Speech Processing
Lab-2 Report

Rajat Tyagi 180020029

Programming Language used : MATLAB

Task A :

Aim : To generate a stationary sine wave with frequency 10 Hz and sampling frequency of 1000 Hz.

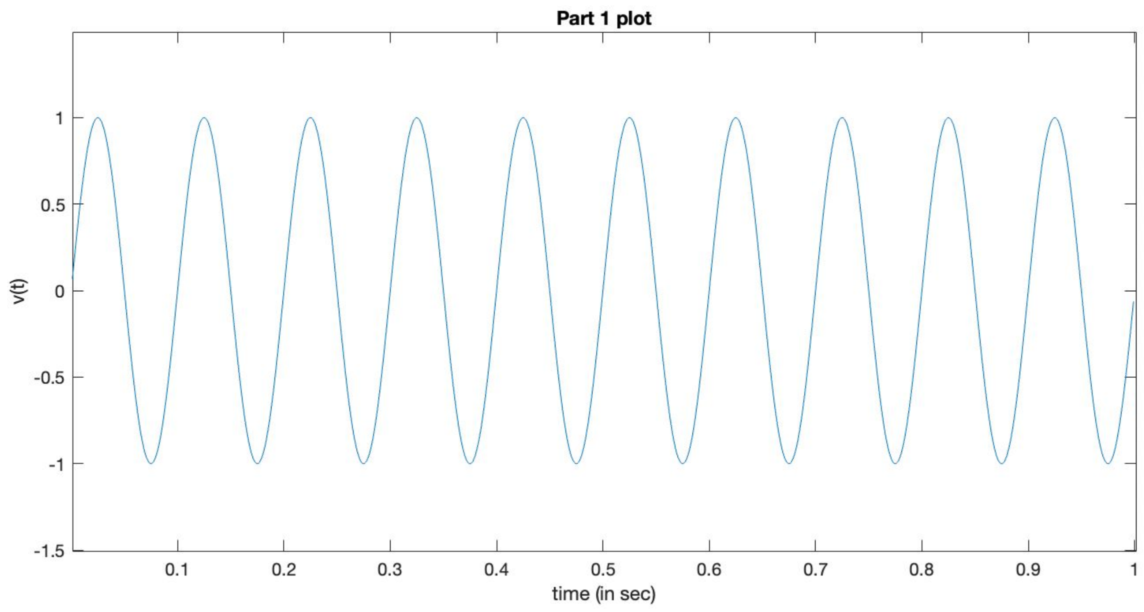
Procedure :

- First we define the required parameters which include timeline , sampling frequency, signal frequency etc.
- Then we use the sin() to create the sine wave.

Code :

```
%%----------------------------Part A ------------------------------

%Sampling Frequency in Hz
Fs = 1000;

%Time vector in seconds
t = 0 : 1/Fs : 1 - 1/Fs;

%Frequency of the sine wave in Hz
fc = 10;

%sine wave generation
sinewav1 = sin(2*pi*fc*t);

%plotting
plot(t,sinewav1);
title("Part 1 plot");
xlabel("time (in sec)");
ylabel("v(t)");
ylim([-1.5 1.5]);
```

Plots :



<u>Task B</u> :

Aim : To Generate a multitone signal consisting of three sine waves with frequency 10Hz, 50Hz and 100Hz.

Procedure :

