AMATH 482 HW2: Rock & Roll and the Gabor Transform

Tiffany Tang
Department of Applied Mathematics
University of Washington
leqitang@uw.edu

Abstract

In this paper, the Gabor transform is used as a filter function to obtain frequencies of music recordings in order to generate the original music notes of the music. With frequency information of the music acquired after the Gabor transform, the data of frequencies at each time point throughout two music clips: Sweet Child O' Mine and Comfortably Numb are visualized with spectrograms. The guitar part of Comfortably Numb is also acquired by filtering out the low- frequency part of the music signal piece.

1 Introduction and Overview

In order to process signal data, the Fourier transform is commonly used and known as the basis of most signal process. However, the Fourier transform has a huge limitation that it could not retain any information from the time domain. The frequencies of a signal could be obtained by this transformation, but the time when they occur could not been showed. Therefore, the Gabor transform has been created to successfully figure out the limitation of the Fourier transform. The Gabor transform could capture both time and frequency information simultaneously. The Gabor transform modifies the original Fourier transform by sliding the local center of filter over the time period and filtering for high amplitude frequencies. By tuning the width of the Gabor window and the translation of the window, we investigated the frequency of music notes in the paper.

2 Theoretical Background

We have seen the Fourier transform that decomposes a time defined function into its frequency components, and is defined by

$$F(k) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} e^{-ikx} f(x) dx$$

And its inverse function is defined by

$$f(k) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} e^{ikx} F(x) dx$$

From the definition, we set that the transform is over the area $x \in [-\infty, \infty]$. However, since the Fourier transform eliminate time-domain information with integral, the Gabor transform is used to slide windows of filter in a time period. The Gabor transform performed as a filter function $g(t-\tau)$ that shift by τ. The filter centered at each τ. The Gabor transform, also known as the short-time Fourier transform (STFT), is given by

$$\widetilde{f_g}(\tau,k)=\int_{-\infty}^{\infty} f(t)g(t-\tau)e^{-kt}dt$$

For a fixed τ the function $\widetilde{f_g}(\tau,k)$ gives the information about the frequency components near time τ.
The inverse of the Gabor transform is given by

$$f(t)=\frac{1}{2\pi\|g\|_2}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\widetilde{f_g}(\tau,k)g(t-\tau)e^{-kt}dt$$

In the above function, we have $\|g\|_2=(\int_{-\infty}^{\infty}|g(t)|^2dt)^{\frac{1}{2}}=1$, which the L_2-norm of g represented as the total energy of a function is set to unity.
To use the Gabor transform in MATLAB, the discrete Gabor transform is used. In this condition, a discrete set of frequencies are required:

$$k=m\omega_0,\tau=nt_0$$

Where m and n are integers and ω_0 *and* t_0 are positive constants (frequency resolutions). The discrete Gabor transform is:

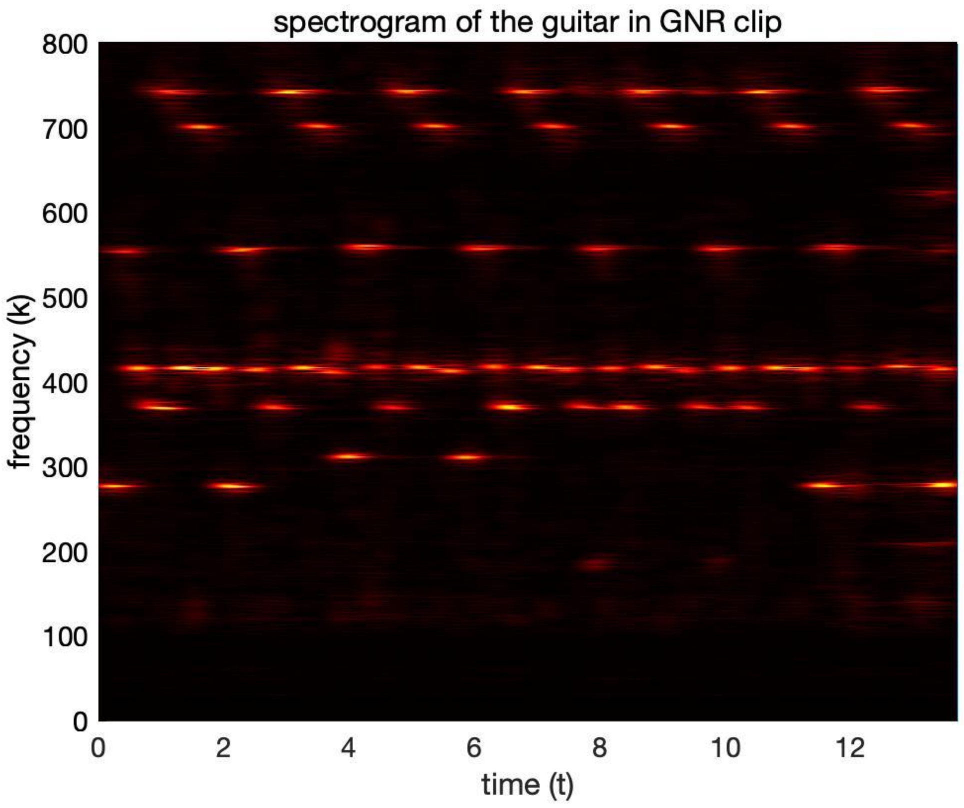
$$\widetilde{f_g}(m,n)=\int_{-\infty}^{\infty} f(t)g(t-nt_0)e^{2\pi im\omega_0 t}dt$$

