Rock & Roll and the Gabor Transform

Jiasui Qin

February 6, 2020

Abstract

When it comes to problems relating to signal filtering, we normally think about transferring data from the spatial domain to the frequency domain to extract the main frequency of the data, which then helps us identify where the filter should be centered at. Several methods have been used widely to achieve this, such as the famous Fourier transform. However, although we know what frequencies are present, we lose information about how the frequencies change over time. If our signals are neither stationary nor periodic, we might want to get both time and frequency information from the signal. This paper will focus on using the Gabor transform to solve this issue.

1 Introduction and Overview

We are given two music clips of the songs Sweet Child O' Mine by Guns N' Roses and Comfortably Numb by Pink Floyd. The first one is 14 seconds long, and the second one is 60 seconds long. Both pieces are converted to a vector representing the amplitude of the music in the time domain and the sampling rate in Hertz.

We will manipulate the data to achieve the following three research goals:

1. Reproduce the music score for the guitar in Sweet Child O' Mine and the bass in Comfortably Numb.

2. Isolate the bass in Comfortably Numb.

3. Reconstruct the music score for the guitar solo in Comfortably Numb.

To solve these problems, we will first apply the idea of the Gabor transform that helps us identify the frequency of the music at different time points. Then, based on these frequencies, we can filter out those unwanted frequencies that are not related to a specific instrument. It helps us isolate different instruments with different music scores and gives us a cleaner result that shows us the information about the specific instrument.

2 Theoretical Background

2.1 Gabor Transform

The basis of he Gabor transform is also called the short-time Fourier transform (STFT). In order to get time information and the associated frequency within that time period, we can split the data into multiple subdomains and compute the frequency of each subdomain using the Fourier transform. Since there might have sharp transitions between regions, the idea of sliding window is then proposed. We can apply the Fourier transform by sweeping a window with a window center τ and a window width a over the whole domain.

Let us consider a filter function $g(t)$. Then, shifting by τ and multiplying by a function $f(t)$ represents

the filtered function $f(t)g(t-\tau)$ with the filter centred at τ. That is, at a specific time τ, the function gives us the frequency components near that specific time.The Gabor transform is therefore given by[3]:

$$\hat{f}_g(\tau, k) = \int_{-\infty}^{\infty} f(t)g(t-\tau)\mathrm{e}^{-ikt}\,dt \tag{1}$$

There are many choices for the filter function g as long as g satisfies centain conditions. First of all, g must be real and symmetric. Second of all, the L_2-norm of g is set to unity. After determining the function of g, we need to decide the window width, which is a that we specified earlier. The size of the window plays an important role in deciding to what extent we can extract information of both time and frequency. In other words, we want to find the best window size that returns both time and frequency information as much as possible. If the window is set too large, then we are basically applying the Fourier transform over the whole signal and lose time information. If the window is too small, it would be hard to extract the frequency information correctly.

Just like Fourier transform, there's also the discrete version of the Gabor transform, since data in real life are mostly discrete. In this case, we will shift the filter with a discrete set of values of τ and find a discrete set of frequencies. Suppose m, n are integers and w_0, t_0 are positive constants, then the Gabor transform is given by[3]:

$$k = mw_0, \tau = nt_0$$

$$\hat{f}_g(m, n) = \int_{-\infty}^{\infty} f(t)g(t-nt_0)\mathrm{e}^{2\pi imw_0 t}\,dt \tag{2}$$

3 Algorithm Implementation and Development

3.1 Domain Discretization

I define L to be the total number of seconds that we want to examine. Therefore, the total range of the time domain t is given by the sampling rate Fs multiplied by L, which is noted as $[0,\ L * Fs]$. The frequency domain is $\frac{1}{L}[-\frac{n}{2}, \frac{n}{2}]$, where n is the length of the time domain. Note that it needs to be shifted using fftshift()[2] to transfer the interval to $[0, n]$ before plotting.

3.2 Gabor transform

I choose the Gaussian filter to be the filter function g. τ value increases from start to end with an interval of 0.1 seconds. The window width a for Sweet Child O' Mine and Comfortably Numb are set to 20 and 50 respectively. The filter function can be represented as

$$g(t-\tau) = \mathrm{e}^{-a((t-\tau)^2)} \tag{3}$$

General steps of the algorithm:

1. Generate the filter function g with the specific τ value and a window size a.
2. Apply the Gabor transform to the original data in the time domain through multiplying the original data by the filter function g.
3. Transfer the signal to the frequency domain using the fast Fourier transform with the Matlab function fft()[1]. We also need to take the absolute value of the result to deal with complex values.
4. Find the largest frequency at each time point.