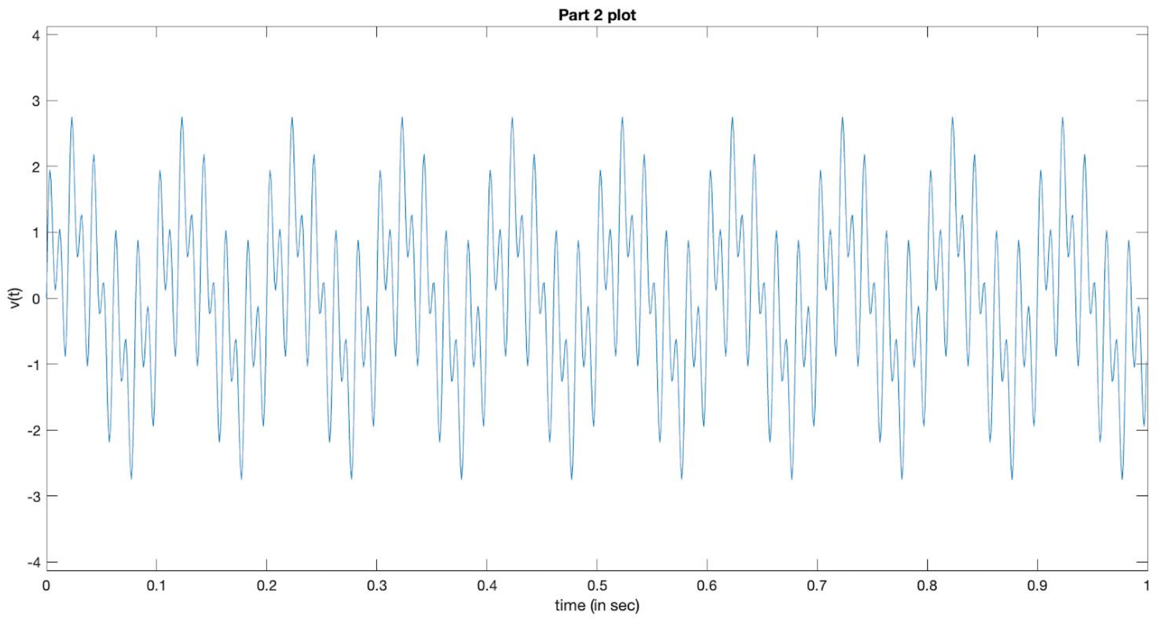
- We first define the timeline, sampling frequency, signal frequencies (fc1,fc2,fc3).
- Then we use the sin() function to generate three sine waves with frequencies fc1, fc2 and fc3.
- Finally we add the three vectors (Superposition) to generate a multitone signal.

Code :

```
%%-----------------------------Part B ------------------------------

%Time vector in seconds
t = 0 : 1/Fs : 1 - 1/Fs;

%Frequency of the sine waves in Hz
fc1 = 10;
```

```
fc2 = 50;
fc3 = 100;

%sine wave generation individual sine waves
sinewav1 = sin(2*pi*fc1*t);
sinewav2 = sin(2*pi*fc2*t);
sinewav3 = sin(2*pi*fc3*t);

%generating Multitone sine wave by superposition of the above three waves
multitone = sinewav1 + sinewav2 + sinewav3;

%plotting
plot(t,multitone);
title("Part 2 plot");
xlabel("time (in sec)");
ylabel("v(t)");
ylim([-1.5*max(multitone) 1.5*max(multitone)]);
```

Plots :



<u>Task C (part a)</u> :

Aim : To generate the following non-stationary multitone sine wave.

i. Sin(2*Π**10*t) 0 < t < 0.2 sec
ii. Sin(2*Π**10*t) + Sin(2*Π**50*t) 0.2 < t < 0.5 sec
iii. Sin(2*Π**10*t) + Sin(2*Π**50*t) + Sin(2*Π**100*t) 0.5 < t < 1 sec

Procedure :

1. Generation of the following 3 stationary sine waves.
   a. Sin(2*Π**10*t)	0 < t < 1 sec
   b. Sin(2*Π**50*t)	0.2 < t < 1 sec	and	0 otherwise
   c. Sin(2*Π**100*t)	0.5 < t < 1 sec	and	0 otherwise

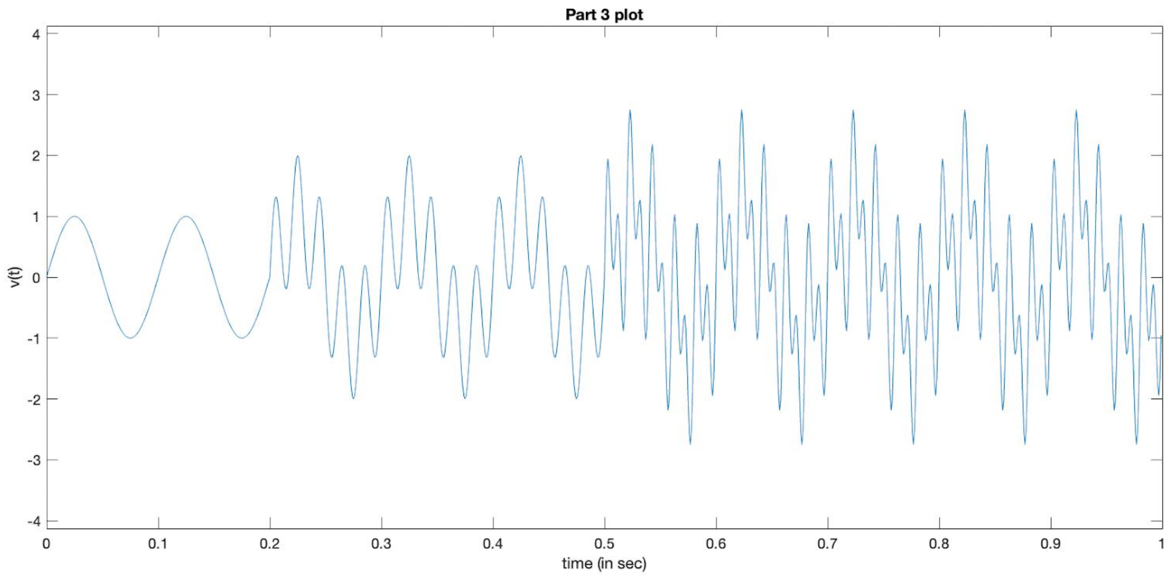
   <u>Note</u> : to generate the sine waves ‘b’ and ‘c’ we first generate a complete sine wave from t = 0 to 1 similar to the above parts of this lab. After this we set the remaining samples to zero.