3 Algorithm Implementation and Development

Two parts are investigated in the paper. One is finding the music score in two music clip. For the bass in the Floyd clip, I filtered this piece around the bass frequency. The other part is to find the guitar solo out of the Floyd clip.
In order to find the music score for the guitar in the GNR clip, we first iterate through all the τ to generate Gabor filter as window function and apply it to signal data. The Gabor filter function is $e^{(-a*(t-\tau))^2}$, where a is the window width and τ is the center of the filter. Then I applied the 3-D Fourier transform (fftn function) to transform the data from spatial domain to frequency domain and rearrange the data by shifting the zero-frequency component to the center of the array (fftshift function). Lastly, the spectrogram of the frequency per time is plotted. The log of the spectrogram was also plotted.
To extracting the bass in the Floyd clip, I filtered the music piece around the bass frequency: 60 to 260 Hertz(`s_filter = s_fft.*fftshift(abs(t)>260)`). And through iteration for each τ to get frequencies across the time period. Then I applied the same step as I used in the guitar in the GNR clip to get the frequency per time. The guitar part of the Floyd's music clip is obtained by filtering out the frequency under 260 Hertz`(fftshift(abs(t)>260))`.

4 Computational Results

With the knowledge about the different frequencies in Hertz for each music note, the music notes played in Sweet Child O' Mine: #C, #F, #G, #C, F, and #F. The last three notes are from higher octaves.