Algorithm 1: Gabor transform

Import data of the music clip
for $j = 1 : 0.1 : \tau$ **do**
  $g = exp(-a * (t - j)^2)$
  Yg = multiply data by g
  Ygt = FFT of Yg
  record the largest frequency in Ygt
end for

3.3 Filtering

To filter out frequencies that do not relate to a certain instrument, I will apply the Gaussian filter with center frequencies obtained from the Gabor transform. According to what I get from the Gabor transform, to isolate the bass from Comfortably Numb, if the center frequency is less that 250 Hz, I will use a filter centered at that frequency. If not, I will use a filter centered at 124 Hz. To isolate guitar, if the center frequency is in the range of [300, 800], I will use a filter centered at that frequency. If not, I will use a filter centered at zero. The following Gaussian function:

$$F(k) = \mathrm{e}^{-5(ks - \text{center frequency})^2} \tag{4}$$

General steps of the algorithm:

1. Initialize the Gaussian filter using the center frequency computed from 3.2.
2. Apply the filter to each signal in the frequency domain.

Algorithm 2: Filtering

Continue from algorithm 1
for $j = 1 : 0.1 : \tau$ **do**
  Yg = Gabor transform of the signal at time point j
  if its center frequency is in the range we want **then**
    filter = Gaussian filter with this center frequency
  else
    filter = Gaussian filter with the specified center frequency
  end if
  Yf = filter.$\times$ Yg
end for

4 Computational Results

4.1 Music scores

After apply the Gabor transform on both music clip. The frequencies for the guitar in Sweet Child O' Mine are changing periodically between 277 Hz, 370 Hz, 415 Hz, and 550 Hz. Their associated music scores are $C\sharp_4/Db_4, F\sharp_4/Gb_4, G\sharp_4, C\sharp_5/Db_5$. The frequencies of the bass in Comfortably Numb are mostly around 83 Hz, 87 Hz, 110 Hz, 123Hz, 246 Hz. Their associated music scores are E_2, F_2, A_2, B_2, B_3.

We can visualize the maximum frequency at each time point and the spectrogram showing the relationship between time and frequency. A brighter area signifies the center frequency at its associated time point.