2. Adding these sine waves (superposition) to make one multitone non stationary wave.

Code :

```
%%----------------------------Part C ------------------------------
% part a)

%Time vector in seconds
t = 0 : 1/Fs : 1 - 1/Fs;

%Frequency of the sine waves in Hz
fc1 = 10;
fc2 = 50;
fc3 = 100;

%sine wave generation individual sine waves

% (1) Sin(2Π*10t)
sinewav1 = sin(2*pi*fc1*t);

% (2) Sin(2Π*50*t)   0.2 sec < t < 1 sec and 0 for 0 < t < 0.2 sec
sinewav2 = sin(2*pi*fc2*t);
sinewav2(1:0.2*Fs) = 0;

% (3) Sin(2Π*100*t)   0.5 sec < t < 1 sec and 0 for 0 < t < 0.5 sec
sinewav3 = sin(2*pi*fc3*t);
sinewav3(1:0.5*Fs) = 0;

%super-position of the three parts
nonStationary = sinewav1 + sinewav2 + sinewav3;

%plotting
plot(t,nonStationary);
title("Part 3 plot");
xlabel("time (in sec)");
ylabel("v(t)");
ylim([-1.5*max(nonStationary) 1.5*max(nonStationary)]);
```

Plots :



<u>Task C (part b)</u> :

Aim : To plot the frequency magnitude response for the full duration of the multitone non-stationary signal and the following three sub durations
(i) 0 to 0.2 sec, **(ii)** 0.2 sec to 0.5 sec and **(iii)** 0.5 sec to 1 sec

Procedure :

- Using the fft() and fftshift() inbuilt functions to find the fft of the signal generated in Task C part a.
- Generation of the frequency line by taking length of the generated frequency response from -N/2 to +N/2 and multiplying with the sampling frequency.

Code :

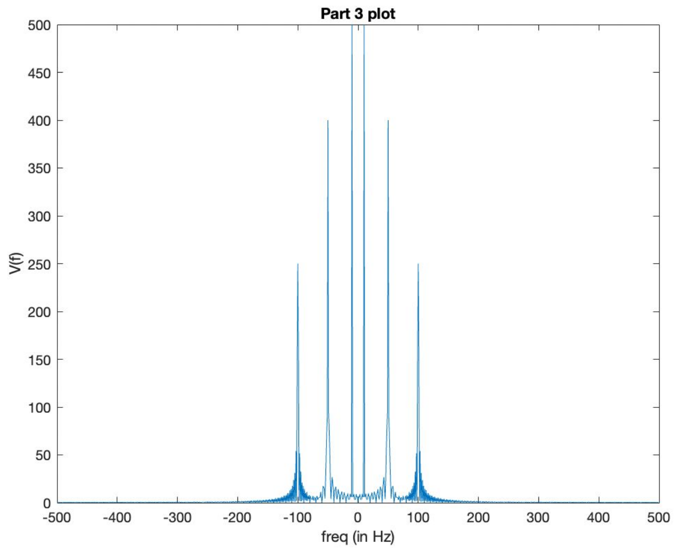
Note: This code continues from the previous code.

```
% part b) spectrum of the generated non stationary signal

fftFull = fftshift(fft(nonStationary));

%length of the FFT of non stationary signal in frequency domain
Len_f = length(fftFull);

%iterating freq from -len/2 to +len/2
freq = -Len_f/2 : 1 : Len_f/2 - 1;

%converting each term of freq into frequency as for the 'k'th term
f=Fs.k/N
freq = Fs*freq/Len_f;

%Plotting
plot(freq,abs(fftFull));
title("Part 3 plot");
xlabel("freq (in Hz)");
ylabel("V(f)");

%individual frequency spectrum

% (1)for duration 0 to 0.2 sec

fft1 = fftshift(fft(nonStationary(1:0.2*Fs),8192));

%length of the FFT of non stationary signal in frequency domain
Len_f = length(fft1);

%iterating freq from -len/2 to +len/2
freq = -Len_f/2 : 1 : Len_f/2 - 1;

%converting each term of freq into frequency as for the 'k'th term
f=Fs.k/N
freq = Fs*freq/Len_f;

%Plotting
plot(freq,abs(fft1));
title("sinewav1 frequency spectrum");
xlabel("freq (in Hz)");
ylabel("V(f)");
xlim([-50,50]);

% (2)for duration 0.2 sec to 0.5 sec

fft2 = fftshift(fft(nonStationary(0.2*Fs:0.5*Fs),8192));

%length of the FFT of non stationary signal in frequency domain
Len_f = length(fft2);

%iterating freq from -len/2 to +len/2
freq = -Len_f/2 : 1 : Len_f/2 - 1;
```