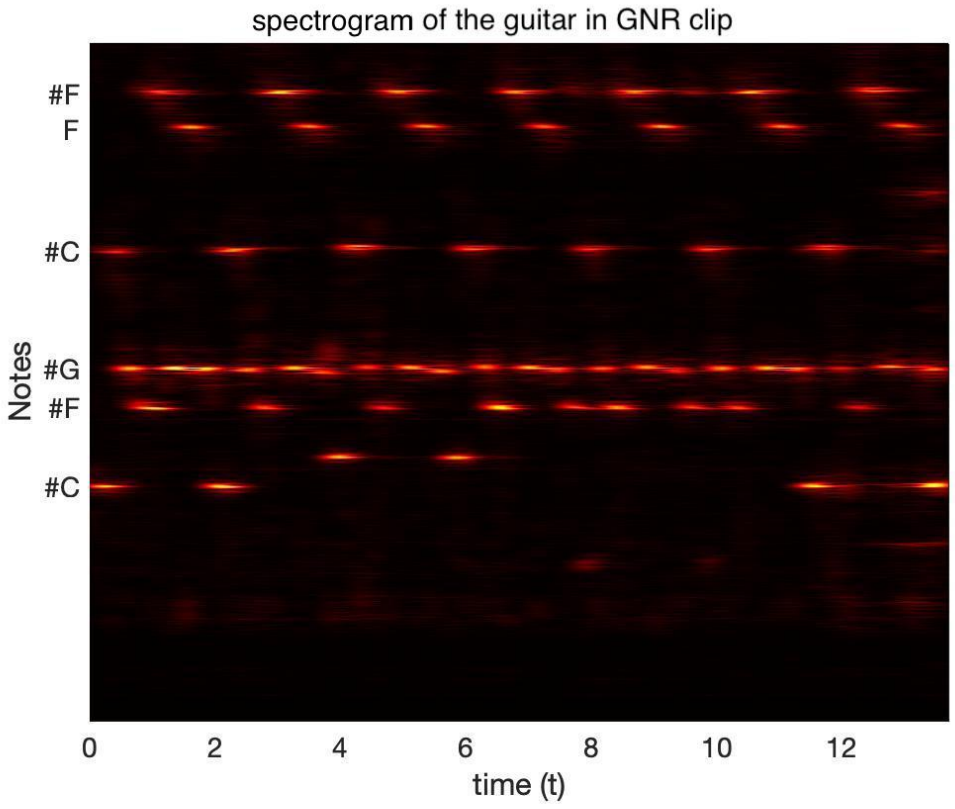


Figure 1 and 2: These two figures present the spectrogram of the guitar in the GNR clip with frequency and notes labeled respectively.

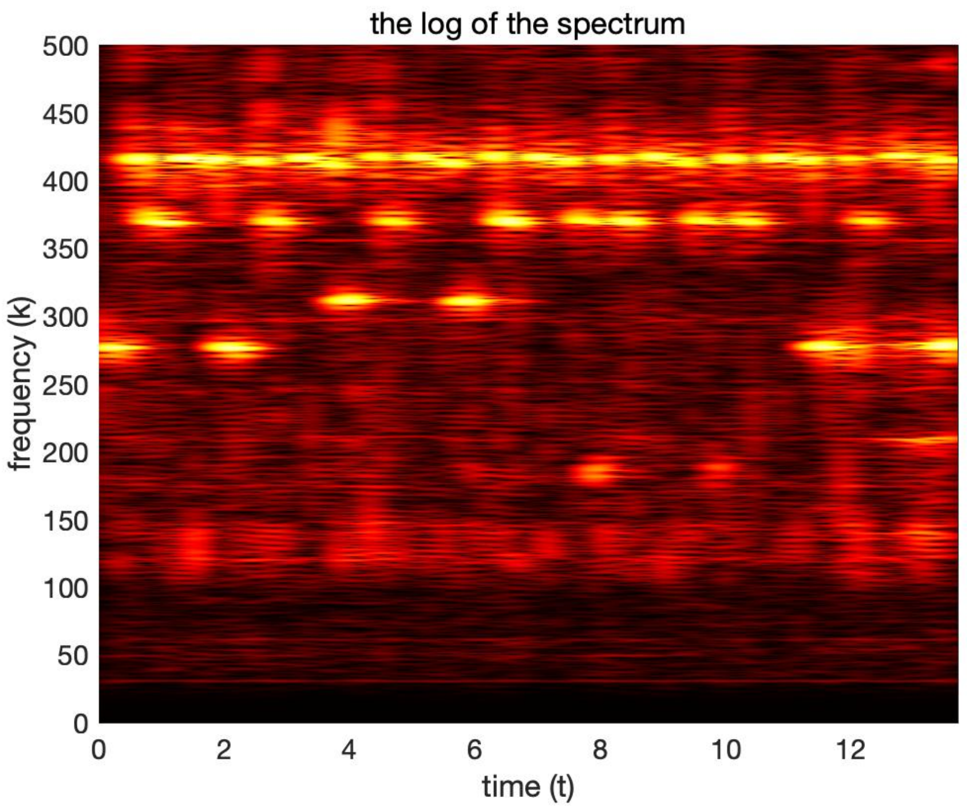


Figure 3: This figure presents the log of spectrogram for the guitar in the GNR clip.

After filtering the music piece around the bass frequency: 60 to 260 Hertz. I obtained the music note at: #D, #G , C, #F, and B for bass part of Comfortably Numb. Some notes showed dimly are the noise that have not been filtered out.

