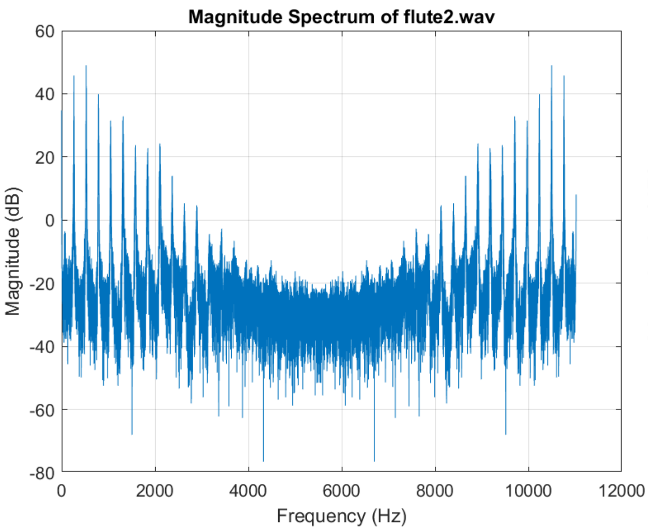
EE386 Digital Signal Processing Lab Jul-Dec 2021

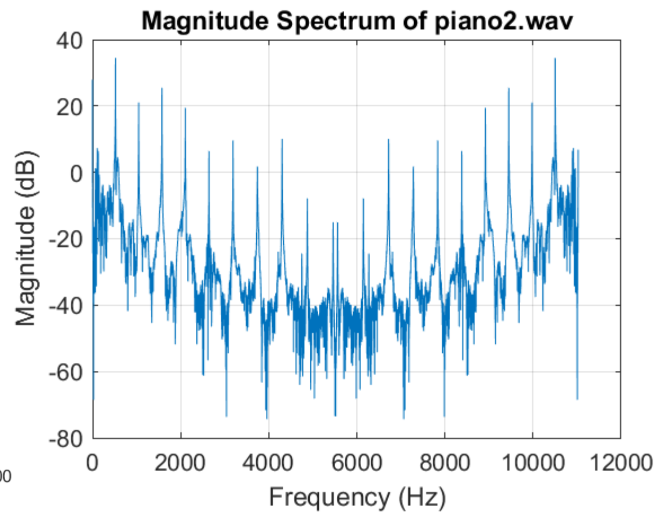
Experiment No 1

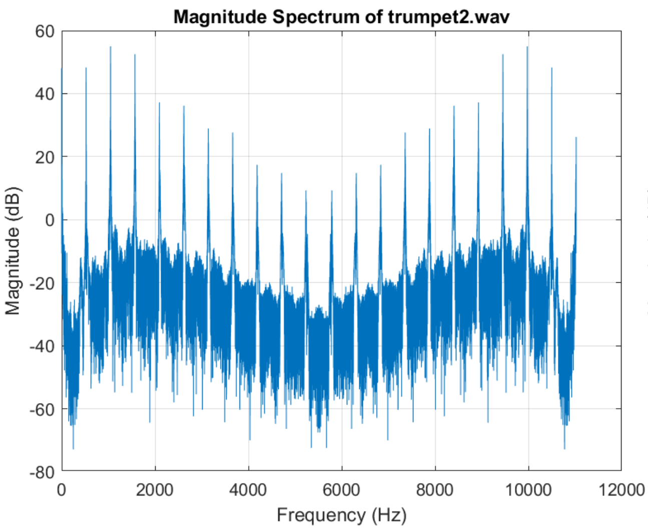
Author: Prakhar Goel *Email:* prakhargoel.211ee247@nitk.edu.in

