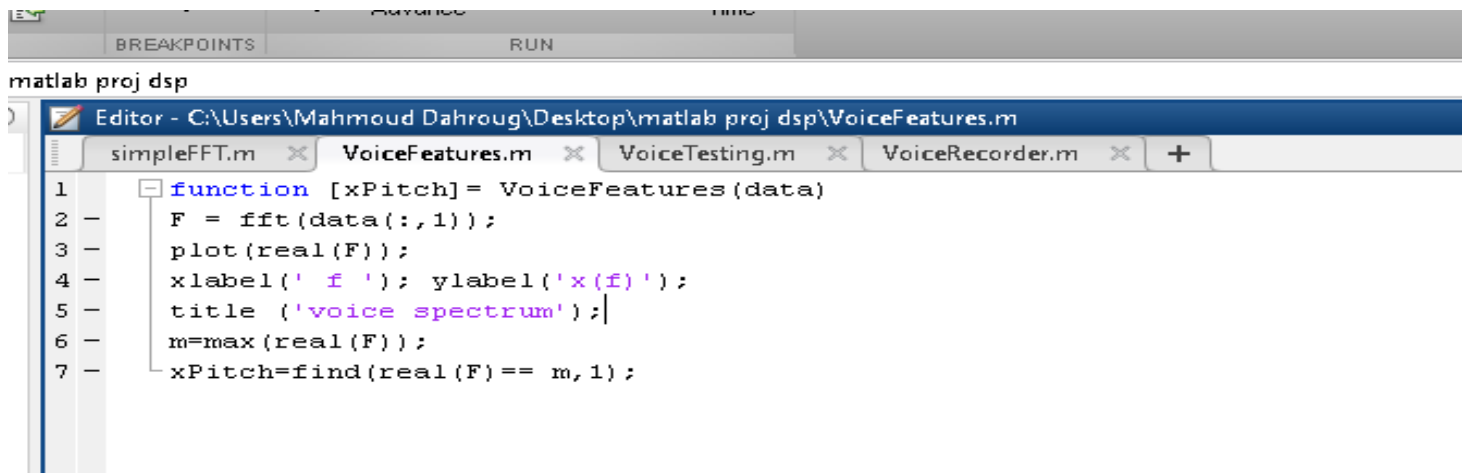


The screenshot shows the MATLAB Editor window with the file 'simpleFFT.m' open. The script contains the following code:

```
1 - F = fft(data(:,1)); % a fast fourier transform to the data recorded to put it the spectrum domain from time domain to observ
2 - plot(real(F));
3 -
```

```
F = fft(data(:,1)); % a fast fourier transform to the data
recorded to put it the spectrum domain from time domain to
observe frequencies
plot(real(F));
```

- **VoiceFeatures.m**



The screenshot shows the MATLAB Editor window with the file 'VoiceFeatures.m' open. The script contains the following code:

```
1 - function [xPitch]= VoiceFeatures(data)
2 -     F = fft(data(:,1));
3 -     plot(real(F));
4 -     xlabel(' f '); ylabel('x(f)');
5 -     title ('voice spectrum');
6 -     m=max(real(F));
7 -     xPitch=find(real(F)== m,1);
```

```
function [xPitch]= VoiceFeatures(data)
F = fft(data(:,1));
plot(real(F));
xlabel(' f '); ylabel('x(f)');
title ('voice spectrum');
m=max(real(F));
xPitch=find(real(F)== m,1);
```

- **VoiceRecorder.m**

oug ▸ Desktop ▸ matlab proj dsp

Editor - C:\Users\Mahmoud Dahroug\Desktop\matlab proj dsp\VoiceRecorder.m

```

1 - clear all
2 - close all
3 - clc
4 - %% creat a recorder object
5 - n=16000;
6 - b=8;
7 - chn=2;
8 - recorder = audiorecorder(n,b,chn); % n will be the number of taken samples for my voice and represented in b bits and chn i
9 - %%
10 - %record user's voice for 7 secs
11 - disp('please record your voice');
12 - drawnow();
13 - pause(1);
14 - recordblocking(recorder,7);
15 - play(recorder);
16 - data = getaudiodata(recorder);
17 - subplot(2,1,1);
18 - plot(data);
19 - xlabel(' t '); ylabel('x(t)');
20 - title ('Analog voice signal in time domain');
21 - subplot(2,1,2);
22 - stem(data);
23 - xlabel(' t '); ylabel('x(t)');
24 - title ('sampled voice signal');
25 - figure; % after running the program talk
26 - %%
27 - % Feature extraction
28 - f = VoiceFeatures(data);
29 - %%
30 - % save user's data
31 - uno = input('enter the user number :');
32 - try
33 -     load database
34 -     F=[F;f];
35 -     C=[C;uno];
36 -     save database F C
37 - catch
38 -     F=f;
39 -     C=uno;
40 -     save database F C
41 - end
42 - msgbox('your voice is recorded') % after running this talk something and memorize it cause you will need it later and ente

```

Workspace: ans, C, d, D, data, detx, f, F, i, ind, recc, sm

Command Window: pl, f, th, de, fx >>

```

clear all
close all
clc
%% creat a recorder object
n=16000;
b=8;
chn=2;
recorder = audiorecorder(n,b,chn); % n will be the number of taken
samples for my voice and represented in b bits and chn is the number
of channels used.
%%
%record user's voice for 7 secs
disp('please record your voice');
drawnow();
pause(1);
recordblocking(recorder,7);
play(recorder);
data = getaudiodata(recorder);
subplot(2,1,1);
plot(data);
xlabel(' t '); ylabel('x(t)');
title ('Analog voice signal in time domain');
subplot(2,1,2);
stem(data);
xlabel(' t '); ylabel('x(t)');
title ('sampled voice signal');
figure; % after running the program talk
%%
% Feature extraction
f = VoiceFeatures(data);
f
%%
% save user's data
uno = input('enter the user number :');
try
    load database
    F=[F;f];
    C=[C;uno];
    save database F C
catch
    F=f;
    C=uno;
    save database F C
end
msgbox('your voice is recorded') % after running this talk
something and memorize it cause you will need it later and enter a
user number

```

- **VoiceTesting.m**

```

Editor - C:\Users\Mahmoud Dahroug\Desktop\matlab proj dsp\VoiceTesting.m
simpleFFT.m  VoiceFeatures.m  VoiceTesting.m  VoiceRecorder.m  +

1 - clear all
2 - close all
3 - clc
4 - %% creat a recorder object
5 - recorder = audiorecorder(16000,8,2);
6 - %%
7 - %record user's voice for 7 secs
8 - disp('please record your voice');
9 - drawnow();
10 - pause(1);
11 - recordblocking(recorder,7);
12 - play(recorder);
13 - data = getaudiodata(recorder);
14 - subplot(2,1,1);
15 - plot(data);% after running the program talk
16 - xlabel(' t '); ylabel('x(t)');
17 - title ('Analog voice signal in time domain');
18 - subplot(2,1,2);
19 - stem(data);
20 - xlabel(' n '); ylabel('x(n)');
21 - title ('sampled voice signal');
22 - figure;
23 - %%
24 - % Feature extraction
25 - f = VoiceFeatures(data);
26 - f
27 - %%
28 - %find out the closest features from database
29 - load database
30 - D=[];
31 - for(i=1:size(F,1))
32 -     d= sum(abs(F(i)-f));
33 -     D=[D d];
34 - end
35 - %%
36 - %smallest distance
37 - sm = inf;
38 - ind=-1;
39 - for(i=1:length(D))
40 -     if (D(i)<sm)
41 -         sm=D(i);
42 -         ind=i;
43 -     end
44 - end
45 - detected_user =C(ind);
46 - disp('the detected class is :');
47 - detected user