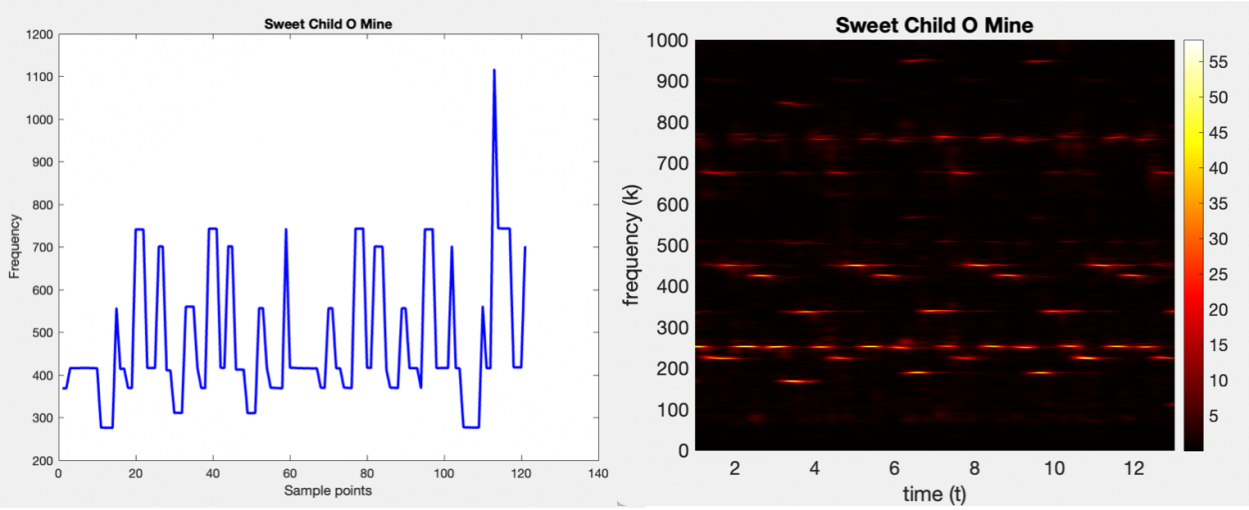

Figure 1: Frequencies of Sweet Child O' Mine at different time

Since the data of Comfortably Numb is too big, I divide the 60-seconds clip into 4 parts. Let's look at the maximum frequency at each time point for all parts and the spectrogram of the first 15 seconds as an example.

