AMATH482
HW2 Rock & Roll and the Gabor Transform
Dean Wang

Abstract
Given two pieces of music clips, I need to use Gabor transform and a series of method to print the sound on the spectrogram. Then, I try to isolate the Comfortably Number to see how many solos could put together.

Introduction and Overview
Even though we know what is going on in the music, but it's difficult to tell the frequency. In order to build the relationship between the frequency and the time, we need to use Gabor transform and multiplying the filter round to find the correlation between time and frequency.

In order to achieve the goal, we first denoise the OVERTONES. Then, we use gaussian window scale to whole script and receive the relationship between each time window and frequency information. Finally, we output the sound in the spectrogram which also contains the note in the stave. For the step of Floyd, we add one more step that is to add a lower frequency wave to separate the base.

Theoretical Background
When we define a time defined function into frequency function, we use Fourier transform. The function are:

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

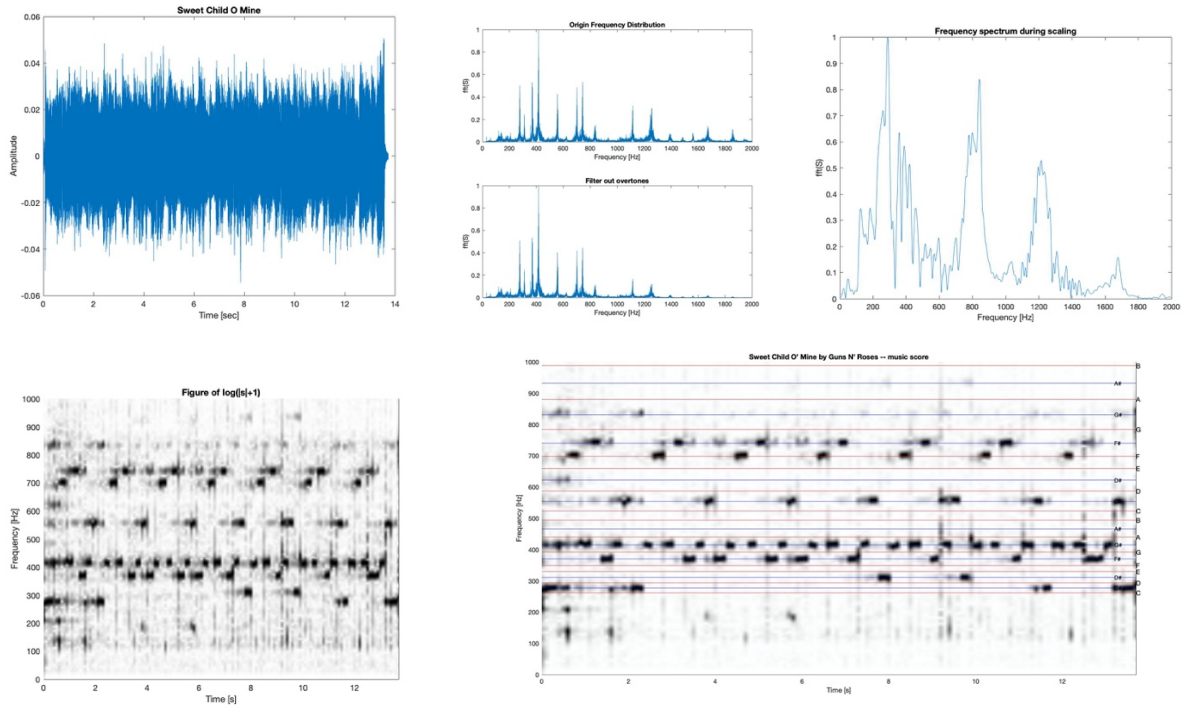
$$f(k) = \frac{1}{2\pi}\int_{-\infty}^{\infty} e^{-ikxF(x)dx} \quad (2)$$

However, this method is very difficult to visualize the data in the time scale. We will use Gabor transform to solve the problem. The Gabor function is :