```



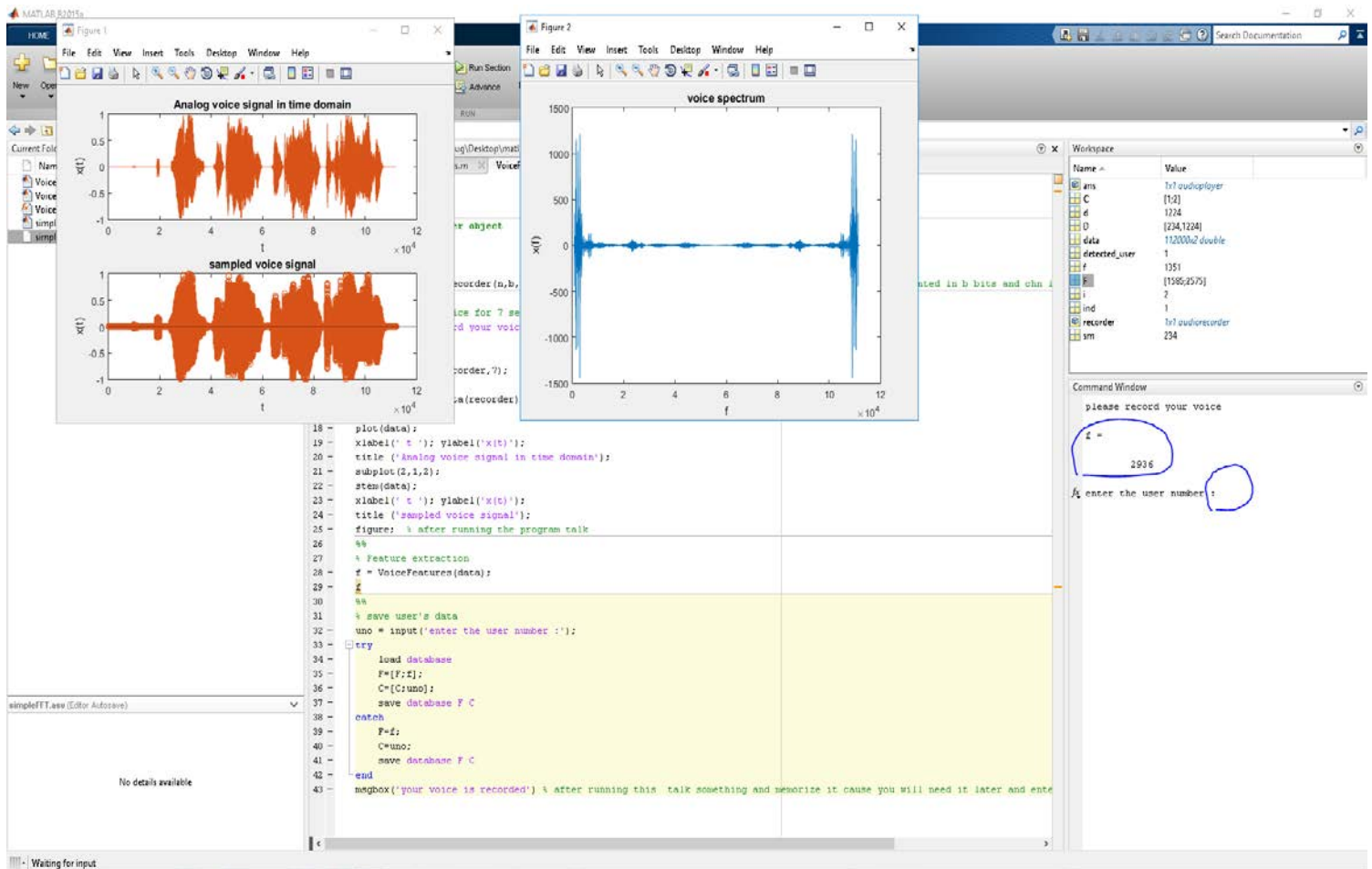
```

clear all
close all
clc
%% creat a recorder object
recorder = audiorecorder(16000,8,2);
%%
%record user's voice for 7 secs
disp('please record your voice');
drawnow();
pause(1);
recordblocking(recorder,7);
play(recorder);
data = getaudiodata(recorder);
subplot(2,1,1);
plot(data);% after running the program talk
xlabel(' t '); ylabel('x(t)');
title ('Analog voice signal in time domain');
subplot(2,1,2);
stem(data);
xlabel(' n '); ylabel('x(n)');
title ('sampled voice signal');
figure;
%%
% Feature extraction
f = VoiceFeatures(data);
f
%%
%find out the closest features from database
load database
D=[];
for(i=1:size(F,1))
    d= sum(abs(F(i)-f));
    D=[D d];
end
%%
%smallest distance
sm = inf;
ind=-1;
for(i=1:length(D))
    if (D(i)<sm)
        sm=D(i);
        ind=i;
    end
end
detected_user =C(ind);
disp('the detected class is :');
detected_user

```

IV-SIMULATION RESULTS:

- First user's i/p voice signal:



Command Window

please record your voice

f =

2936

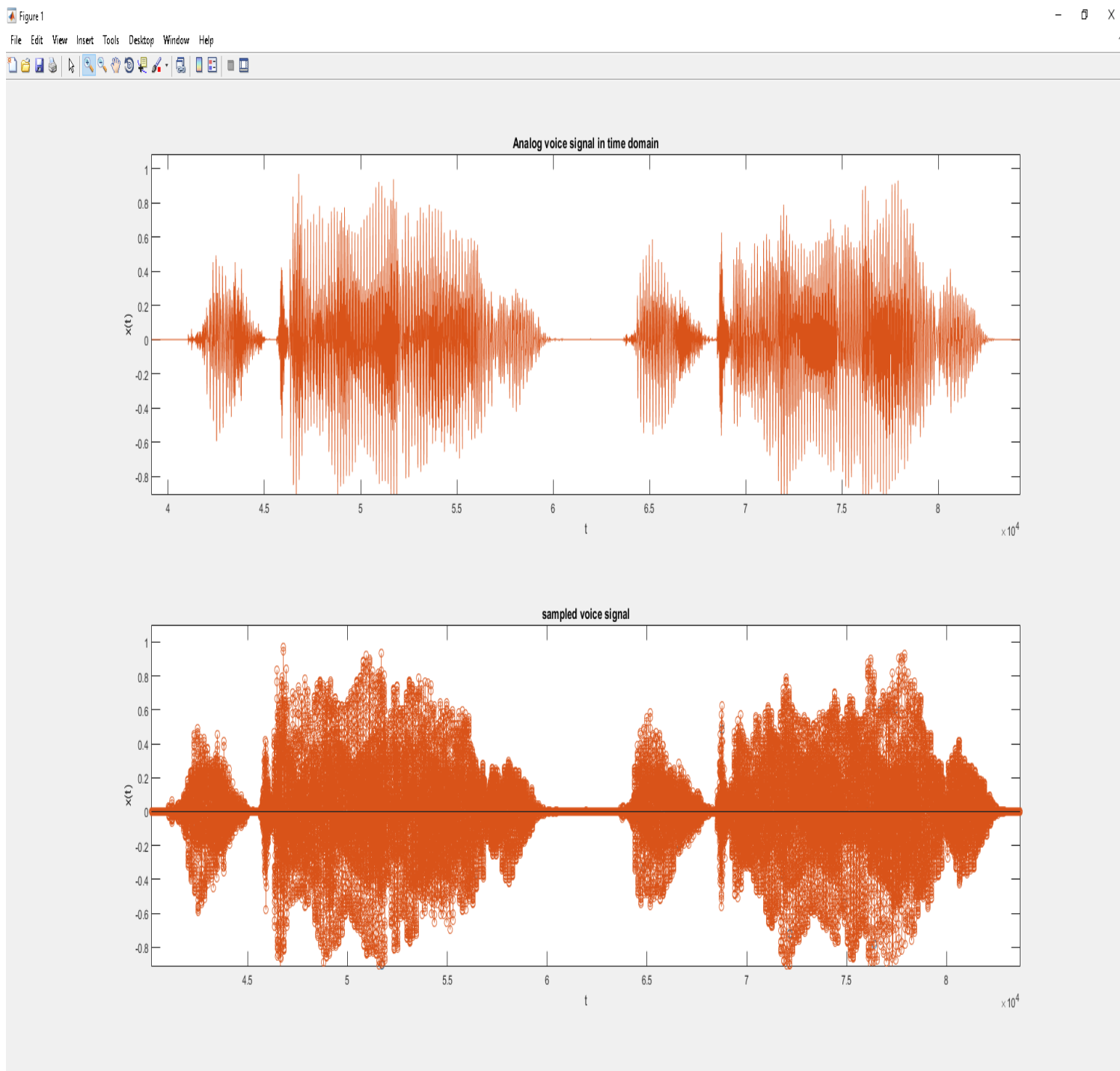
enter the user number :1

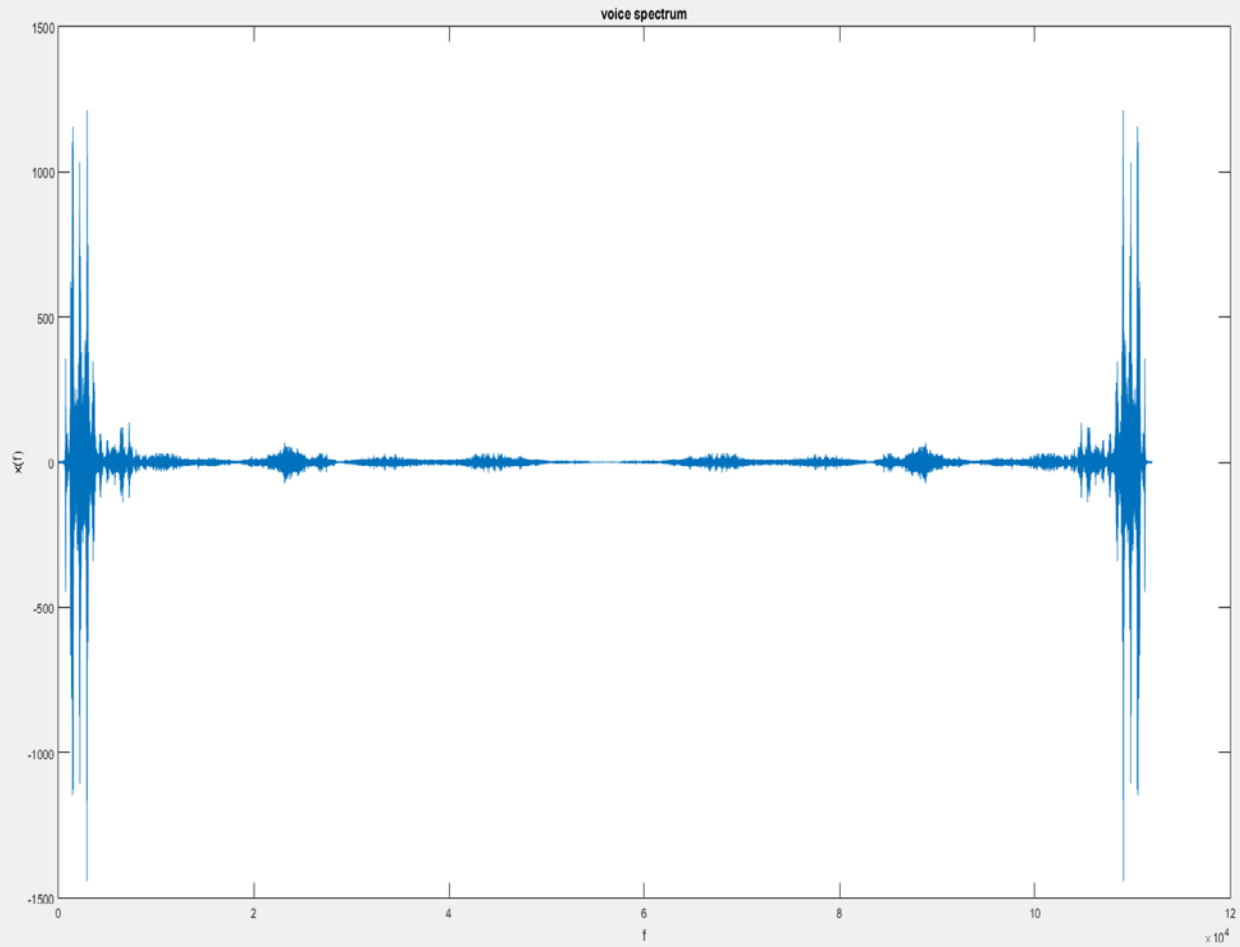
>> |

EDITOR - VOICEREORDER.M

1x1 double

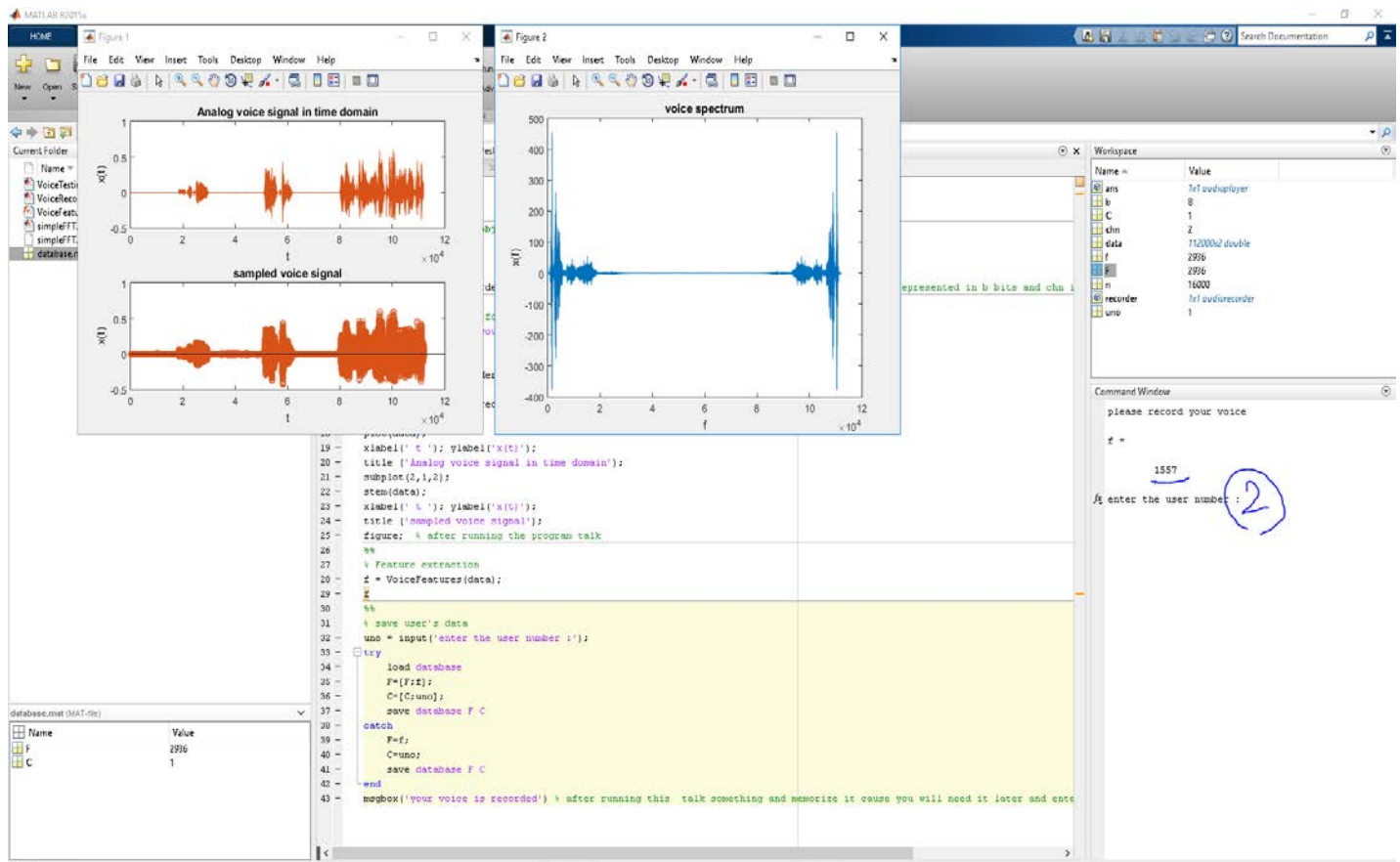
	1	2	3	4	5
1	2936				
2					
3					
4					
5					
