Mini project 3

Study and demonstrate visualizations tools to be used with DSP (Javascript or Python)

Chenhui Zhu
zhuch@bu.edu

1. Introduction

In this mini project, an audio signal is displayed virtually by using the combination of three different python libraries. This signal can be shown both in time domain and frequency domain.

2. Library Choices

Before introducing the progress of how to display the signal, the libraries' information will be introduced firstly. There are three different libraries used in this mini project, such as Matplotlib, Wave, Numpy.

2.1 Matplotlib

The Matplotlib license is based on the Python Software Foundation (PSF) license. It is a python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Besides, plots, histograms, bar chart, etc. can be easily generated by its' modules, called pyplot.

2.2 Numpy

Numpy is a fundamental package for scientific computing with Python. It contains serval main functions, such as a powerful -dimensional array object, broadcasting functions, useful linear algebra, etc. In this project, Numpy is used as an efficient multi-dimensional container of audio data.

2.3 Wave

The wave module provides an easy access to wav file for python user. In this project, it is used to import the audio file and transform to the data.

3. Methods

At the beginning of the process, a wav file is imported. By using the wave modules, the audio's amplitudes are stored in the numpy array. At last, the wave form is plotted by the matplotlib. The figure 1 shows the code for this project.

```python
import wave
import numpy as np
import matplotlib.pyplot as plt

f = wave.open(r"EC516_voice.wav","rb")
params = f.getparams()
[nchannels, sampwidth, framerate, nframes] = params[:4]
str_data = f.readframes(nframes)
f.close()
wave_data = np.fromstring(str_data,dtype = np.short)
wave_data.shape = -1,2
wave_data = wave_data.T
time=np.arange(0,nframes/2)/framerate
# print(params)
plt.figure(1)
plt.subplot(211)
plt.plot(time,wave_data[0])
plt.xlabel("time/s")
plt.title('Wave')


N=40000
start=0

df = framerate/(N-1)

freq = [df*n for n in range(0,N)]

wave_data2=wave_data[0][start:start+N]
c=np.fft.fft(wave_data2)*2/N

plt.subplot(212)
plt.plot(freq[:round(len(freq)/2)],abs(c[:round(len(c)/2)]),'r')
plt.title('Freq')
plt.xlabel("Freq/Hz")
plt.show()
```

Figure 1. The code for mini project3.

4. Results

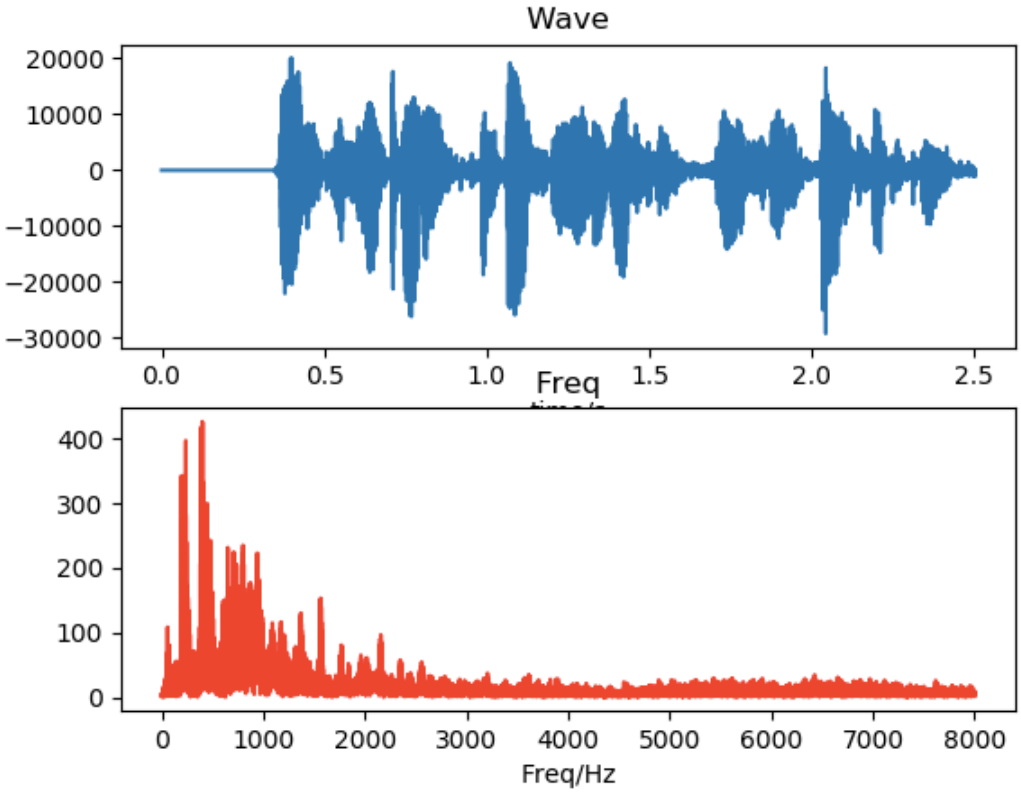

Figure 2. the waveform of an audio record

As the figure 2 shown above, the first image is the waveform of the origin audio record, which is shown in time domain. And the second image is the frequency response of the audio record, which is generated by the FFT.