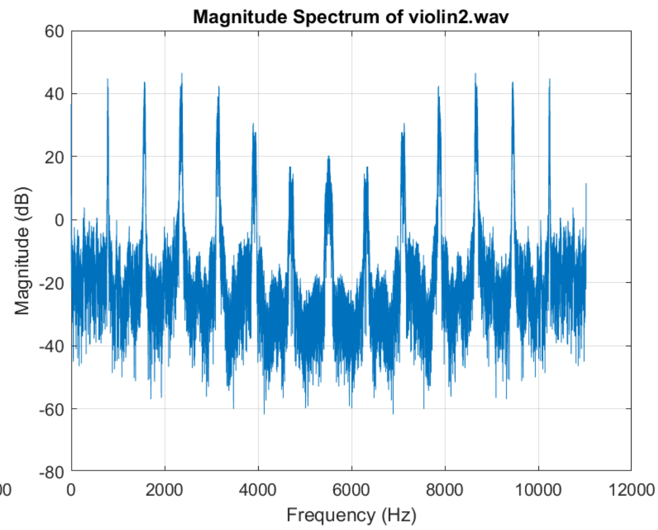
Problem 1 (Generating Signals)

(1) Write a function to plot the magnitude spectrum of a signal using the FFT function.You are supplied with the recordings of four different instruments (piano, trumpet, violin and flute). Find the frequency value (in Hz)corresponding to the fundamental harmonic (the peak) of the four different recordings . Use the FFT algorithm and plot magnitude (in dB scale) versus frequency (Hz).









Fundamental frequency of flute: 524.33 Hz

Fundamental frequency of piano: 523.62 Hz

Fundamental frequency of trumpet: 1047.21 Hz

Fundamental Frequency of violin: 2369.12 Hz

(1.b) Find the β of fluteβ.wav for which the fundamental frequency of piano2.wav is closest.

We observed the fundamental frequency of all the flute wav files as 1063.2Hz , 524.3 Hz, 793.5 Hz and 524.4 HZ respectively. It was observed that the frequency of piano wav file is 523.62 Hz. On comparing it was found that Flute2.wav was the closest.

Problem 2 (Key-Lock of my whistle)

(Recording)

A 3 second recording of my whostling is done.

(Plotting)

The plot of my whistle is being done using the like in Q1 and the fundamental frequency is found out of the whistle.

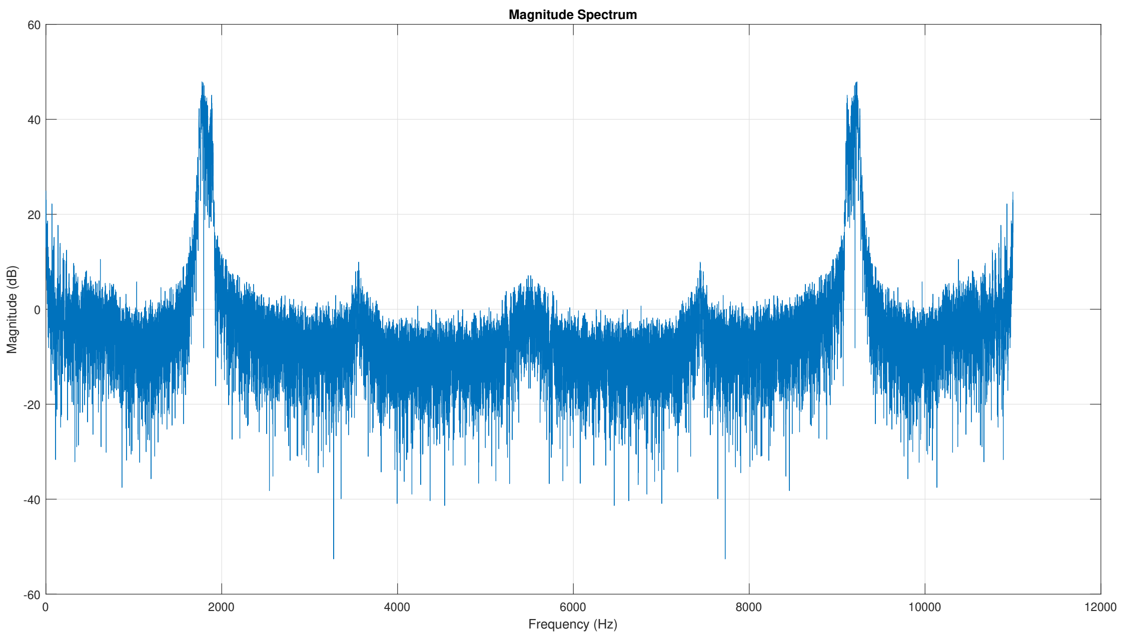


Figure 1: Magnitude spectrum of sample whistle which is acting as reference or key for the lock.