6					
7					
8					





From the figures above it is shown that the first user's voice input signal has a peak frequency of 2936 Hz and the database storing it, and it shows also the reference number for it which is (1) and the other figures show the spectrum of the signal, the signal in time domain and the sampled signal.

- Second user's i/p voice signal:



Command Window

```

please record your voice

f =

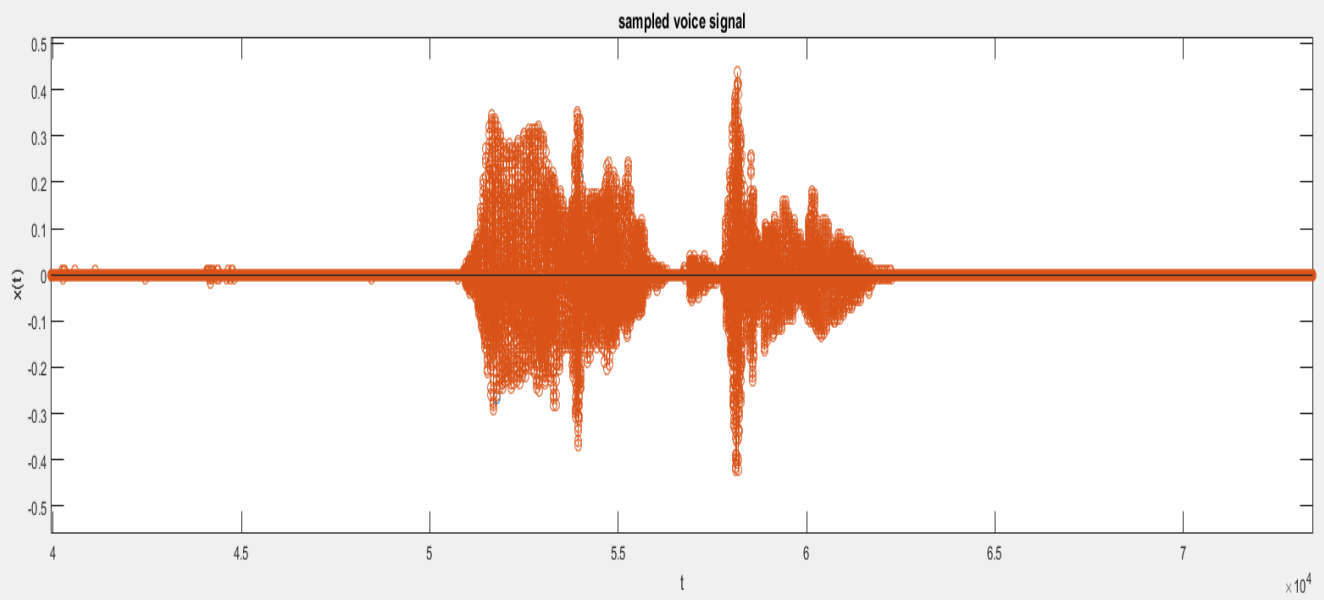
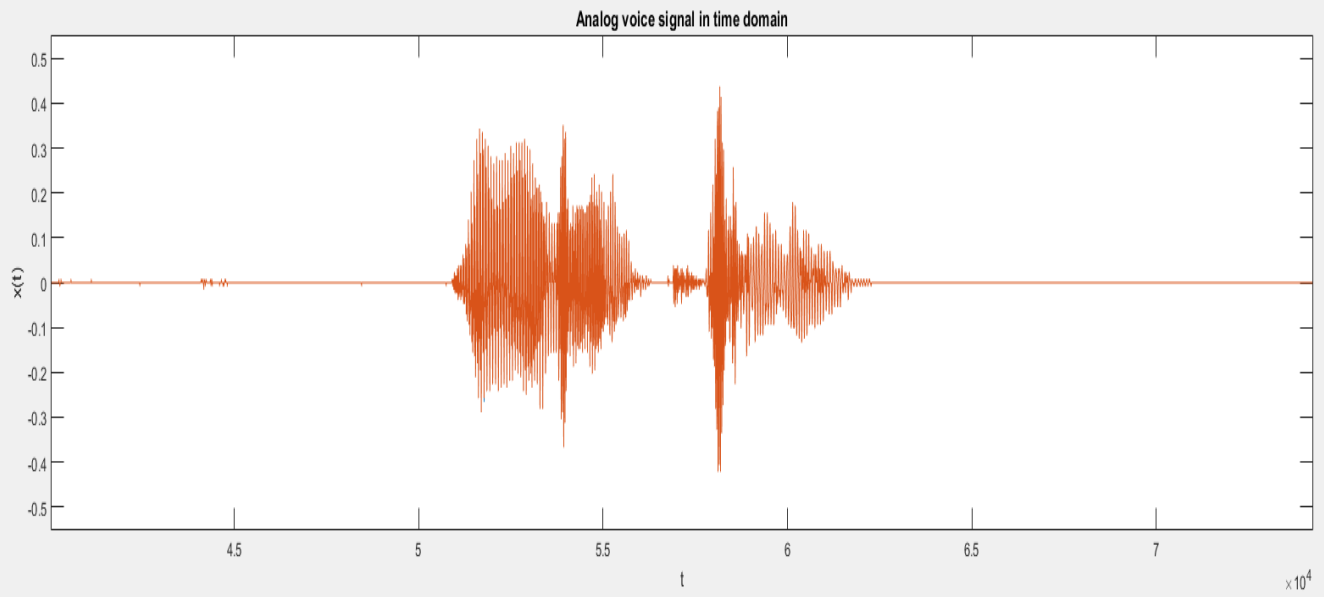
    1557

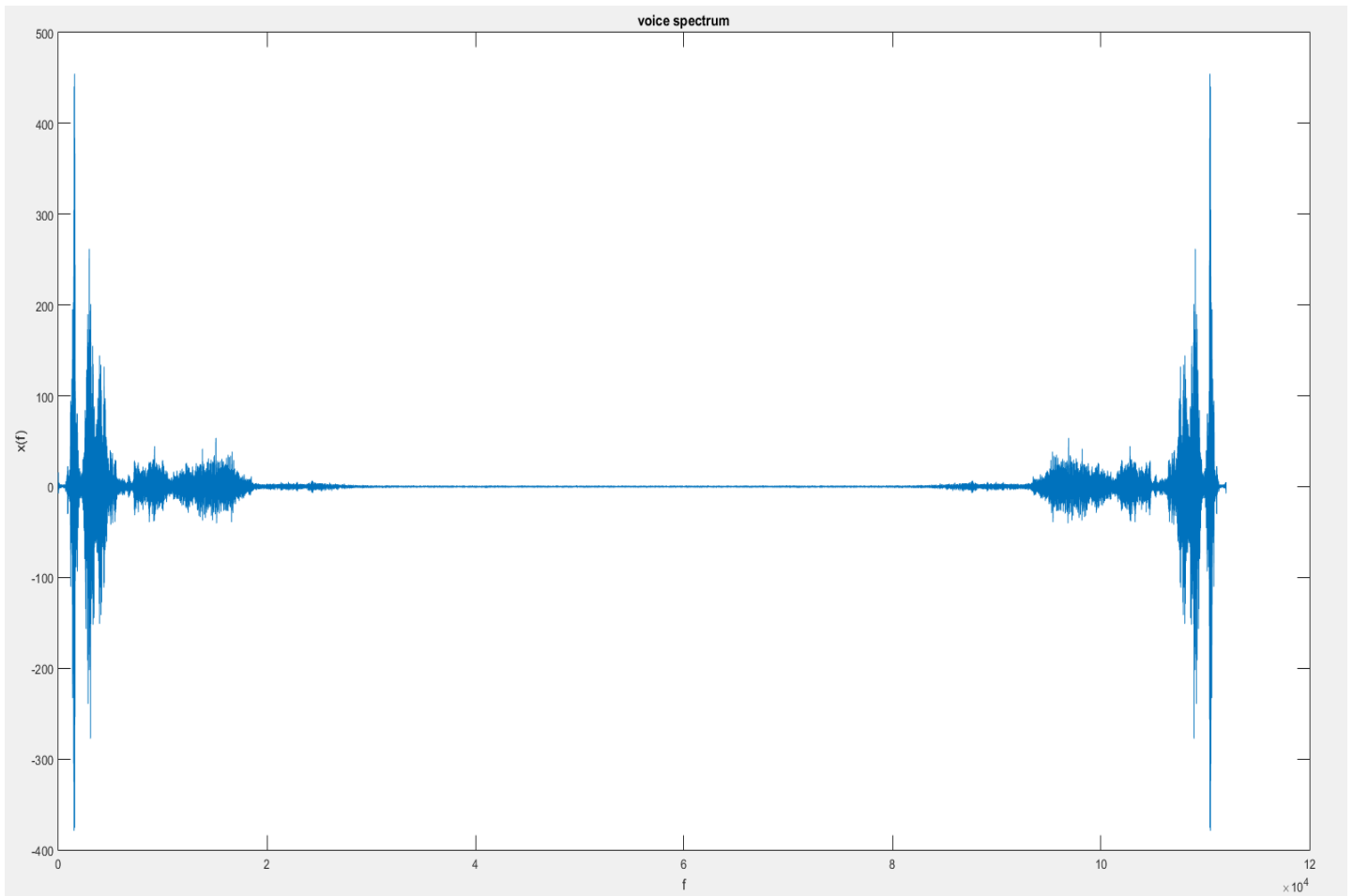
enter the user number :2
fx >> |

