Chingpo Lin
Amath482
HW2

Abstract

There are two song I need to analyze. By using Gabor transform, I can evaluate the time and and frequency domain in a spectrogram, and I can reproduce the music score. With a filter function in frequency domain, I can isolate the bass part in comfortably numb.

Introduction and Overview

Given two music, I would like to analyze them by seeing their music score. To do so, I will put them into a spectrogram to see their relation between time and frequency, and analyze each of the frequency corresponding to music score to form a complete music score. Also, I will do a filter to filter out the bass part in comfortably numb. By knowing the comfortably numb without bass part, I can see how much guitar solo I can put together.

Theoretical Background

Last time, Fourier transform allow us to transform between time and frequency domain. However, it is not clear when we need to see the relation between time and domain. So, we introduce Gabor transform. By introduce variable τ and a, it's clear to see the Fourier transform in a short time. Here is the comparison of FFT and GT:

$$\text{FFT: } f_t(k) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} f(x) e^{-ikx} dx \quad (1)$$

$$\text{GT: } f_g(\tau, k) = \int_{-\infty}^{\infty} f(t)g(t-\tau)e^{-ikt}dt \quad (2)$$

For the GT, it allows us to see the frequency information around time τ. Like what I did in Fourier transform, to actually adapt this formula in the music, I need a discrete form. However, in this form, we cannot simply take arbitrary τ, and a discrete set is needed, which is:

$$k = m\omega_0\,, \quad (3)$$

$$\tau = nt_0 \qquad (4)$$

Then, a discrete Gabor transform can be created:

$$f_g(m, n) = \int_{-\infty}^{\infty} f(t)g(t-nt_0)e^{2\pi im\omega_0 t}dt \quad (5)$$

However, the plot is not good for conveying information. Thus, we need a spectrogram clearly see the time and frequency domain in 2D, which is clear and easy to use. In acquiring this spectrogram, a variable a which is the window size greatly affect the resolution we get.

Algorithm Implementation and Development

First, to see the music score, I want to see the spectrogram of the song. I read the song and get the signal and frequency. I setup the Gaussian transform for each of the tone by times the signal with

$$exp(-a(t - tau(i))^2 \quad (6)$$

Where a is the window size and tau is the center of window. Then we filter it by what we did in Fourier transfer, I get rid of most of the noise in the time vs frequency domain

in spectrogram. For the GNR song, letting a equals 1300 will result in some highlighted in the graph which looks better. By changing the tau value in filter function, I find noise will be best eliminated in $\tau = -0.2/L$. Then, I can get a good Sweet Child O's Mine in guitar range (eliminate all frequency not from 80 to 1200. For the bass range of comfortably numb, I use $\tau = -1/L$ and get the bass range for it (eliminate all frequency not from 60 to 250). And at last, we eliminate the bass part to see the number of guitar solo.

Computational Result

1. The graph of spectrogram for sweet child O' Mine with only guitar range

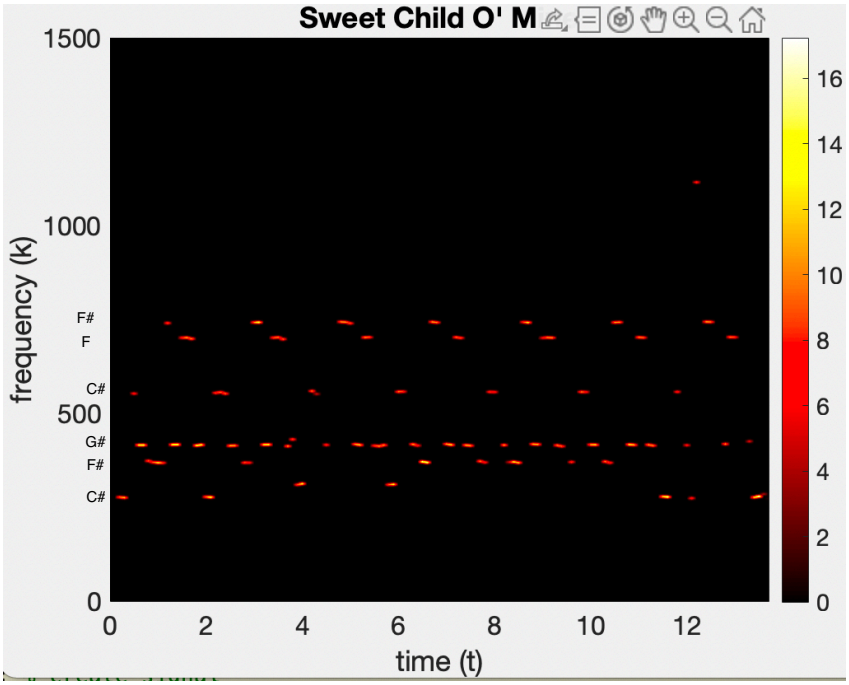

figure (1): spectrogram for GNR with frequency and time

2. The graph of spectrogram for Comfortably Numb with only bass range and without bass range (other range was eliminated)