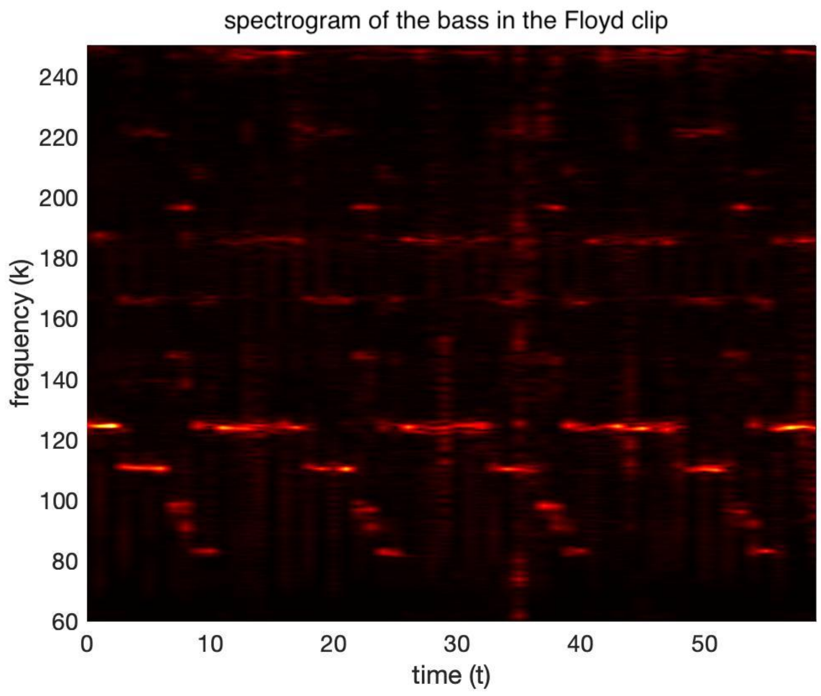


Figure 4: presents the spectrogram of guitar in the GNR clip with frequency labeled.

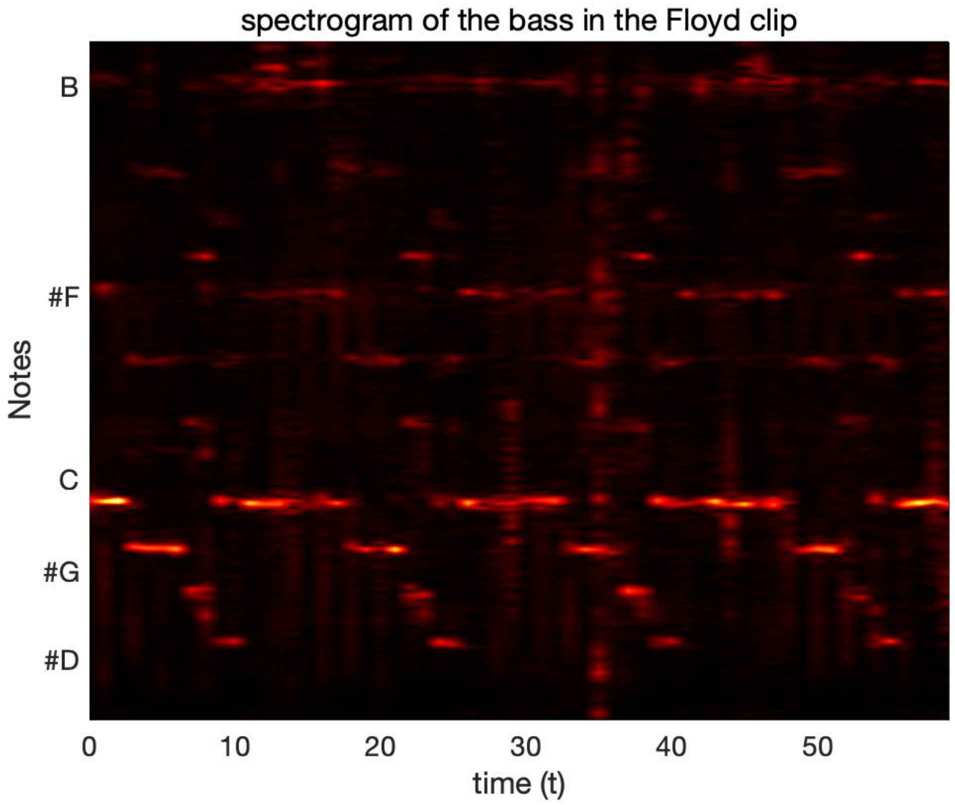


Figure 5: presents the spectrogram of the guitar in the GNR clip (comfortably numb) with music notes labeled.