(Subproblem 3: Design a keylock: Write a function to record 3 second audio clips that output "ACCESS GRANTED" when the fundamental frequency of the recorded audio matches with the fundamental frequency of the reference within a 5% error, and "ACCESS DENIED" otherwise.)

(Solution)

A function `whistle_recorder` was defined which recorded a whistle and saved the same in a `.wav` file. In another file named `whistle_check`, the whistle was compared with sample whistle. The tolerance for unlocking was taken to be ± 5 % around the sample fundamental frequency i.e. 1774.6667 Hz. Plot and magnitude spectrum of test whistle was also plotted later. A whistle was recorded which was able to unlock the lock. It's plot, magnitude spectrum and fundamental frequency is given below:

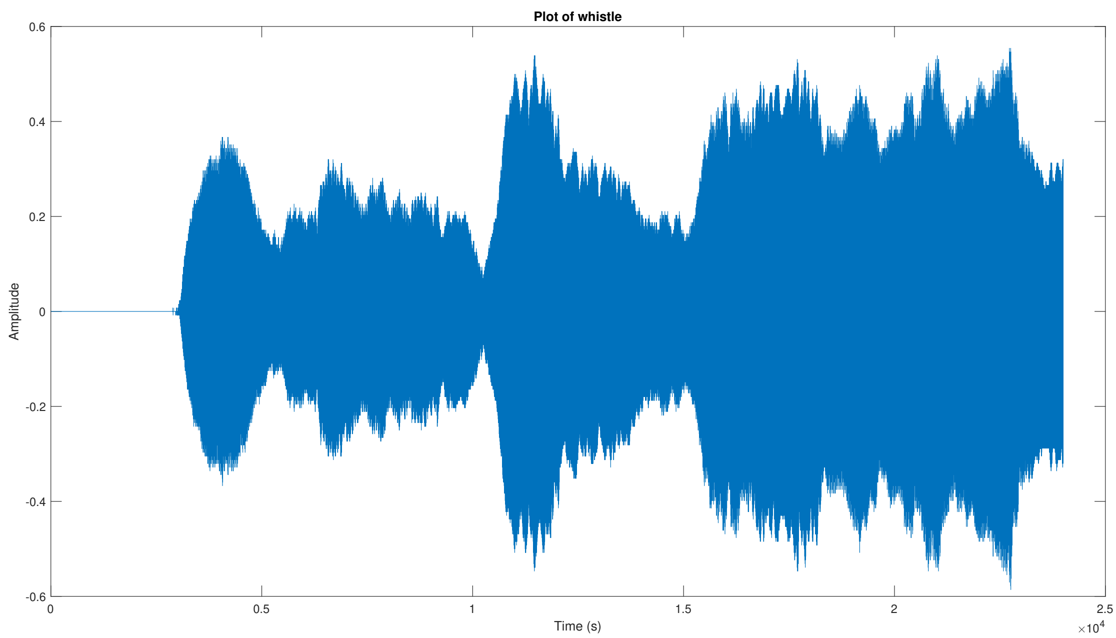


Figure 2: Plot of whistle which unlocked the lock.