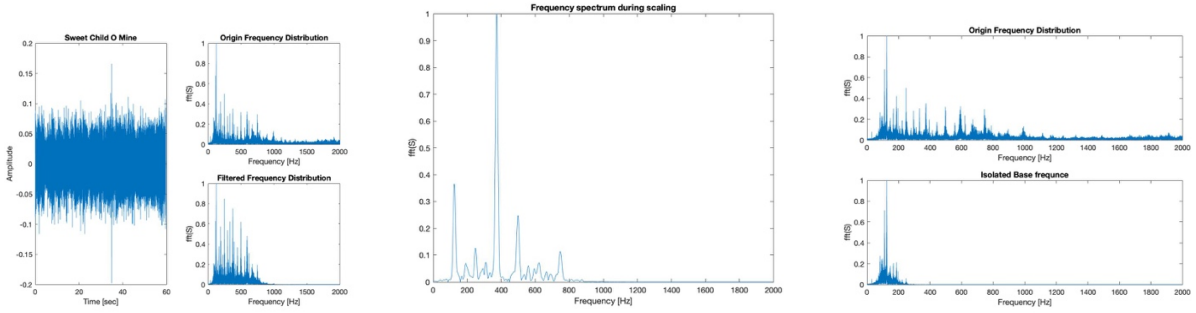
$$f_g(m,n) = \int_{-\infty}^{\infty} f(t)g(n-nt_0)e^{2\pi imw_0 t}dt \quad (3)$$

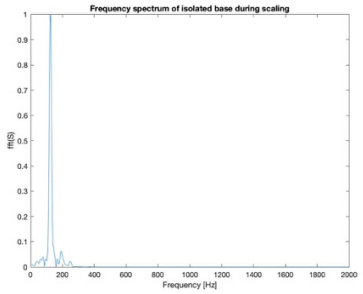
Since we cannot shift the window by any arbitrary τ- it can only be taken in some discrete set of values. Thus, we discrete the set of frequency $k = mw_0$ and $\tau = nt_0$. Both m and n are positive constant.

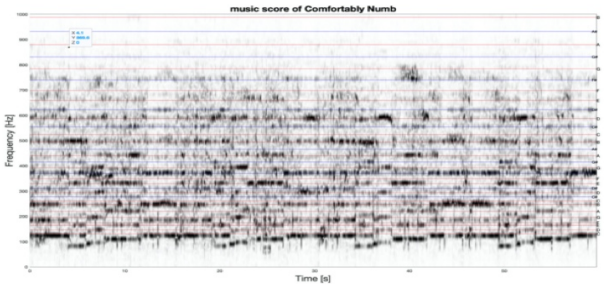
Algorithm Implementation and Development
To be specific in the GNR, we sett signal, frequency domain in same array size, and fft signal plot spectrum in signal domain (Hz). Then, use a gaussian filter to filter out overtones from the instruction, instruments' frequency is around middle C, so we set the filter's center frequency as 300Hz. use gaussian window function after comparing different setting of parameter a, which has a relative promising result in both time and frequency resolution. Finally draw the log(|s|+1) graph