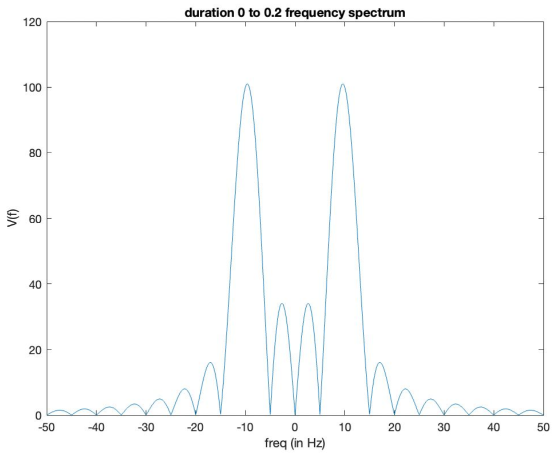
```
%converting each term of freq into frequency as for the 'k'th term
f=Fs.k/N
freq = Fs*freq/Len_f;

%Plotting
plot(freq,abs(fft2));
title("sinewav2 frequency spectrum");
xlabel("freq (in Hz)");
ylabel("V(f)");
xlim([-150,150]);

% (3)for duration 0.5 sec to 1sec

fft3 = fftshift(fft(nonStationary(0.5*Fs:end),8192));

%length of the FFT of non stationary signal in frequency domain
Len_f = length(fft3);

%iterating freq from -len/2 to +len/2
freq = -Len_f/2 : 1 : Len_f/2 - 1;

%converting each term of freq into frequency as for the 'k'th term f
Fs.k/N
freq = Fs*freq/Len_f;

%Plotting
plot(freq,abs(fft3));
title("sinewav3 frequency spectrum");
xlabel("freq (in Hz)");
ylabel("V(f)");
xlim([-200,200]);
```

Plots :

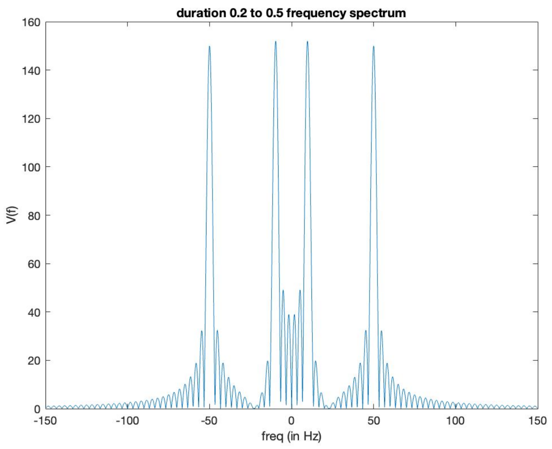
1. For multitone non-stationary signal



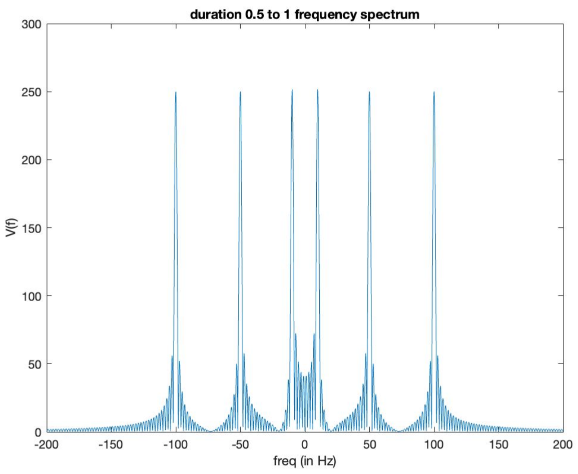
2. For duration 0 to 0.2 sec



3. For duration 0.2 to 0.5 sec



4. For duration 0.5 to 1 sec



Observations :
We can see that the frequency spectrum of the full duration of the non stationary signal is not the same as the ones taken in a smaller sub duration.