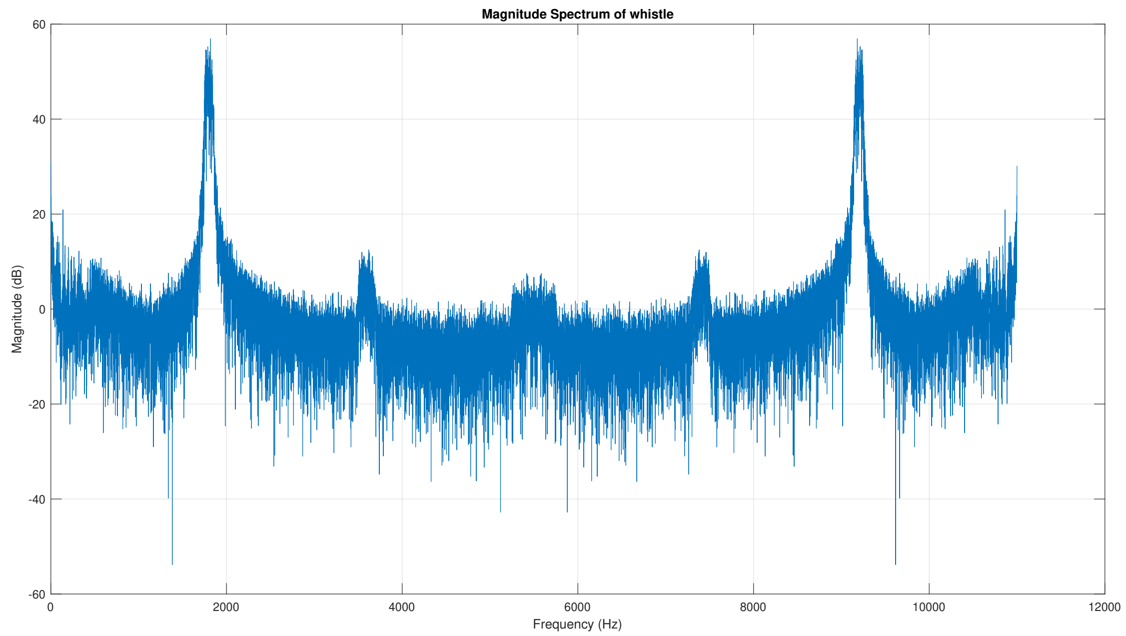


Figure 3: Magnitude spectrum of whistle which unlocked the lock.

Test-Case 2: ACCESS DENIED

A whistle was recorded which was unable to unlock the lock. It's plot, magnitude spectrum and fundamental frequency is given below:

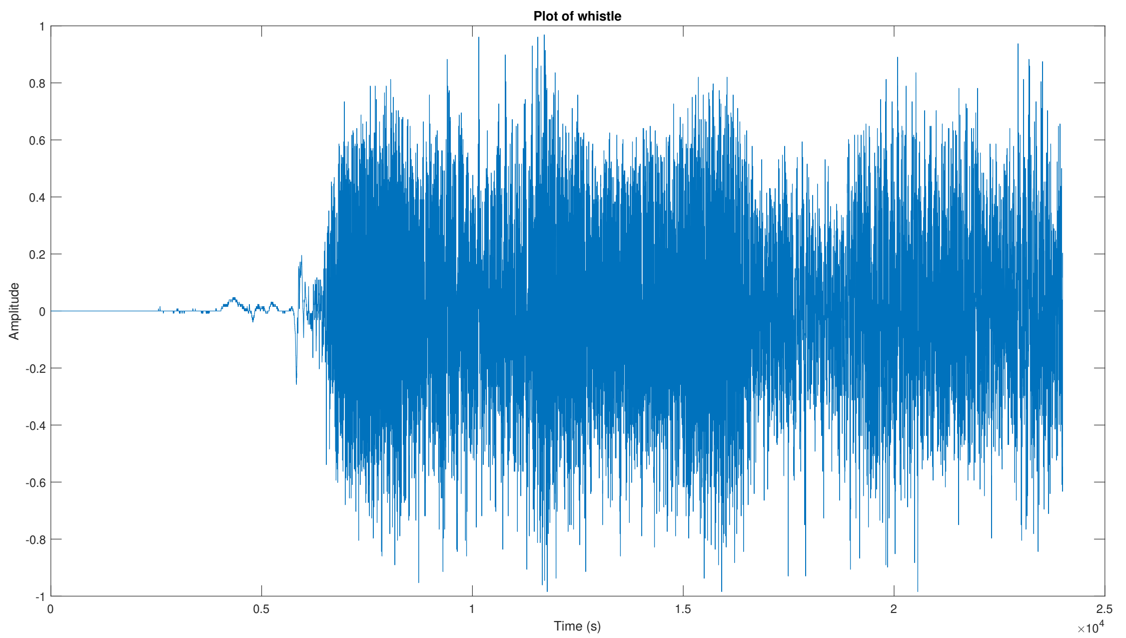


Figure 4: Plot of whistle which failed to unlock the lock.

