Coding Project 2: Parsing musical frequency signatures

Hao Li
Department of Applied Mathematics
University of Washington
`haoli25@uw.edu`

Abstract

In this project, we aim to isolate and reconstruct the different types of instruments from a sound clip. The clip is from the first forty seconds of the song *'I'm Shipping Up to Boston'* by Dropkick Murphys. The nice thing about this particular song is that the instruments are introduced one at a time. Hence, the key idea for this project is to use the *Gabor Transform* to determine the frequencies as it changes over the time. In other words, we are transforming signals over a specific window of time and slide it through. By doing so, we will be able to create spectrograms and study the frequencies from the clip. Then we will determine the threshold for three instruments: drum, electric-guitar, and banjo and try to isolate their specific frequencies from the original clip.

1 Introduction

For this sound clip, we are using a sample rate of 44100 which will give us a length of 1938240 `np.array`. In order to make *MATLAB/Autograder* to run, we will split the the sound clip into four smaller chunks so that we will apply Gabor transform to the four chunks individually and generate four spectrograms. To isolate the instrument, we will simply set all points that are out side of the threshold range to zero and keep the rest.

2 Theoretical Background

The limitations of the direct Fourier transform is that it cannot localize a signal in both the time and frequency domains. The Hungarian physicist/mathematician/electrical engineer Gábor Dénes first proposed a formal method for localizing both time and frequency by introducing a Gaussian function to the Fourier kernel $g(\tau - t)$ [1].

2.1 The Gabor Transform

The Gabor transform is a special case of the short-time Fourier transform which is used to determine the sinusoidal frequency and phase content of local section of a signal as it changes over time [3]. Gábor introduced the the kernel

$$g_{t,\omega}(\tau = e^{i\omega\tau}\, g(\tau - t) \tag{1}$$

where a Gaussian function is multiplied the Fourier kernel. Then the *Gábor transform* is defined as the following:

$$\mathcal{G}[f](t,\omega) = \tilde{f}_g(t,\omega) = \int_{-\infty}^{\infty} f(\tau)\bar{g}(\tau - t)e^{-i\omega\tau}\, \mathrm{d}\tau = (f, \bar{g}_{t,\omega}) \tag{2}$$

where the bar denotes the complex conjugate of the function. Note that we assume the Fourier kernel g to be real and symmetric for some nice properties to hold. The parameter τ gives us the control to pick out the frequency information at each instant time by sliding the time-filtering window down the whole signal. Similar to Fourier transform, Gábor transform has its inverse which is defined as the following:

$$f(\tau) = \frac{1}{2\pi}\frac{1}{\|g\|^2}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\tilde{f}_g(t,\omega)g(t-\tau)e^{i\omega\tau}\,\mathrm{d}\omega\,\mathrm{d}t. \tag{3}$$

Because the Gábor transform has both t and ω as variables, its inverse will be a double integral as we see above.

2.2 The Discrete Gabor Transform

like the discrete Fourier transform, the Gábor transform is computed by discretizing the time and frequency domain in practice. Thus a discrete version of the transform for a given signal f of length L is defined as the following [2].

$$\mathcal{G}[f](m,n) = \sum_{l=0}^{L-1} f(l)\bar{g}(l-nt_0)e^{-2\pi i m\omega_0 l/L}. \tag{4}$$

Note that if $0 < t_0, \omega_0 < 1$, the signal has been oversampled, resulting in time frames that provide exceptional localization of the signal in both the time and frequency domains. On the other hand, if $\omega_0, t_0 > 1$, then the signal is under-sampled and these windows are incapable of reproducing the signal [1].