```

Editor - VoiceRecorder.m

	1	2
1	2936	
2	1557	
3		
4		
5		
6		
7		



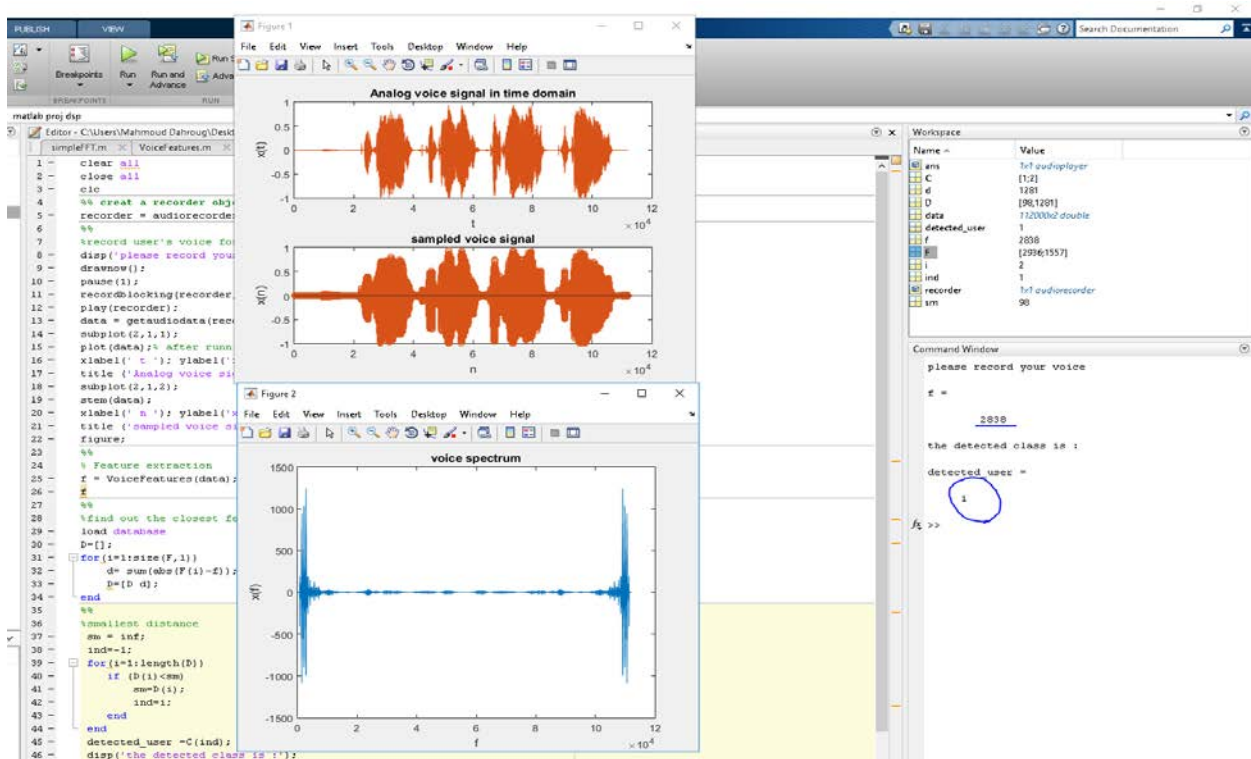


From the figures above it is shown that the second user's voice input signal has a peak frequency of 1557 Hz and the database storing it in addition to the stored one for the first user, and it shows also the reference number for it which is (2) and the other figures show the spectrum of the signal, the signal in time domain and the sampled signal.