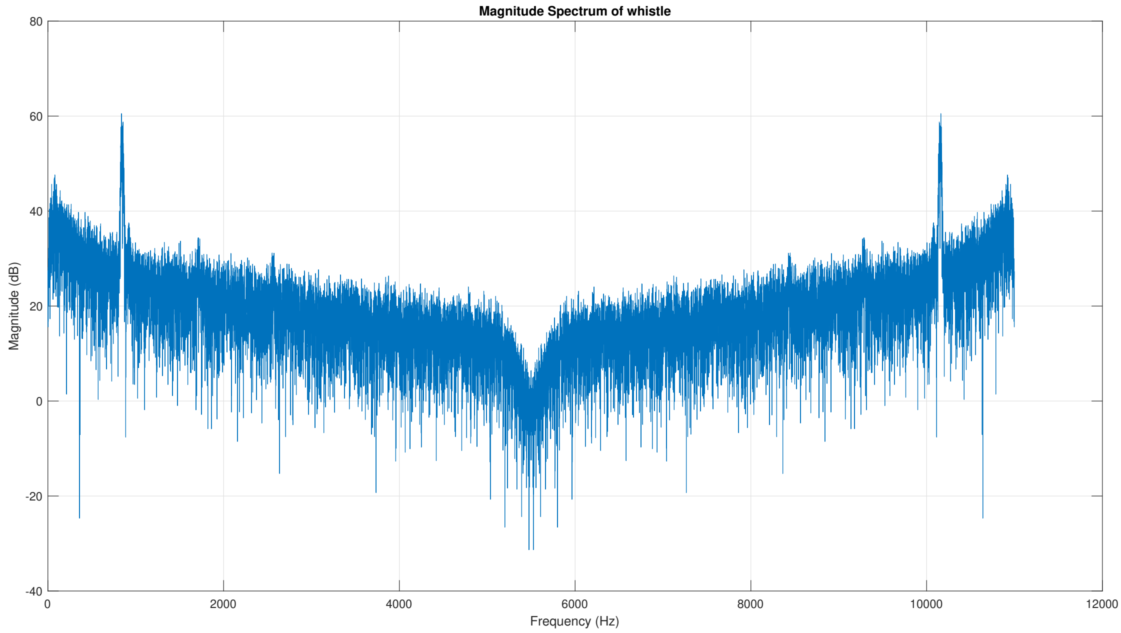
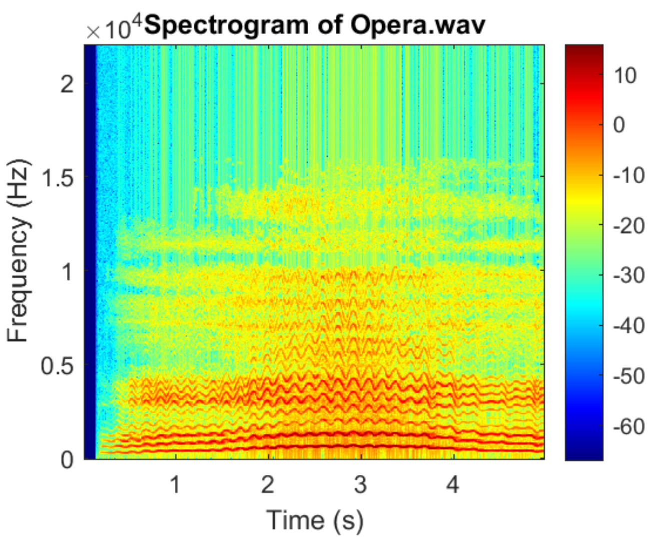