3 Numerical Methods

The basic algorithm can be implemented as follows:

i Splitting our original signal into four pieces which yield the discretization points for each piece to be `n=484560` and the length of each signal to be `n/Fs` where `Fs=44100` is our sample rate. Then t and τ will be from 0 to `L` with step-size `1/Fs` and 0.1 respectively.

ii Set the Gaussian coefficient to $a = 400$, then for each τ_i, we are going to apply the Gábor filter to the signal and take it to the frequency domain by applying `np.fft.fft`.

iii Still inside the τ_i for loop, we are going to locate the absolute maximum value of the frequency from the last step in the range of [0:1800] so that we can make a standard Gaussian filter centered at those peak frequencies.

iv Apply the filter to the our Gábor transformed frequency and use `np.fft.fftshift` to reorder our frequencies. After the for loop, we will have the spectrogram for our first piece at each τ. Repeat the same process to find all four pieces.

v Take our original signal again and set the frequency range for each instrument that we want to isolate.

vi For each instrument, we set the frequencies of the original signal that are outside our range to zero or keep it the same otherwise. Finally, we will take `np.fft.ifft` and `np.fft.fftshift` to take them back to the right time domain.

4 Results

4.1 Spectrograms

The following picture is the spectrograms for each piece. Note that we are taking the `np.log(X+1)` where X is original value of the spectrogram. In order to have better visualization, we also set y-limit from 0 to 700.