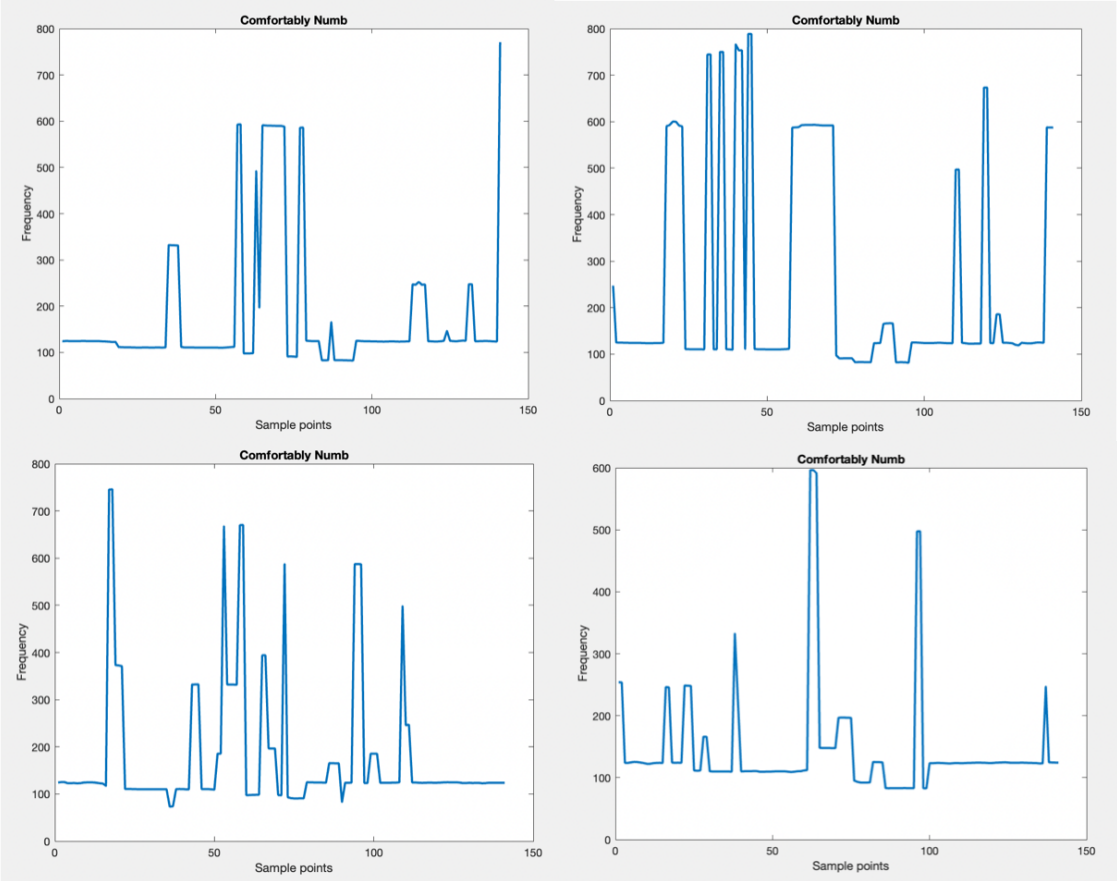

Figure 2: Frequencies of Comfortably Numb at different time

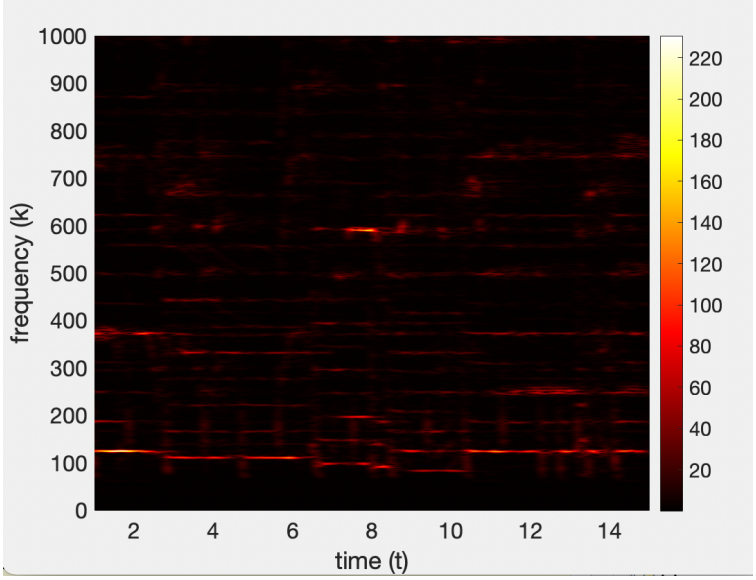

Figure 3: Spectrogram of the first 15 seconds of Comfortably Numb

4.2 Isolation of the bass

Based on the center frequency I compute from the Gabor transform, we can see from the following graph that the filtered signal only keeps frequencies below 300 Hz and filters out the others. In other words, only the bass portion is extracted. Let's visualize what the filtered frequency of the first 15 seconds of Comfortably Numb looks like. The graph on the left shows the comparison between the original frequencies and the filtered frequencies.

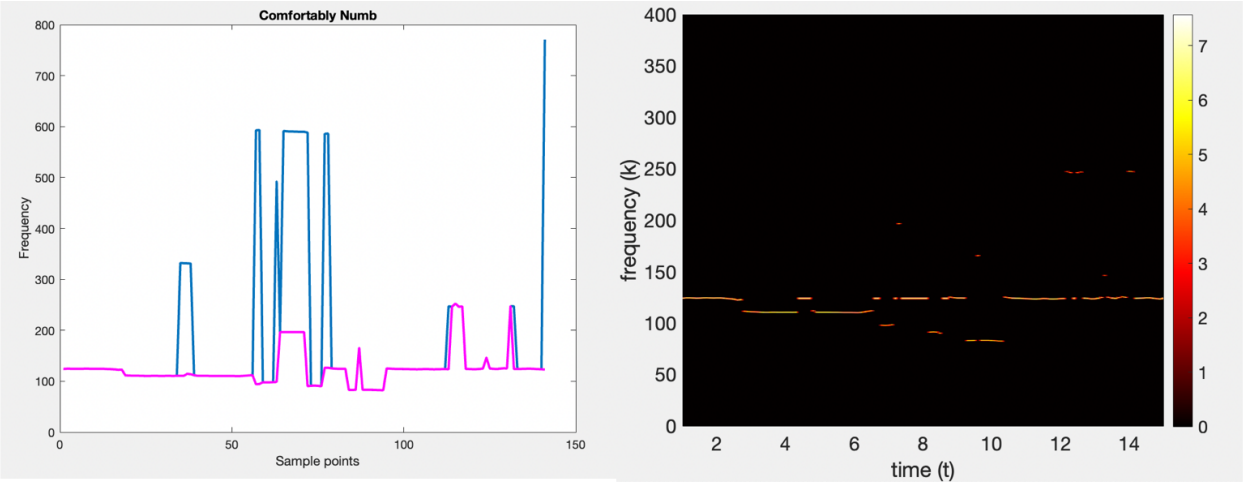

Figure 4: Filtered frequencies of the bass from the first 15 seconds of Comfortably Numb

4.3 Isolation of the guitar

Similar as above, after applying the Guassian filter with center frequencies extracted using the Gabor transform, we can see that the filtered signal only keeps frequencies between 300 Hz and 800 Hz. These filtered frequencies show us the guitar portion of the song.