Inference and Conclusion :
There is a difference in the frequency response of the full duration and a sub duration because when we take the frequency response of the full duration it gives us peaks at the frequencies which are present in the overall duration of the signal, where as when we pick a small sub duration of the signal we can see that only a few frequencies are present at that instant. Hence we can conclude that taking the frequency response is not the right way to get the characteristics of a non stationary signal
To get a complete frequency understanding of speech we need to go for STFT (Short Term Fourier Transform), STFT is the technique used for plotting a spectrogram.

<u>Task D part(a)</u> :

Aim : To record the phrase **“Saakshat Speech Processing”**,resample it to 8 kHz and plot it.

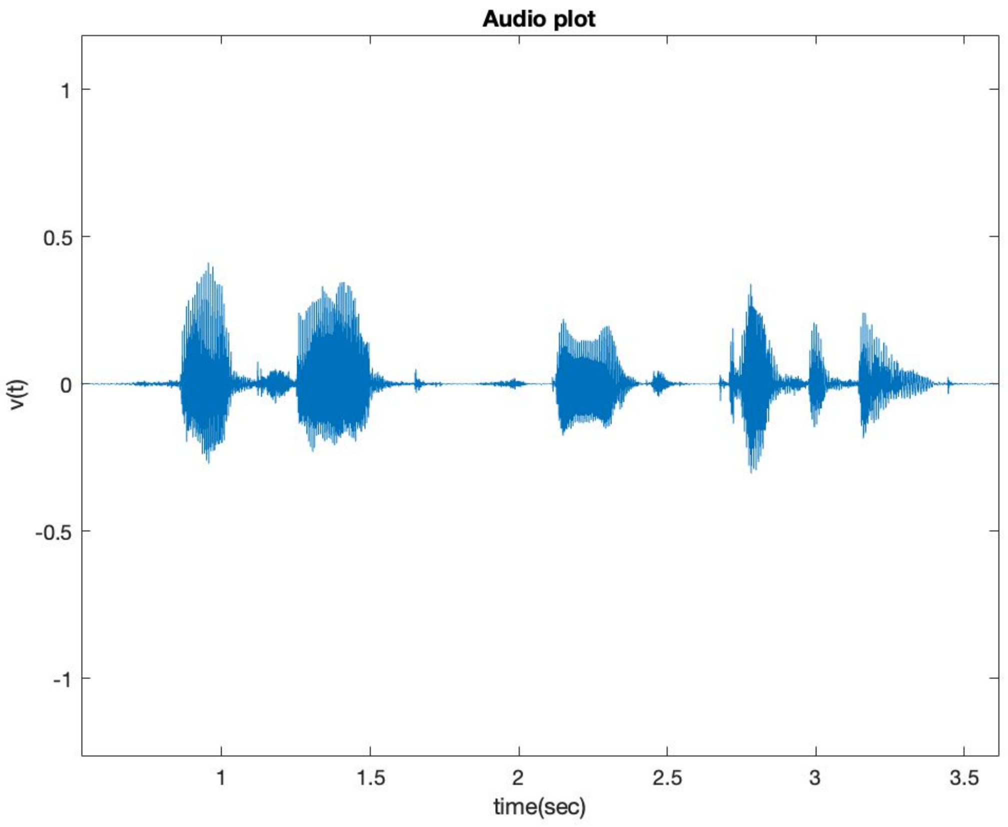
Procedure :
1. Record the audio with sampling frequency 48 kHz.
2. Use the resample() function for resampling audio from 48 kHz to 8 kHz.
3. Then plot it.

**The code is self explanatory

Code :

```
%%-----------------------------Part D ------------------------------
%Reading the original recorded audiofile
[y,fs] = audioread('SaakshatSP.wav');

%time duration of the given audio file
time_duration = length(y)/fs;

%re sampling frequency in Hz
Fs = 8000;

%using inbuilt resample function
y_new = resample(y,Fs,fs);

%generating the time axis
t = 0 : 1/Fs : time_duration - 1/Fs;

%plotting
plot(t,y_new);
ylim([-4*max(y_new) 4*max(y_new)]);
xlabel("time(sec)");
ylabel("v(t)");
```

Plot :