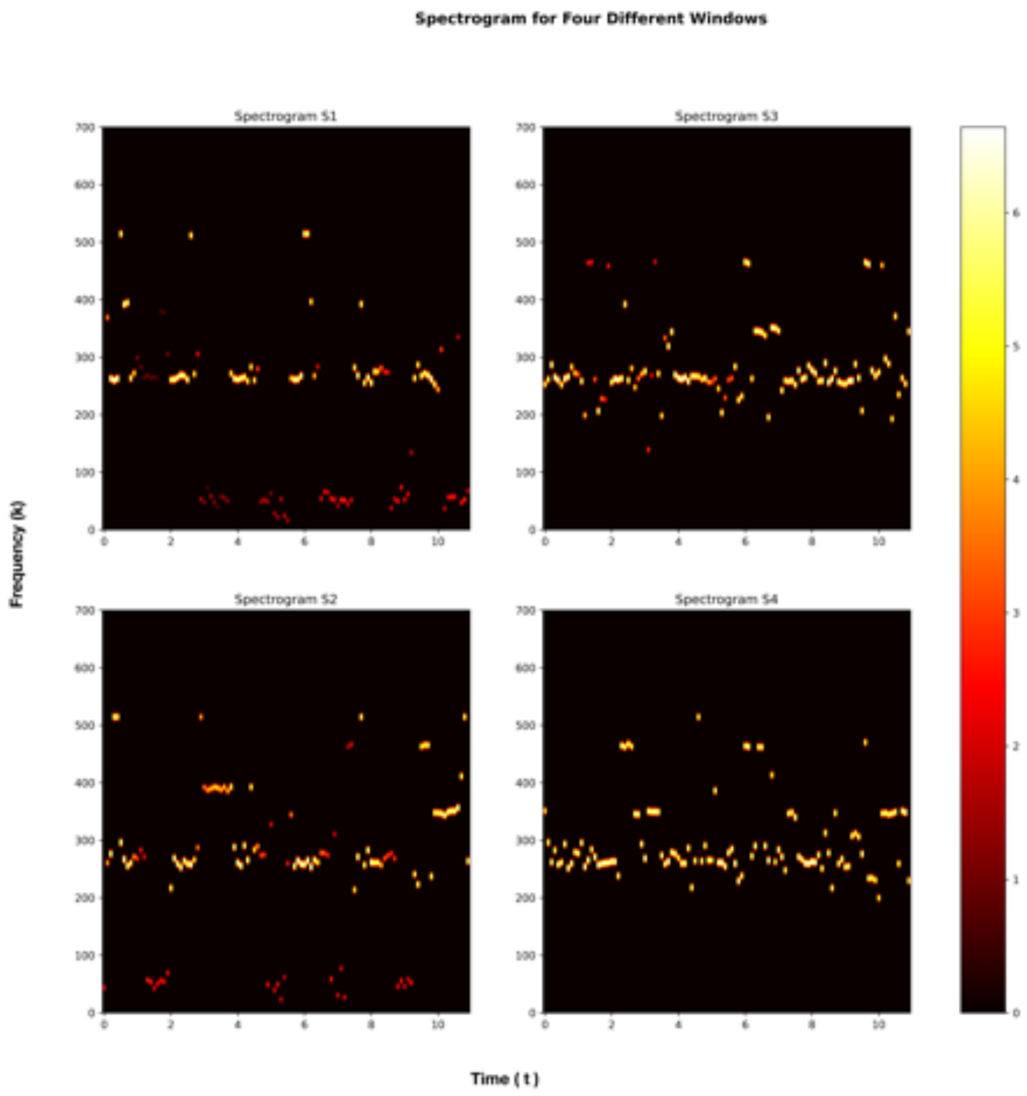

Figure 1: Spectrogram for the whole signal split into four consecutive windows, namely, S1, S2, S3 and S4

Upon first listening, the first instrument sounds like a cello because of its distinctive low frequency and it is then followed by a violin with a relatively high frequency. Then the next instrument is banjo because it has a very unique sound and at around 7 second, the drum come into play. Then at round the 14 second, the electric-guitar starts playing and i can barely hear the cello after 20 second later. It might not be accurate but we can verified by the looking at the plot.
First we see, we have a consistent signal at around 250 Hz and I'm pretty sure that is the drum because it plays almost the entire time in the sound clip. Next we see we have some regular beats at around 60 Hz in the first two windows and later we cannot hear it anymore. So, that is probability cello. For the first two windows, we also see some sparse signals at around 400 - 550 Hz. That might be banjo, guitar or violin.
I play piano so I found this music sheets online. Since the piano ranges from 27 - 4000 Hz, I played the intro part of this song on piano. The sound is in D major and the highest key in the first 40 seconds is F5 sharp which is around 700 Hz and most keys are between D4 to D5 (300 - 580 Hz). This is the tricky part because guitar, banjo, and violin all can play at this range. Therefore, if they are playing in the same key, the frequency will be the same, making it difficult to isolate these instruments based solely on frequency.

4.2 Instrument Isolation

By searching google, I also found that the drum is usually under 250 Hz which verified what we found from the spectrogram. The electric guitar is between 80 - 1200 Hz and banjo is between 294 - 588 Hz. Hence, it is hard to isolate banjo and guitar because the guitar can cover whole frequency range of banjo. The following plot is the amplitude (in absolute value) of each instrument.