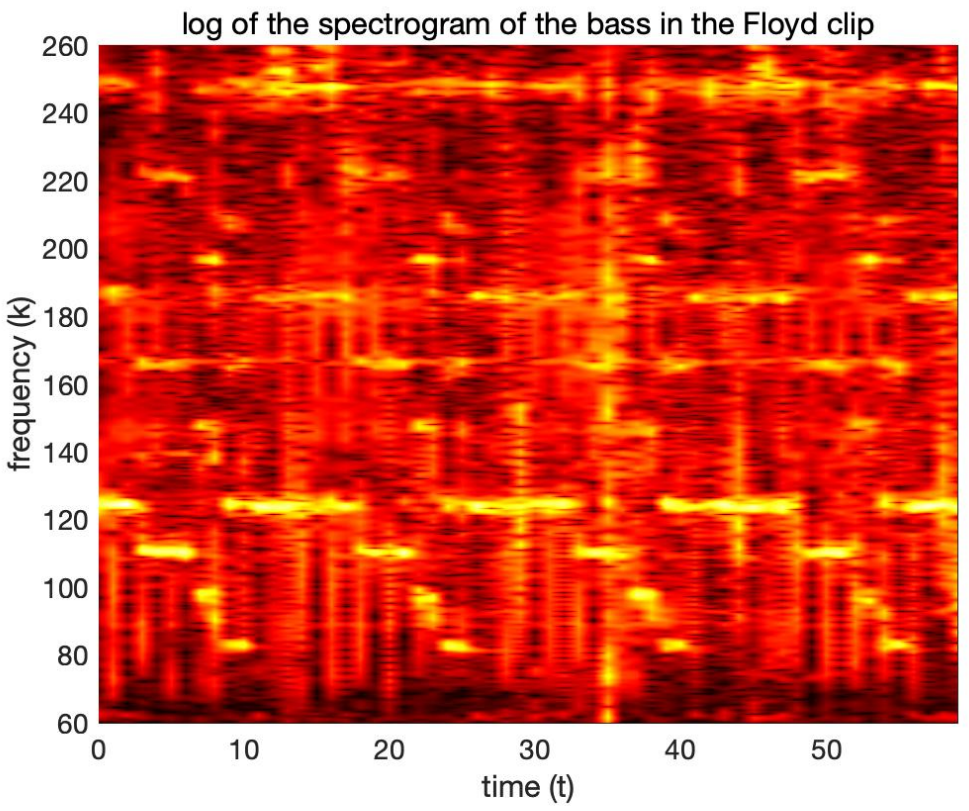


Figure 6: this graph presents the log of spectrogram for the guitar in the GNR clip.