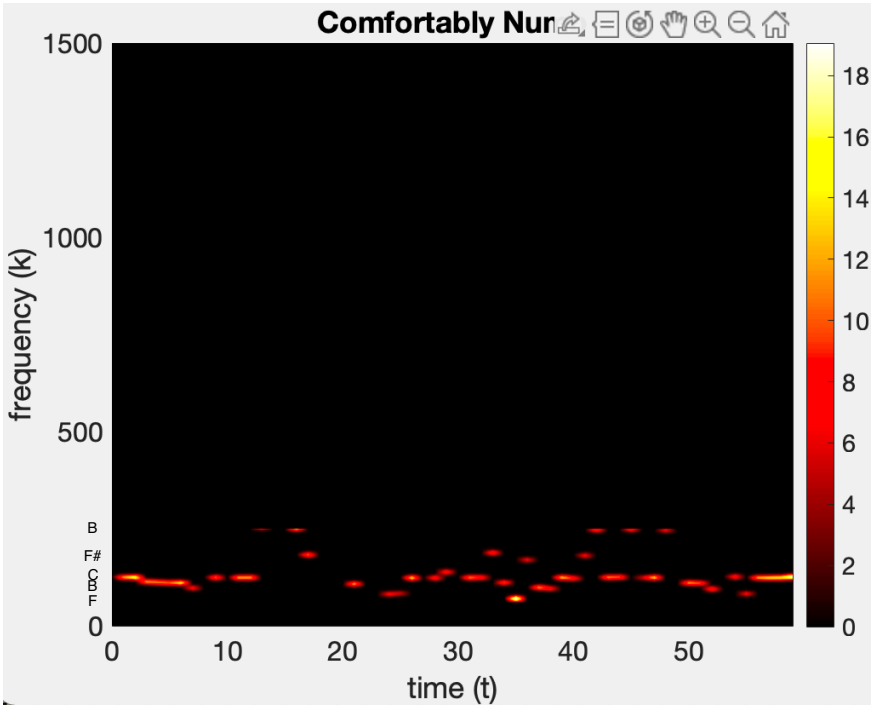

figure (2): spectrogram for Floyd with frequency and time

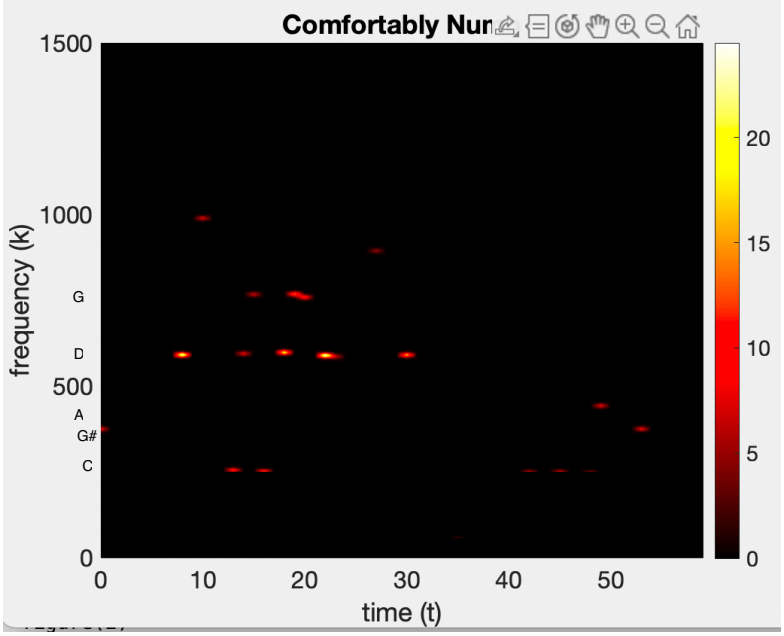

figure (3): the spectrogram for Floyd without bass part

So, the music score for the GNR is: {F# F C# G# F# C#}, and the music score for bass part of Floyd is: {B F# C B F} from **highest to lowest**, and the order is shown in graph. Also, the guitar solo for the part without bass part has music score is: {G,D,A,G#,C}

Summary and Conclusions

In conclusion, by use of Gabor transform, we can successfully connect frequency and time in a spectrogram. And for the song part, we can easily know the music score by see which time we have which frequency with looking at the relation between frequency and music scale. We said Fourier transform is a useful tool to transform between time and frequency domain. Then, the Gabor transform is more like a tool to explore the relationships between both domains.