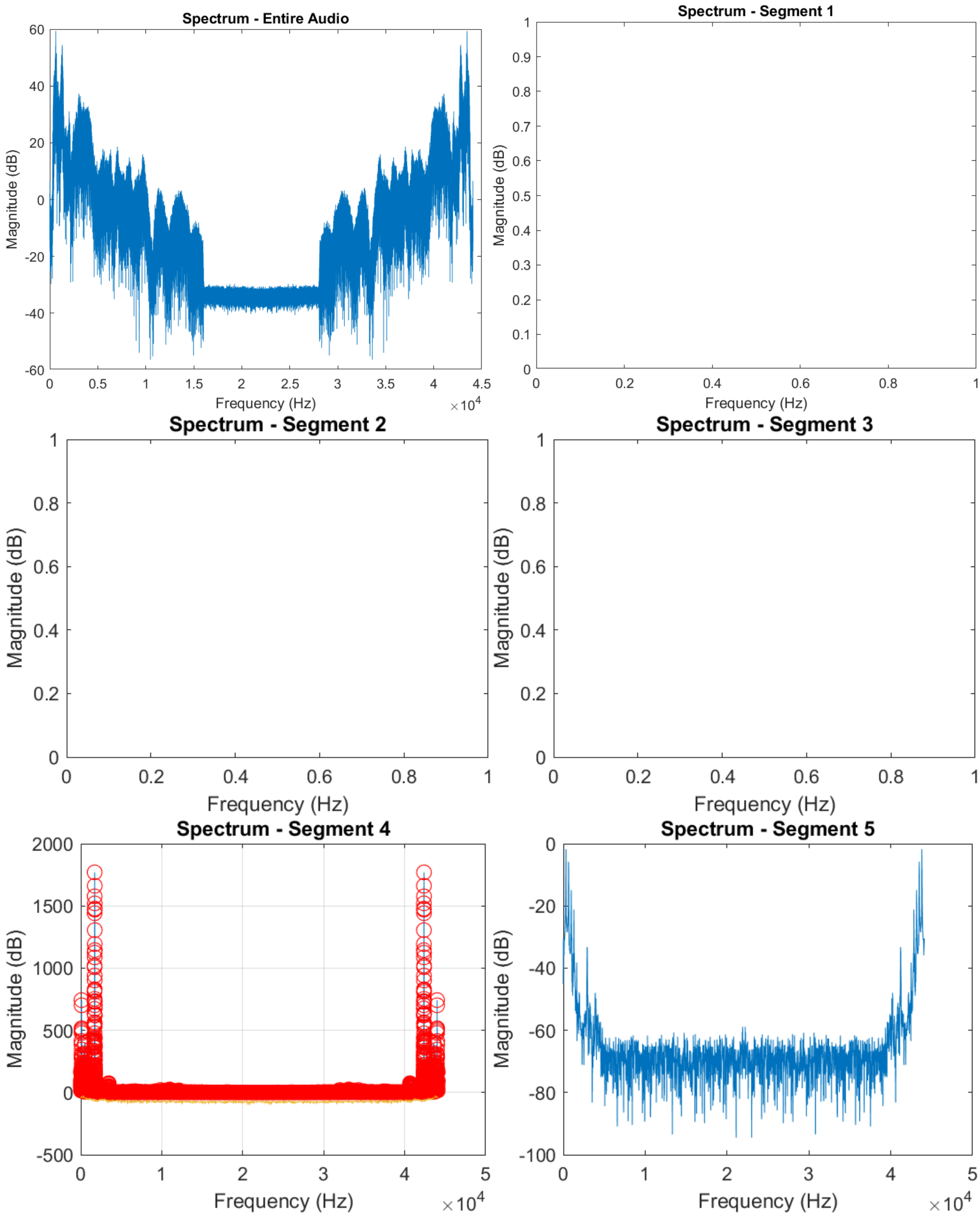
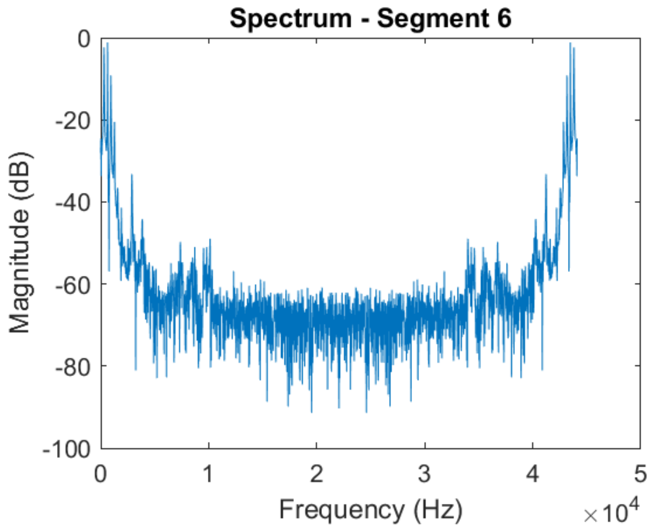