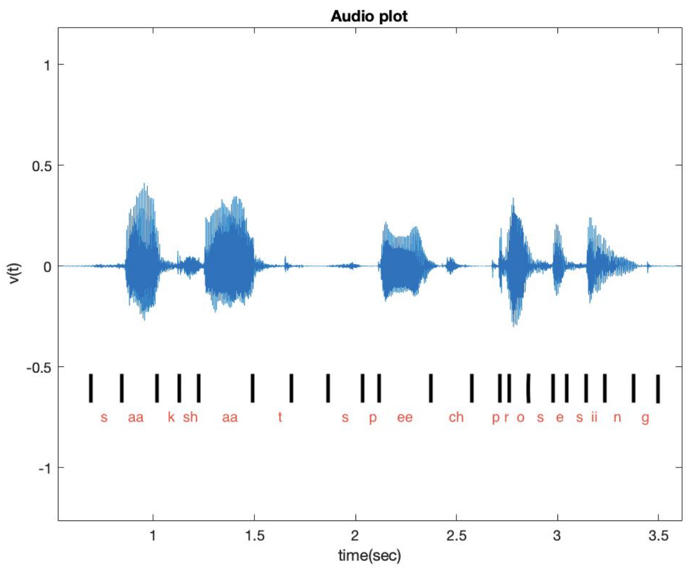


<u>**Task D part(b)**</u> :

Aim : To study the spectral properties of different categories of sound present in the recorded audio.

Procedure :

1. To identify different sounds in the giver audio plot.



2. Finding different categories of sound, in this experiment we found the following categories of sounds.
   a. Vowels (voiced/periodic) : aa , i , e , o
   b. Sibilant sounds : s , sh
   c. Nasal sounds : n
   d. Unvoiced and unaspirated consonants : ch, t, k
3. Plotting the frequency spectrum for each type of sounds. For time domain plotting we don't write a different code, instead we will zoom into the existing plot and crop out the required part.

Code :

1. Vowels (aa):

```
%resampling frequency
Fs = 8000;
%Taking aa part from audio file
aa_sound = y_new(1.24*Fs : 1.51*Fs);
```

```
%Frequency domain plotting
fft_aa = fftshift(fft(aa_sound,8192));

%length of the FFT of non stationary signal in frequency domain
Len_f = length(fft_aa);

%iterating freq from -len/2 to +len/2
freq = -Len_f/2 : 1 : Len_f/2 - 1;

%converting each term of freq into frequency as for the 'k'th term
f=Fs.k/N
freq = Fs*freq/Len_f;

%plotting
plot(freq,abs(fft_aa));
title("aa sound");
xlabel("freq (in Hz)");
ylabel("V(f)");
```

2. Sibilant (sh):

```
%resampling frequency
Fs = 8000;
%Taking sh part from audio file
aa_sound = y_new(1.11*Fs : 1.25*Fs);
%Frequency domain plotting
fft_sh = fftshift(fft(sh_sound,8192));

%length of the FFT of non stationary signal in frequency domain
Len_f = length(fft_sh);

%iterating freq from -len/2 to +len/2
freq = -Len_f/2 : 1 : Len_f/2 - 1;

%converting each term of freq into frequency as for the 'k'th term
f=Fs.k/N
freq = Fs*freq/Len_f;

%plotting
plot(freq,abs(fft_sh));
title("aa sound");
xlabel("freq (in Hz)");
ylabel("V(f)");
```

3. Nasal (n):

```
%Taking n part from resampled audio file
n_sound = y_new(3.18*Fs : 3.38*Fs);

fft_n = fftshift(fft(n_sound,8192));

%length of the FFT of non stationary signal in frequency domain
Len_f = length(fft_n);
```

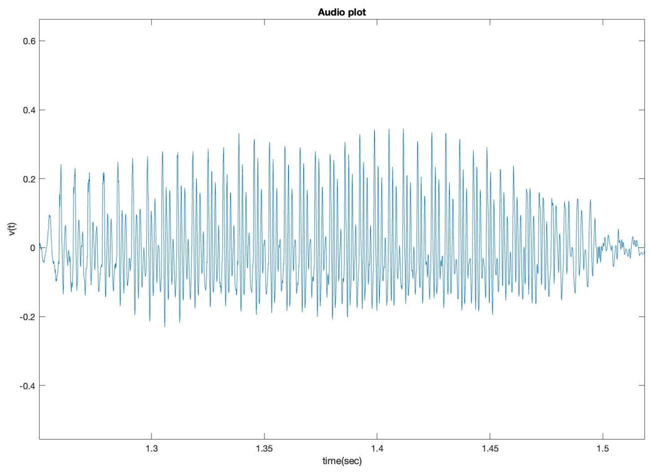
```
%iterating freq from -len/2 to +len/2
freq = -Len_f/2 : 1 : Len_f/2 - 1;

%converting each term of freq into frequency as for the 'k'th term
f=Fs.k/N
freq = Fs*freq/Len_f;

%plotting
plot(freq,abs(fft_n));
title("sh sound");
xlabel("freq (in Hz)");
ylabel("V(f)");
```