Appendix A - function in code

audioread: get data from audio file
fftshift: shift the array, and shift zero frequency to the center of array
zeros: create a vector or matrix with all 0 in given dimension
fft: used for Gabor transform, and shift From time to frequency domain
max: picks the maximum value in given input
pcolor: plot the spectrogram with color

Appendix B - Matlab code

```
%% GNR
% Clean workspace
clear all; close all; clc

figure(1)
[y, Fs] = audioread('GNR.m4a');
tr_gnr = length(y)/Fs; % record time in seconds
```

```
t = (1:length(y)) / Fs;
plot(t,y);
xlabel('Time [sec]');
ylabel('Amplitude');
title('Sweet Child O'' Mine');
%p8 = audioplayer(y,Fs); playblocking(p8);

n = length(y);
k = (2*pi/tr_gnr)*[0:(n/2 - 1) -n/2:-1];
ks = fftshift(k);

S = y'; % create signal
tau = 0:0.1:tr_gnr;
a = 1300;
Sgt = zeros(length(y), length(tau));

for i = 1:length(tau)
    g = exp(-a*(t-tau(i)).^2);
    Sf = g.*S;
    Sft = fft(Sf);
    [M,I] = max(abs(Sft(:)));
    ftau = -0.2/length(tau);
    filter = Sft.*exp(ftau*((k-k(I)).^2));
    filter(80 > k./(2*pi) | k./(2*pi) > 1200) = 0; % this filter the guitar
    Sgt(:,i) = fftshift(abs(filter));
end
```

```
figure(2)
pcolor(tau,ks./(2*pi),Sgt)
shading interp
set(gca,'ylim',[0 1500],'Fontsize',16)
colormap(hot)
colorbar
xlabel('time (t)'), ylabel('frequency (k)')
title('Sweet Child O'' Mine');
%% Floyd
% Clean workspace
clear all; close all; clc

figure(1)
[y, Fs] = audioread('Floyd.m4a');
tr_gnr = length(y)/Fs; % record time in seconds
t = (1:length(y)) / Fs;
plot((1:length(y))/Fs,y);
xlabel('Time [sec]');
ylabel('Amplitude');
title('Comfortably Numb');
%p8 = audioplayer(y,Fs); playblocking(p8);

n = length(y);
k = (1/tr_gnr)*[0:(n/2 - 1) -n/2:-1];
ks = fftshift(k);

S = y'; % create signal
```

```
tau = 0:1:tr_gnr;
a = 5300;
Sgt = zeros(length(y) - 1, length(tau));

for i = 1:length(tau)
    g = exp(-a*(t-tau(i)).^2);
    Sf = g.*S;
    Sft = fft(Sf);
    [M,I] = max(abs(Sft(:)));
    s = (k-k(I)).^2;
    ftau = -1/length(tau);
    filter = Sft(1:length(s)).*exp(ftau*((k-k(I)).^2));
    filter(60 > k | k > 250) = 0;
    Sgt(:,i) = fftshift(abs(filter));
end

figure(2)
pcolor(tau,ks,Sgt)
shading interp
set(gca,'ylim',[0 1500],'Fontsize',16)
colormap(hot)
colorbar
xlabel('time (t)'), ylabel('frequency (k)')
title('Comfortably Numb')

%% Floyd Q2
```

```
% Clean workspace
clear all; close all; clc

figure(1)
[y, Fs] = audioread('Floyd.m4a');
tr_gnr = length(y)/Fs; % record time in seconds
t = (1:length(y)) / Fs;
plot((1:length(y))/Fs,y);
xlabel('Time [sec]');
ylabel('Amplitude');
title('Comfortably Numb');
%p8 = audioplayer(y,Fs); playblocking(p8);

n = length(y);
k = (1/tr_gnr)*[0:(n/2 - 1) -n/2:-1];
ks = fftshift(k);

S = y'; % create signal

tau = 0:1:tr_gnr;
a = 5300;
Sgt = zeros(length(y) - 1, length(tau));

for i = 1:length(tau)
    g = exp(-a*(t-tau(i)).^2);
    Sf = g.*S;
    Sft = fft(Sf);
```

```
    [M,I] = max(abs(Sft(:)));
    s = (k-k(I)).^2;
    ftau = -1/length(tau);
    filter = Sft(1:length(s)).*exp(ftau*((k-k(I)).^2));
    filter(k > 60 & k < 250) = 0; % this filter out the bass
    Sgt(:,i) = fftshift(abs(filter));
end

figure(2)
pcolor(tau,ks,Sgt)
shading interp
set(gca,'ylim',[0 1500],'Fontsize',16)
colormap(hot)
colorbar
xlabel('time (t)'), ylabel('frequency (k)')
title('Comfortably Numb')
```