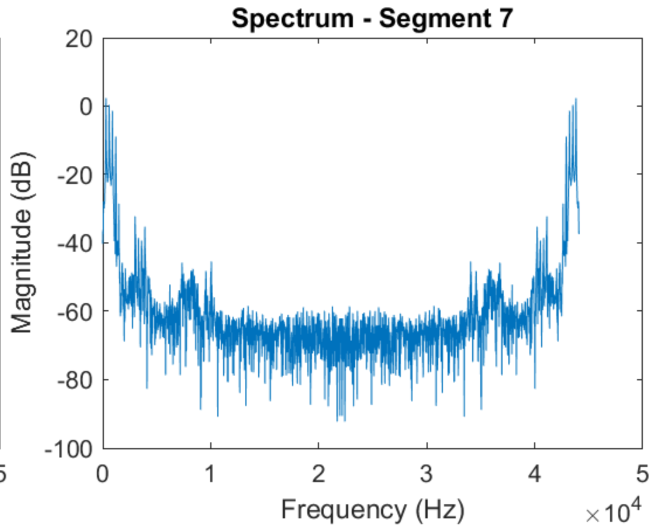
Figure 5: Magnitude spectrum of whistle which failed to unlock the lock.

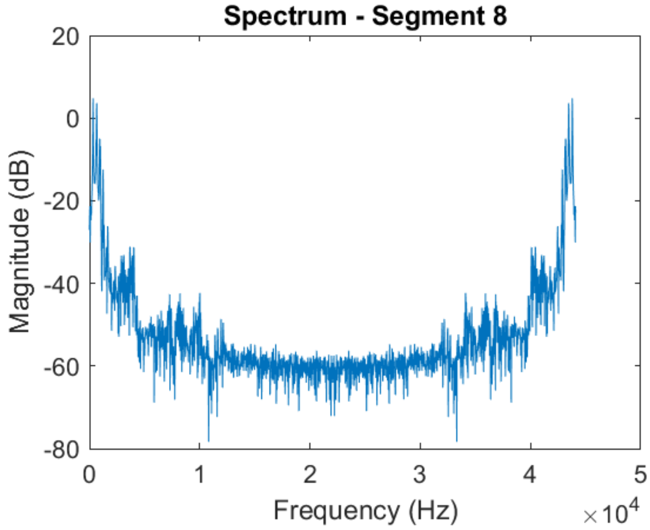
Problem 3 **Plot the spectrum of Opera.wav. How do you capture the temporal variations of the spectrum? The .wav file has around 22000 samples. Try plotting the spectrum for every 2000 samples (separately) and compare the figures (around 10 figures). Do you see the difference in the location of the dominant peaks in each figure? Comment on it.**