➤ Output results:

Note: At testing, a user should repeat the same words he or she recorded at the first time with the same tone exactly so that the program works efficiently.

- First user's o/p voice signal:



Command Window

```

please record your voice

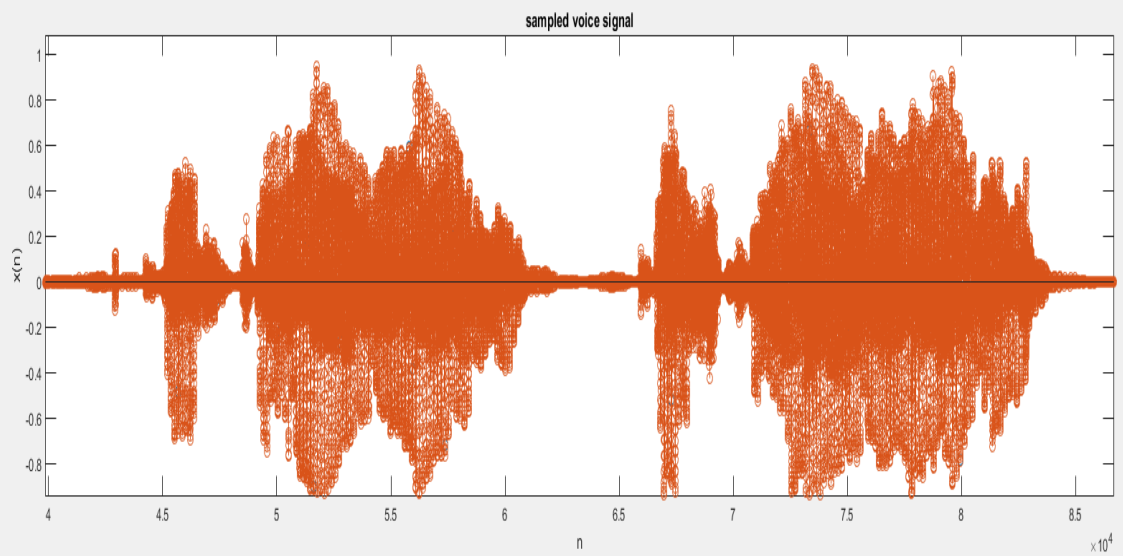
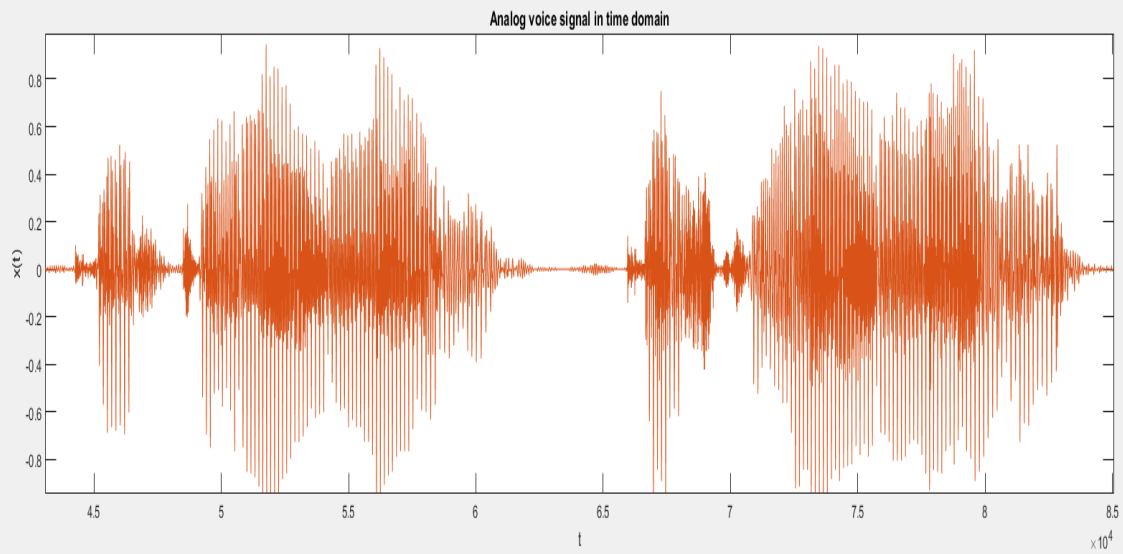
f =

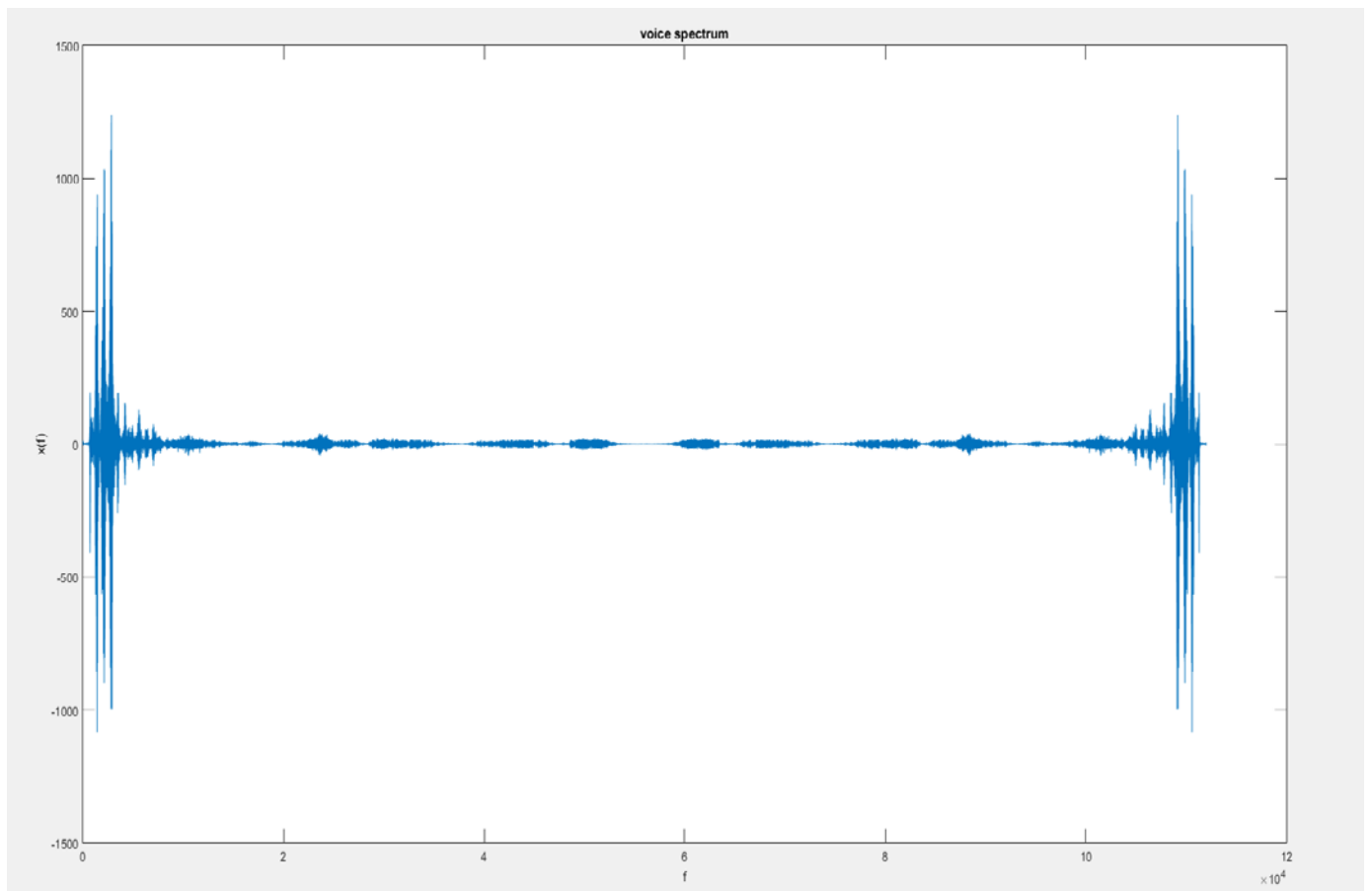
    2838

the detected class is :

detected_user =

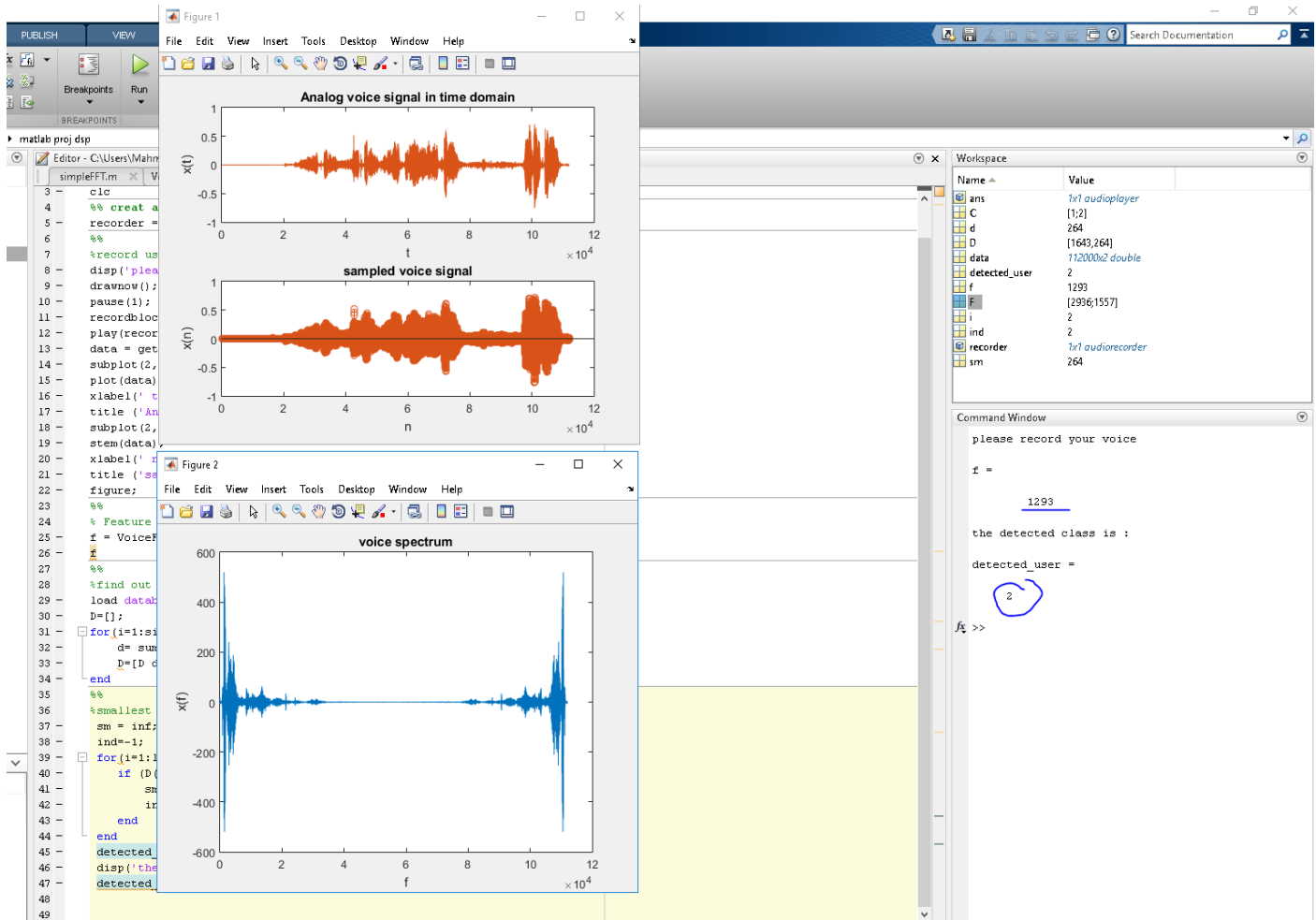
    1
  