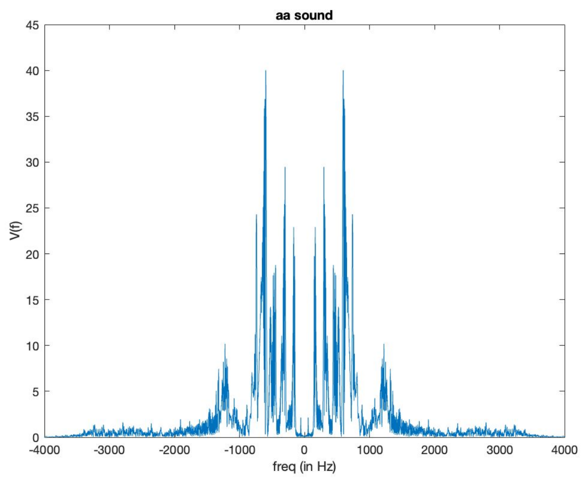
4. Unvoiced and unaspirated consonant (t):

```
%Taking t part from resampled audio file
t_sound = y_new(1.6*Fs : 1.7*Fs);

fft_t = fftshift(fft(t_sound,8192));

%length of the FFT of non stationary signal in frequency domain
Len_f = length(fft_t);

%iterating freq from -len/2 to +len/2
freq = -Len_f/2 : 1 : Len_f/2 - 1;

%converting each term of freq into frequency as for the 'k'th term
f=Fs.k/N
freq = Fs*freq/Len_f;

%plotting
plot(freq,abs(fft_t));
title("t sound");
xlabel("freq (in Hz)");
ylabel("V(f)");
```

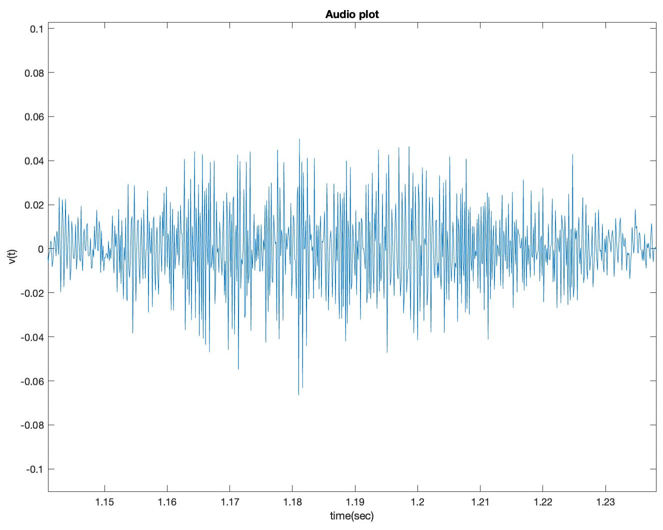
Plots :

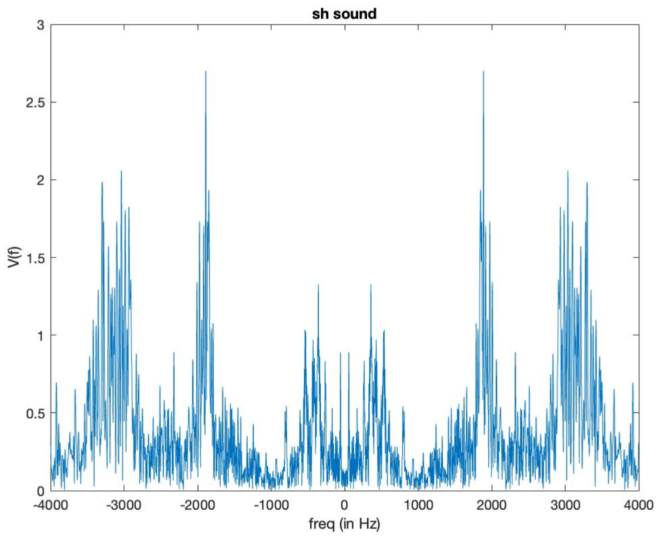
1. Vowel (aa)