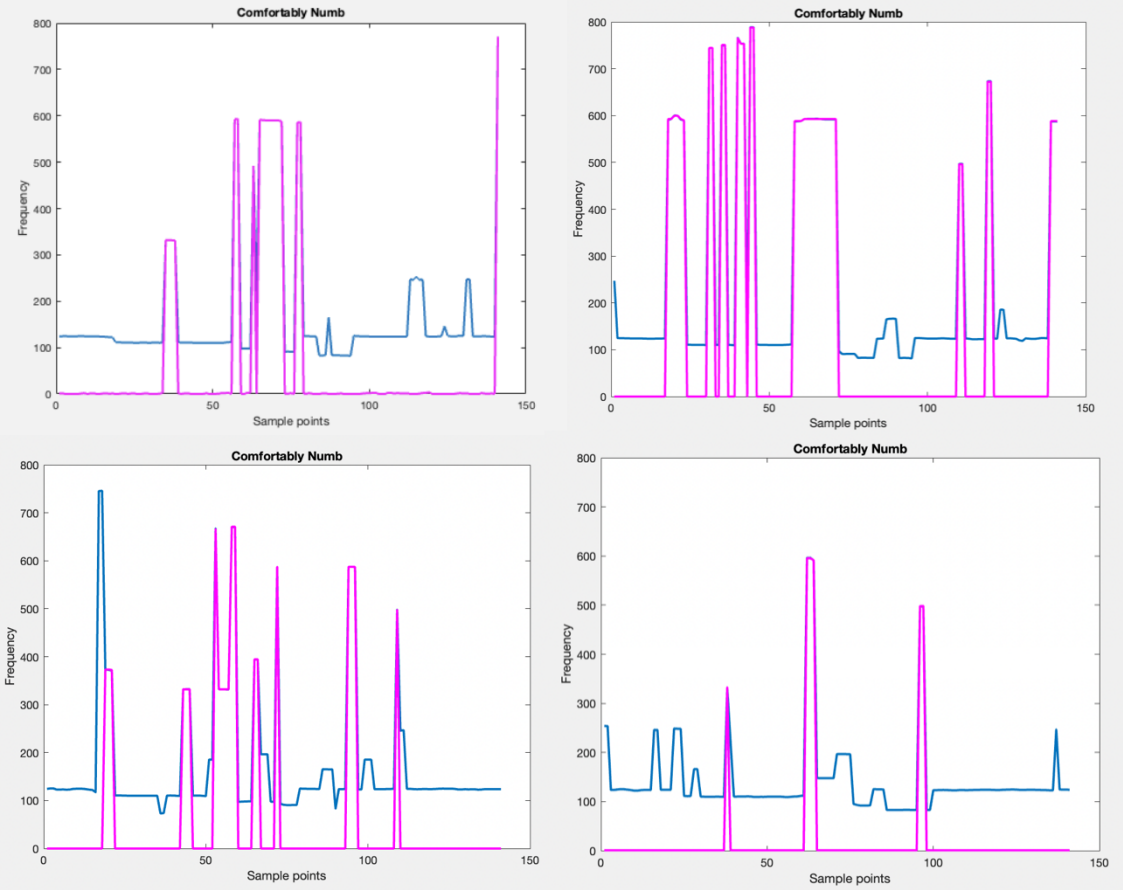

Figure 5: Filtered frequencies compared to the original frequencies in Comfortably Numb

5 Summary and Conclusions

As we can see from the result, applying the Gabor transform solves the issue of losing information about time while obtaining information about frequency. When signals are non-stationary and non-periodic like the music signals we have, using the Gabor transform helps us recover information about a signal in both the time and the frequency domain. It does an amazing job finding the center frequency associated with each specific time by first localizing in the time domain with the τ value. After identifying these center frequency at each time point, we are then able to construct filters to remove unwanted data.

References

[1] *fft()*. URL: `https://www.mathworks.com/help/matlab/ref/fft.html`.

[2] *fftshift()*. URL: `https://www.mathworks.com/help/matlab/ref/fftshift.html`.

[3] Jose Nathan Kutz. *Data-driven modeling & scientific computation: methods for complex systems & big data*. Oxford University Press, 2013.

Appendix A MATLAB Functions

- `[Y, Fs] = audioread(filename)` reads data from the file named filename, and returns sampled data, y, and a sample rate for that data, Fs.
- `t2 = linspace(0,L,n+1)` returns a row vector of `n+1` evenly spaced points between `0` and `L`.
- `fft(Yg)` computes the fast Fourier transform of the matrix `Yg`.
- `fftshift(X)` rearranges a Fourier transform X by shifting the zero-frequency component to the center of the array.
- `ind2sub(size(Ygt),find(Ygt == max(Ygt))` returns the indices of the maximum value in the matrix `Ygt`.