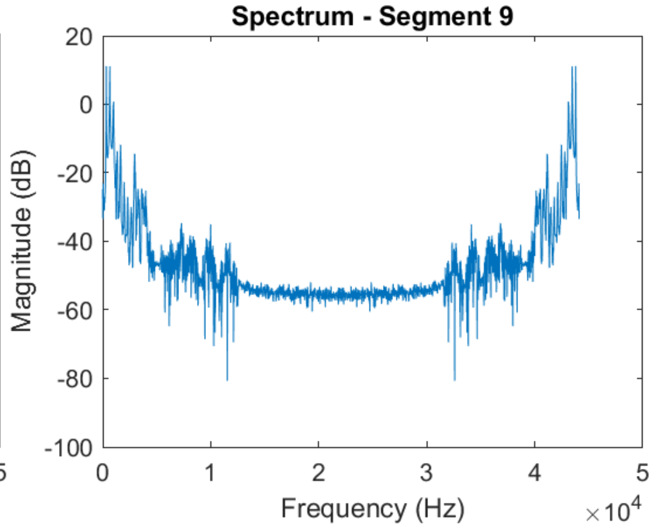


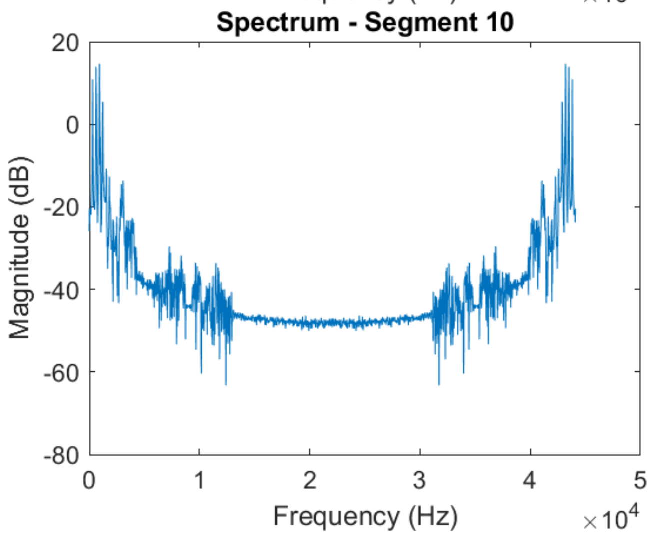

Spectrum - Entire Audio
Magnitude (dB)
Frequency (Hz)
$\times 10^4$
Spectrum - Segment 1
Magnitude (dB)
Frequency (Hz)
Spectrum - Segment 2
Magnitude (dB)
Frequency (Hz)
Spectrum - Segment 3
Magnitude (dB)
Frequency (Hz)
Spectrum - Segment 4
Magnitude (dB)
Frequency (Hz)
$\times 10^4$
Spectrum - Segment 5
Magnitude (dB)
Frequency (Hz)
$\times 10^4$

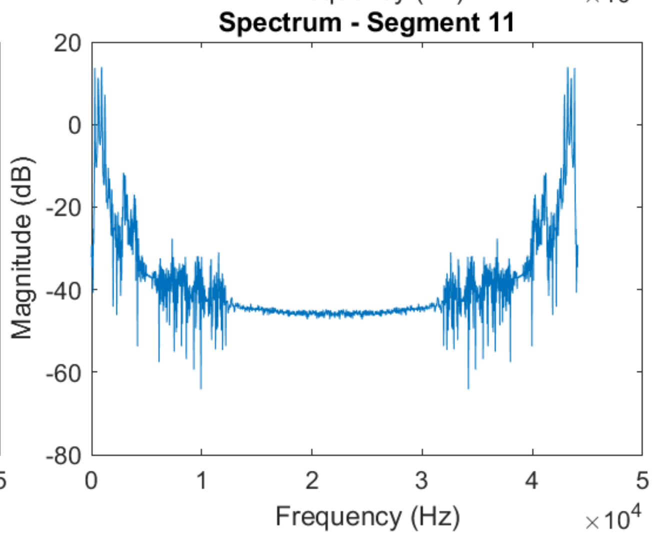












Opera.wav had 2,20,000 samples, as found by using the len() function. I performed FFT on the audio, taking 20000 samples at a time, to obtain a rudimentary STFT operation. Thus, it is simultaneously in the time and frequency domain.

Observations:

The dominant peaks, i.e., the fundamental peaks for each of the plots are different from each other. It varies from 139Hz, 173Hz, 190Hz, 230Hz, 267Hz, 282Hz, 282Hz, 283Hz, 247Hz, 219Hz, 183Hz. The frequencies increase and reach a maximum at the 8th window, following

which they decrease. This matches what we should expect after listening to Opera.wav. While listening to the audio, you will observe that the singer sings around 15 notes (Do-Re-MiFa-So-La-Ti-Do-Ti-La-So-Fa-Mi-Re-Do). Hence, theoretically, by creating 15 windows, we are isolating each of these notes. Thus we should be able to obtain the frequencies of each of the 15 notes. These will be increasing in frequency till they hit a maximum at the 8th window (the second 'Do' note), following which they decrease, and each of the frequencies to the left of this maximum mirror their corresponding values on the right. The same thing is observed in our case with 11 windows, but since the window taken is not perfect, we are getting erroneous results, although it conforms to the expected trend. The spectra obtained were as follows:
Github: https://github.com/Prakhar3006/DSP