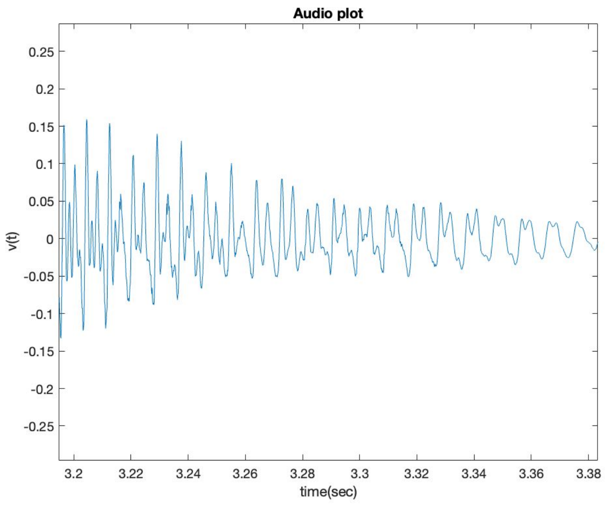


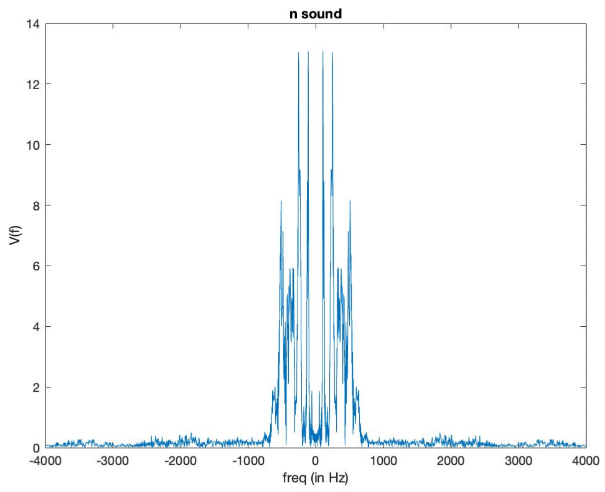
2. Sibilant sound (sh):



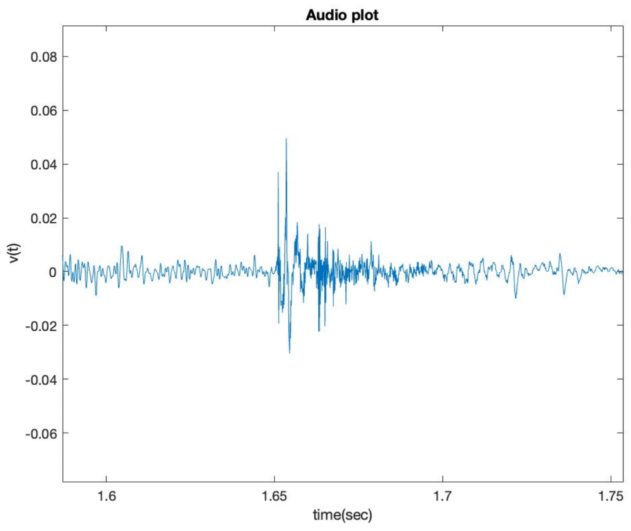


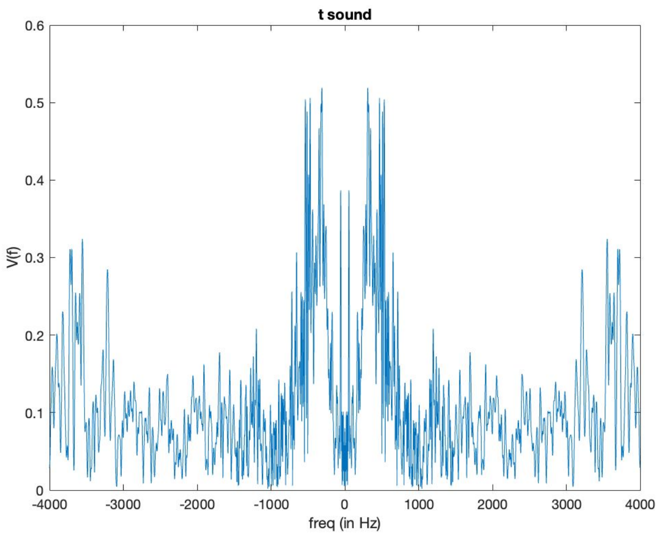
3. Nasal (n):





4. Unvoiced and unaspirated consonant (t):





Observations :
We observe that different sounds have different frequency characteristics, the vowels and nasals definite frequencies, whereas the consonants and sibilants have a mixture of all the frequencies.

Inference and Conclusion :
By the observations we can conclude that the speech signal as a whole is non stationary.