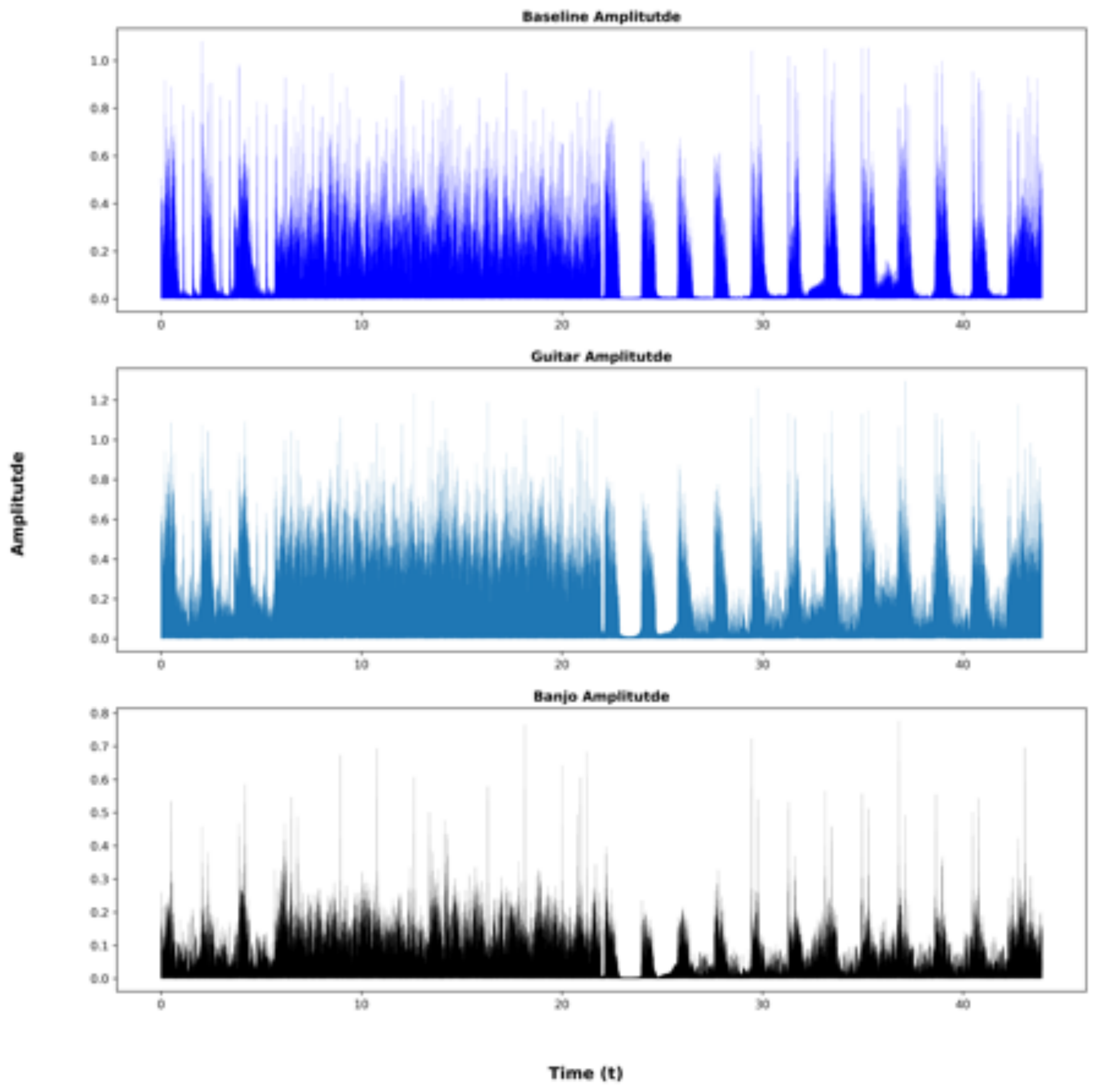

Figure 2: sound amplitude vs time for each instrument

One we noticed is that in general, guitar has a higher amplitude (average at around 0.5) and banjo has much lower amplitude (average at around 0.1).

5 Conclusion

In this project, we were challenged by localizing a signal in both the time and frequency domain. We tackled this problem by applying Gábor transform. We transformed signals over a specific window of time and slide it through. By doing so, we were able to create spectrograms and study the frequencies from the clip. However, during the isolation process, we realized that some string instruments had very wide range of frequencies. If they played the same key at the same time, we won't to isolate them clearly. In the future work, we can continue process this song because later we can try to isolate man's voice in the chorus part.

6 Acknowledgment

Thanks for the help from Professor Amin's office hour and users from the `# project-help` channel in `Amath 4/582` discord server.

References

[1] J.N. Kutz. *Data-Driven Modeling & Scientific Computation: Methods for Complex Systems & Big Data*. Data-driven Modeling & Scientific Computation: Methods for Complex Systems & Big Data. OUP Oxford, 2013.

[2] Christoph Wiesmeyr, Nicki Holighaus, and Peter Søndergaard. Efficient algorithms for discrete gabor transforms on a nonseparable lattice. *IEEE Transactions on Signal Processing*, 61, 07 2013.

[3] Wikipedia contributors. Gabor transform — Wikipedia, the free encyclopedia. `https://en.wikipedia.org/w/index.php?title=Gabor_transform&oldid=1138586695`, 2023. [Online; accessed 10-February-2023].