```



From the figures above it is shown that a random user's voice output signal has a peak frequency of 2838 Hz which is very close to the first user's input voice signal, so the program was able to identify this random user as the first user by referring to its specific chosen number for it, which is number (1) and the other figures also shows the spectrum of the signal, the signal in time domain and the sampled signal.

- second user's o/p voice signal:



```

Command Window

please record your voice

f =

    1293

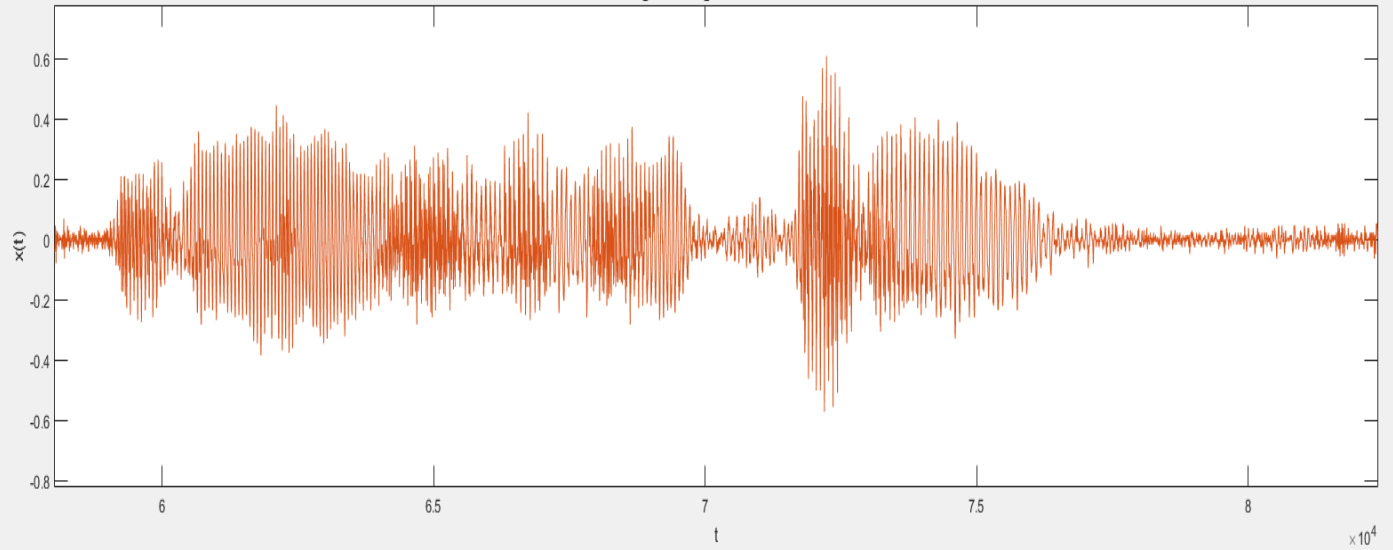
the detected class is :

detected_user =

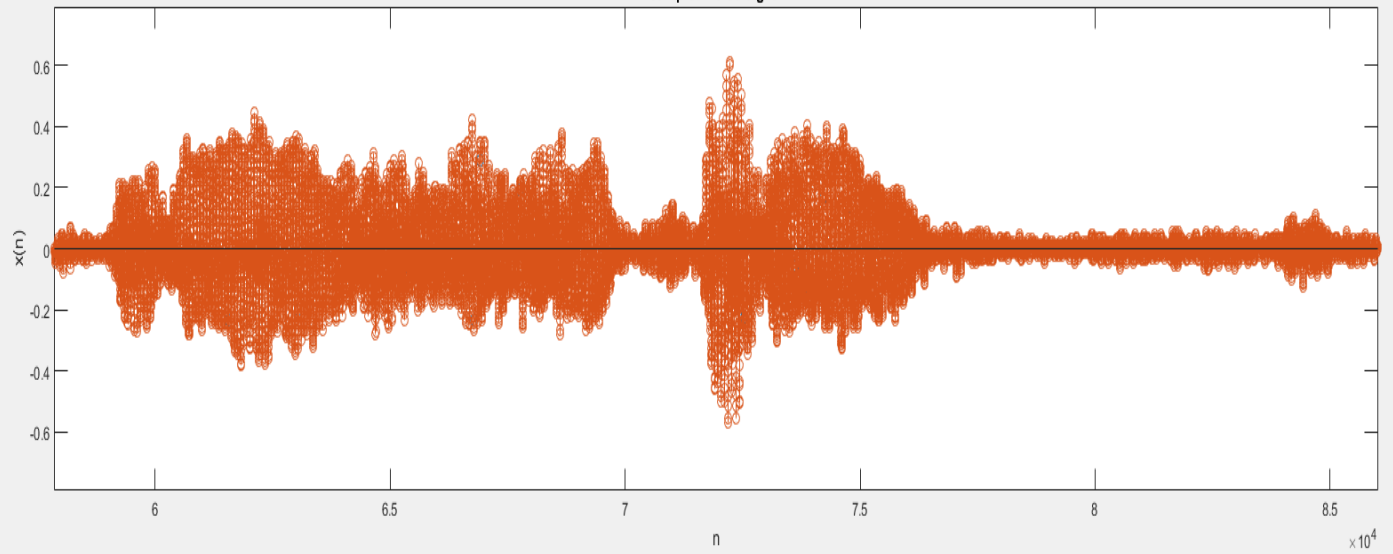
    2

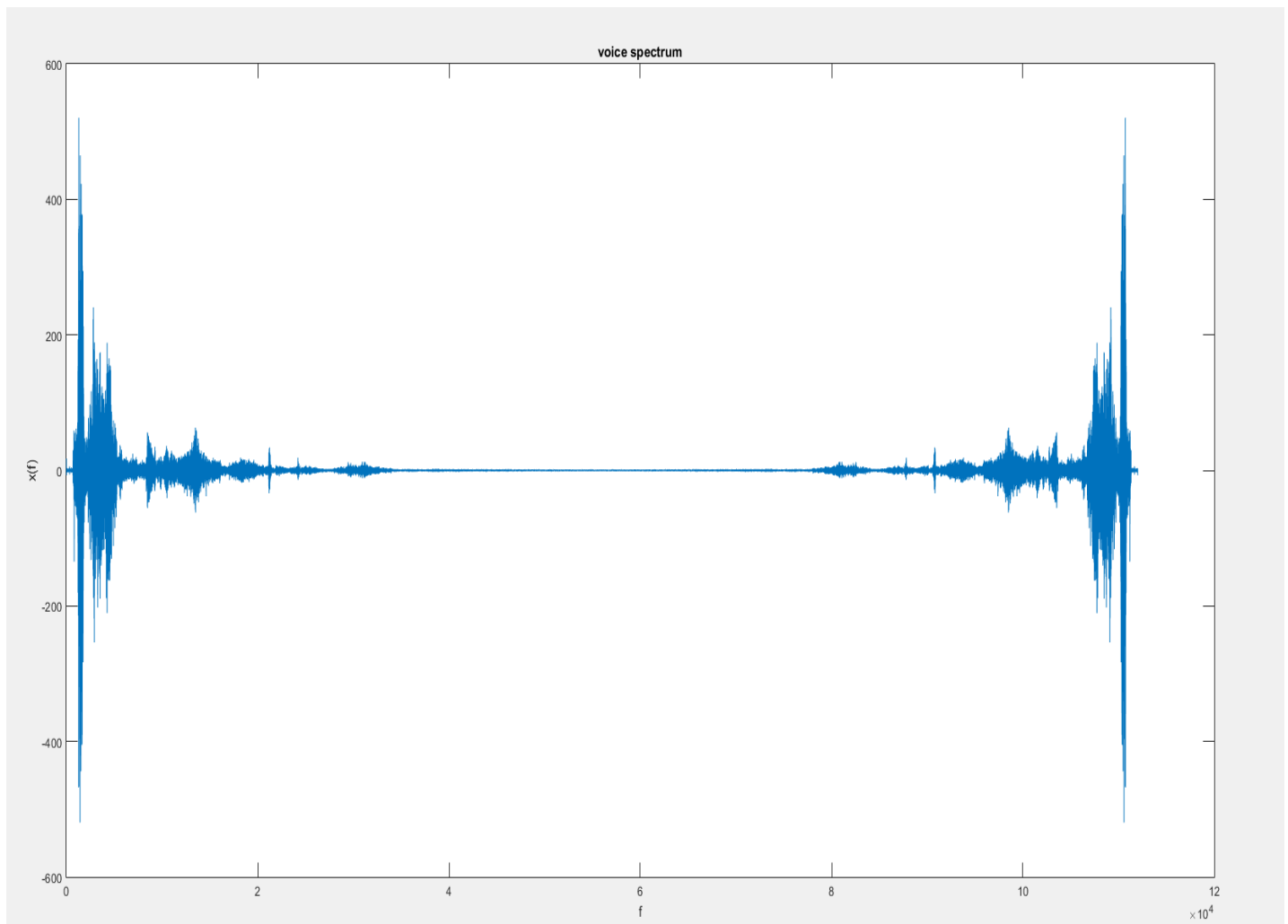
fx >> |
  
```

Analog voice signal in time domain



sampled voice signal





From the figures above it is shown that a random user's voice output signal has a peak frequency of 1293 Hz which make the best match to the second user's input voice signal, so the program was able to identify this random user as the second user by referring to its specific chosen number for it, which is number (2) and the other figures also shows the spectrum of the signal, the signal in time domain and the sampled signal.

V-CONCLUSION:

This project aims to determine the true identity of a specific speaker. The speaker will speak a word to the system, and the actual word itself can be any word. The system can accept any word because it is a text-independent system, meaning there is no specified word need. The system will determine the identity of a user by examining the vowel sounds, from the input speech signal. The vowel sounds will be analyzed in the frequency domain, specifically by looking at the peaks, or formants, of the frequency response of the signal. These formants will be compared with the formants of all of the group members previously stored in the database of the system. The group member with the highest resulting value after the comparison is the one identified as the speaker by the system.