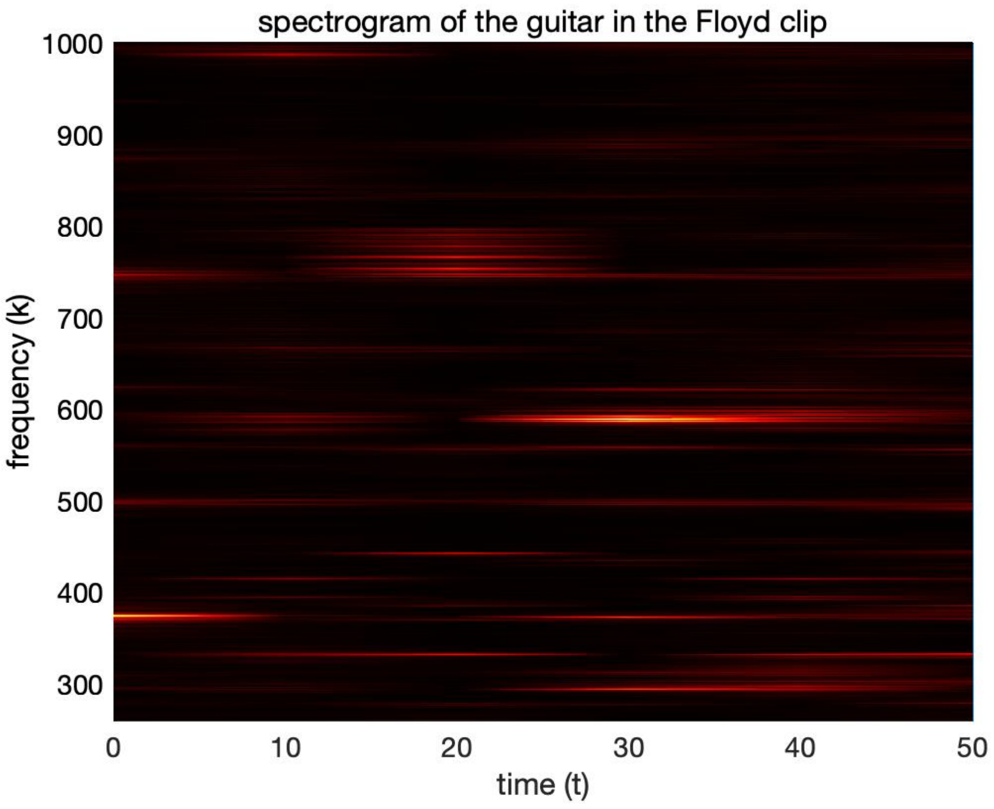


Figure 7: the spectrogram of the guitar in Comfortably Numb.

After filtering the music piece around the guitar frequency: > 260 Hertz. I obtained the music note at: #F, B, #D, A, and B for the guitar part of Comfortably Numb.

5 Summary and Conclusions

The Gabor transform is a practical tool that could be used to find the frequencies of signal data in time period. It provides information not only in the frequency domain but also in the time domain. The frequencies of two music piece: the guitar of Sweet Child O' Mine produced by the Gun and Roses and the bass of Comfortably Numb by Pink Floyd were obtained by filtering through Gabor transform. The guitar part of the Comfortably Numb was also obtained by filtering out the low frequency of the song. However, there are still limitations presented. Some noise is still showed in the spectrogram of the Comfortably Numb. The current Gabor transform used might not be the most suitable filter. Therefore, more filters are needed to be investigated to have a clearer spectrogram of the music clip. The broader application of the Gabor transform in various fields are still researched for the future usage.

Appendix A. MATLAB Functions

1. Y = fftn(X) returns the multidimensional Fourier transform of an N-D array using a fast Fourier transform algorithm
2. Y = ifftn(X) returns the multidimensional discrete inverse Fourier transform of an N-D array using a fast Fourier transform algorithm
3. Y = fftshift(X) rearranges a Fourier transform X by shifting the zero-frequency component to the center of the array
4. y = linespace(x1, x2, n) returns a row vector of n evenly spaced points between x1 and x2.
5. pcolor(X,Y,C) specifies the *x*- and *y*-coordinates for the vertices. The size of C must match the size of the *x*-*y* coordinate grid.