Appendix B MATLAB Code

```
close all; clc; clear all
figure(1)
[Y, Fs] = audioread('GNR.m4a');
trgnr = length(Y)/Fs; % record time in seconds
plot((1:length(Y))/Fs,Y);
xlabel('Time [sec]'); ylabel('Amplitude');
title('Sweet Child O Mine');
p8 = audioplayer(Y,Fs); playblocking(p8);

L = trgnr;
n = length(Y);
t2 = linspace(0,L,n+1); t = t2(1:n);
k = (1/L)*[0:n/2-1 -n/2:-1];
ks = fftshift(k);
y = Y.';

a = 20;
tau = 1:0.1:13;
Ygtspec = [];
Yft = [];
Yfb = [];
freq = [];
for j = 1:length(tau)
    g = exp(-a*(t - tau(j)).^2);
    Yg = g.*y;
    Ygt = abs(fftshift(fft(Yg)));
    Ygtspec(:,j) = Ygt;
    freq_p = ind2sub(size(Ygt),find(Ygt == max(Ygt)));
    freq_max = ks(freq_p(1));
    freq(j)= abs(freq_max);
end
figure(2)
plot(1:length(freq),freq,'b','Linewidth',2)
xlabel('Sample points'); ylabel('Frequency');
title('Sweet Child O Mine');

figure(3)
%pcolor(tau,ks,log(abs(Ygtspec)+1))
pcolor(tau,ks,Ygtspec)
shading interp
set(gca,'ylim',[0,1000],'Fontsize',16)
colormap(hot)
colorbar
xlabel('time (t)'), ylabel('frequency (k)')
title('Sweet Child O Mine');
```

Listing 1: Matlab code for Sweet Child O' Mine

```
close all; clc; clear all
[Y, Fs] = audioread('Floyd.m4a');
trgnr = length(Y)/Fs; % record time in seconds

L = 15;
n = L*Fs;
t2 = linspace(0,L,n+1); t = t2(1:n);
k = (1/L)*[0:n/2-1 -n/2:-1];
ks = fftshift(k);
y = Y.';

a = 50;
tau = 1:0.1:15;
y = y(1:n);
Ygtspec = [];
Yftspec_bass = [];
Yftspec_guitar = [];
freq = [];
freq_bass = [];
freq_guitar = [];
for j = 1:length(tau)
    % extract frequencies at different time
    g = exp(-a*(t - tau(j)).^2);
    Yg = g.*y;
    Ygt = abs(fftshift((fft(Yg))));
    Ygtspec(:,j) = Ygt;
    freq_p = ind2sub(size(Ygt),find(Ygt == max(Ygt)));
    freq_max = ks(freq_p(1));
    freq(j)= abs(freq_max);

    %isolate the bass
    gaussian = exp(-5*(k-124).^2);
    if abs(freq_max) < 250
        gaussian = exp(-5*(k + freq_max).^2);
    end
    Yf_bass = gaussian.*(fft(y));
    Yft_bass = abs(fftshift(Yf_bass));
    Yftspec_bass(:,j) = Yft_bass;
    freq_p_bass = ind2sub(size(Yft_bass),find(Yft_bass == max(Yft_bass)));
    freq_max_bass = ks(freq_p_bass(1));
    freq_bass(j)= abs(freq_max_bass);

    %isolate the guitar
    gaussian = exp(-5*k.^2);
    if abs(freq_max) > 300 && abs(freq_max) < 800
        gaussian = exp(-5*(k + freq_max).^2);
    end
    Yf_guitar = gaussian.*(fft(y));
    Yft_guitar = abs(fftshift(Yf_guitar));
    Yftspec_guitar(:,j) = Yft_guitar;
    freq_p_guitar = ind2sub(size(Yft_guitar),find(Yft_guitar == max(Yft_guitar)));
    freq_max_guitar = ks(freq_p_guitar(1));
    freq_guitar(j)= abs(freq_max_guitar);
end
```

Listing 2: Matlab code for Comfortably Numb