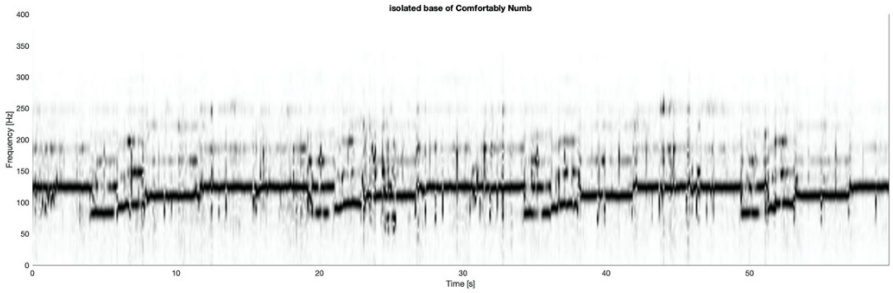


For Flyod, we decrease the resolution of original clip to decrease the data size or it will encounter insufficient processing memory. Then, we set the signal of frequency domain in same array size and plot spectrum in signal domain (Hz) record time in seconds. We use gaussian window function to scale over the song to get time-related frequency domain. After comparing different setting of parameter a, value 2000 can have a relative promising result in both time and frequency resolution. The next step is to plot the spectrogram and add lines of tones figure to find guitar solo and truncate the low frequency. Later on, we truncate the frequency domain in half to save plotting time and draw line of frequencies in staff system. In order to filter around low frequency to isolate the base, we set signal of frequency domain in same array size and plot spectrum in signal domain (Hz). After comparing different setting of parameter a, we value 2000 can have a relative promising result in both time and frequency resolution.



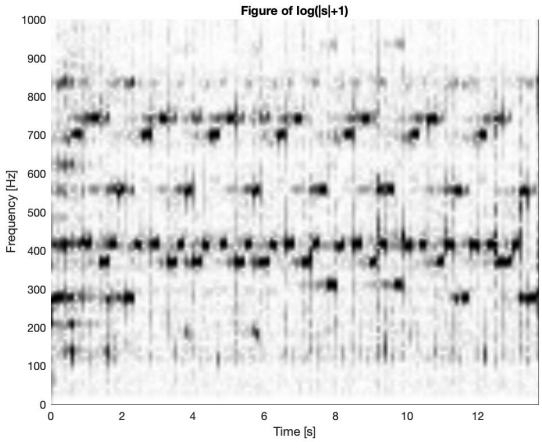


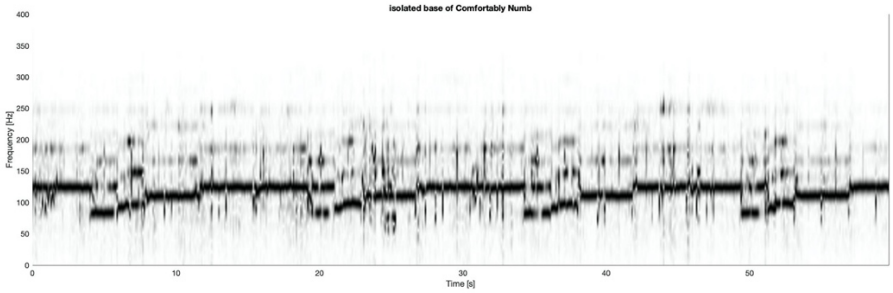




Computational Result

Through the use of the Gabor filtering we used in class, reproduce the music score for the guitar in the GNR clip, and the bass in the Floyd clip. Both are clearly identifiable.





Summary and Conclusions

We finish the transform from the sound to the visualization of frequency through time. In the process, we use gaussian window to scale the clips and Gabor transformation. We finally print the spectrogram and change the filter to view the effect of it.