Appendix B. Matlab Code

```
%% guitar for GNR
clear all;clc; close all;
[y, Fs] = audioread('GNR.m4a');
trgnr = length(y)/Fs; % record time in
seconds4plot((1:length(y))/Fs,y);

s=y'; % signal
L = trgnr;
n=length(s); % length of signal
t2 = linspace(0, L, n+1);
t=t2(1:n);
k =(1/L)*[0:n/2-1 -n/2:-1]; % 1/L instead of 2*pi/L
ks=fftshift(k);



tau = 0:0.1:L;
a= 15;
%sgt_spec = zeros(n,length(tau))
for i = 1:length(tau)
   g = exp(-a*(t-tau(i)).^2); % window function
   sg = g.* s;
   sgt = fft(sg);
   sgt_spec(:, i) = fftshift(abs(sgt));
end
figure(1)
pcolor(tau, ks, sgt_spec)
shading interp
%ylim([0 400])

set(gca, 'ylim', [0 800],'FontSize', 14)

xlabel('time (t)'), ylabel('frequency (k)'),title('spectrogram of the
guitar in GNR clip')

ylabel('frequency (k)')
colormap(hot)
%%
figure(2)
pcolor(tau, ks, sgt_spec)
shading interp
xlabel('time (t)')
set(gca, 'ylim', [0 800],'FontSize', 14)
```

```
yticks([277.18, 369.99, 415.30, 554.37, 698.46, 739.99])
yticklabels(["#C", "#F", "#G", "#C", "F", "#F"])
ylabel('Notes')
colormap(hot)

%% log of the spectrum for GNR clip
figure(3)
for i = 1:length(tau)
    g = exp(-a*(t-tau(i)).^2); % window function
    sg = g.* s;
    sgt = fft(sg);
    sgt_spec(:, i) = log(fftshift(abs(sgt))+1);
end

pcolor(tau, ks, sgt_spec)
shading interp
%ylim([0 400])

set(gca, 'ylim', [0 800],'FontSize', 14)
xlabel('time (t)'), ylabel('frequency (k)'), title('the log of the
spectrum')

ylabel('frequency (k)')
colormap(hot)

%% isolate the bass of the Floyd clip
clear all;clc; close all;
[y, Fs] = audioread('Floyd.m4a');
trgnr = length(y)/Fs; % record time in
seconds4plot((1:length(y))/Fs,y);


s=y'; % signal
L = trgnr;
n=length(s); % length of signal
t2 = linspace(0, L, n+1);
t=t2(1:n);
k =(1/L)*[0:n/2-1 -n/2:-1]; % 1/L instead of 2*pi/L
ks=fftshift(k);

s_fft = fft(s);
s_filter = s_fft.*fftshift(60<abs(t)<260);
s_bass = ifft(s_filter);
```

```matlab
tau = 0:1:L;
a= 10;
for i = 1:length(tau)
    g = exp(-a*(t-tau(i)).^2); % window function
    sg = g.* s_bass;
    sgt = fft(sg);
    sgt_spec(:, i) = fftshift(abs(sgt));
end
sgt_spec(end,:) = [];
figure(1)
pcolor(tau, ks,sgt_spec)
shading interp
set(gca, 'ylim',[60 260], 'FontSize', 14)

xlabel('time (t)')
ylabel('frequency (k)')
title('spectrogram of the bass in the Floyd clip')
colormap(hot)
%% notes of the frequency
figure(2)
pcolor(tau, ks,sgt_spec)
shading interp
colormap(hot)
set(gca, 'ylim',[60 260], 'FontSize', 14)
title('spectrogram of the bass in the Floyd clip')
xlabel('time (t)')
yticks([77.762, 103.83, 130.81, 185.00, 246.94])
yticklabels(["#D", "#G", "C", "#F", "B"])
ylabel('Notes')

%% log of the spectrogram for Floyd clip
clc; clear all;close all;
figure(3)
clear all;clc; close all;
[y, Fs] = audioread('Floyd.m4a');
trgnr = length(y)/Fs; % record time in
seconds4plot((1:length(y))/Fs,y);


s=y'; % signal
L = trgnr;
n=length(s); % length of signal
t2 = linspace(0, L, n+1);
t=t2(1:n);
```

```
k =(1/L)*[0:n/2-1 -n/2:-1]; % 1/L instead of 2*pi/L
ks=fftshift(k);

s_fft = fft(s);
s_filter = s_fft.*fftshift(60<abs(t)<260);
s_bass = ifft(s_filter);

tau = 0:1:L;
a= 10;

%sgt_spec = zeros(2635921, 60)
for i = 1:length(tau)
    g = exp(-a*(t-tau(i)).^2); % window function
    sg = g.* s_bass;
    sgt = fft(sg);
    sgt_spec(:, i) = log(fftshift(abs(sgt))+1);
end
sgt_spec(end,:) = [];
figure(1)
pcolor(tau, ks,sgt_spec)
shading interp
set(gca, 'ylim',[60 260], 'FontSize', 14)

xlabel('time (t)')
ylabel('frequency (k)')
title('log of the spectrogram of the bass in the Floyd clip')
colormap(hot)
%% part 3 guitar of the spectrogram in the CN clip
figure(3)
s=y'; % signal
L = trgnr;
n=length(s); % length of signal
t2 = linspace(0, L, n+1);
t=t2(1:n);
k =(1/L)*[0:n/2-1 -n/2:-1]; % 1/L instead of 2*pi/L
ks=fftshift(k);

s_fft = fft(s);
s_filter = s_fft.*fftshift(abs(t)>260);
s_bass = ifft(s_filter);

sgt_spec= [];
a= 10;
tau = 0:10:L;
```

```matlab
for i = 1:length(tau)
   g = exp(-a*(t-tau(i)).^2); % window function
   sg = g.* s;
   sgt = fft(sg);
   sgt_spec(1:2635921, i) = fftshift(abs(sgt));
end
sgt_spec(end,:) = [];
pcolor(tau, ks, sgt_spec)
shading interp
%ylim([0 400])
set(gca, 'ylim',[260 1000], 'FontSize', 14)
%yyaxis left
xlabel('time (t)')
ylabel('frequency (k)')
title('spectrogram of the guitar in the Floyd clip')
colormap(hot)

yyaxis right
yticks([369.99, 493.88, 622.25, 880.00, 987.77])
yticklables(["#F", "B", "#D", "A", "B"])
ylabel('Notes')
```