Appendix

HW2-GNR

```
close all
clear all
figure(1)
[y, Fs] = audioread('GNR.m4a');
% [y, Fs] = audioread('Floyd.m4a');
tr_real = 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);
%%

n=length(y);
L=tr_real;
t2 = linspace(0,L,n+1);

k = (1/L)*[0:n/2-1 -n/2:-1];

t = t2(1:length(k));
S=y(1:length(k));

ks = fftshift(k);
St=fft(S);
figure(2)
subplot(2,1,1)
plot(ks,abs(fftshift(St))/abs(max(St)))
xlim([0,2000])
xlabel('Frequency [Hz]');
ylabel('fft(S)')
title('Origin Frequency Distribution')
%setting signal, frequency domain in same array size, and fft signal
%plot spectrum in signal domain(Hz)

tau=1e-6;
k0=300;
filter = exp(-tau*(k - k0).^2);
St=St'.*filter;
S=ifft(St);
%use a gaussian filter to fliter out overtones
%from the instruction, instruments' frequency is around middle C,
%so we set the filter's center frequency as 300Hz.
subplot(2,1,2)
plot(ks,abs(fftshift(St))/abs(max(St)))
xlim([0,2000])
xlabel('Frequency [Hz]');
ylabel('fft(S)')
```

```
title('Filter out overtones')

%%
%use gaussian window function
a=1500;%after comparing different setting of parameter a,
%this value can have a relative promissing result in both time and frequency
resolution
tau=0:0.1:tr_real;
i=1;
for j=1:length(tau)

    g = exp(-a*(t - tau(j)).^2);
    sg=g.*S;
    sgt=fft(sg);
    sgt_spec(:,j)= fftshift(abs(sgt));
    i=i+1;
    disp(['scaling',num2str(j),'/',num2str(length(tau))])
    figure(3)
    sgtf=fftshift(sgt);
    plot(ks,abs(sgtf)/max(abs(sgtf)))
    xlim([0,2000])
    xlabel('Frequency [Hz]');
    ylabel('fft(S)')
    title('Frequency spectrum during scaling')
    pause(0.01)


end


%%
figure( 'Position', [10 10 1200 600])

spec=1-(abs(sgt_spec)./max(abs(sgt_spec)));
pcolor(tau,ks(1,0.5*length(ks):length(ks)),spec(0.5*length(ks):length(ks),:))
%truncate the frequency domain in half to save plotting time
ylim([0,1000])
shading interp
colormap(gray)
hold on

%draw line of frequencies in staff system
ctone=[261.63 293.66 329.63 349.23 392.00 440.00 493.88 523.55 587 659 698
784 880 988];
%frequency of tones from C1
tonename=['C','D','E','F','G','A','B','C','D','E','F','G','A','B','C','D','E'
];
for i=1:length(ctone)
    plot([tau(1),tau(end)],[ctone(i),ctone(i)],'-r')
    text(tau(end),ctone(i),tonename(i),'FontSize',10)
    hold on
end
semitone=[277 311 370 415 466 554 622 740 831 932];
%frequency of semitones C1#
tonename2={'C#','D#','F#','G#','A#','C#','D#','F#','G#','A#','C#','D#','F#','
G#','A#'};
for i=1:length(semitone)
```

```
    plot([tau(1),tau(end)],[semitone(i),semitone(i)],'-b')
    text(tau(end-5),semitone(i),string(tonename2(i)),'FontSize',9)
    hold on
end
title('Sweet Child O,Äô Mine by Guns N,Äô Roses -- music score')
ylabel('Frequency [Hz]');
xlabel('Time [s]')
%%
figure(5)
spec=log(abs(sgt_spec)+1);
spec=1-spec./max(spec);
pcolor(tau,ks(1,0.5*length(ks):length(ks)),spec(0.5*length(ks):length(ks),:))
shading interp
ylim([0,1000])
colormap(gray)
title('Figure of log(|s|+1)')
ylabel('Frequency [Hz]');
xlabel('Time [s]')
%as the hw requirement,plot log(|s|+1)
%%
disp(['please wait for the plot of spectrogram'])
disp(['finish'])
```

HW2-Floyd

```
close all
clear all
figure(1)
% [y, Fs] = audioread('GNR.m4a');
[y, Fs] = audioread('Floyd.m4a');

% y=y(1:0.2*length(y));%truncate the clip for test convinience
subplot(2,2,[1 3])
plot((1:length(y))/Fs,y);
xlabel('Time [sec]'); ylabel('Amplitude');
title('Sweet Child O Mine');
% p8 = audioplayer(y,Fs); playblocking(p8);
%%
%decrease the resolution of original clip to decrease the data size
%or it will encounter insufficient processing memory
samplerate=10;
sample=1:samplerate:length(y);
y=y(sample);
Fs=Fs/samplerate;

%%
%this part setsignal, frequency domain in same array size, and fft signal
%plot spectrum in signal domain(Hz)
tr_real = length(y)/Fs; % record time in seconds
n=length(y);
L=tr_real;
t2 = linspace(0,L,n+1);
k = (1/L)*[0:n/2-1 -n/2:-1];
t = t2(1:length(k));
S=y(1:length(k));
```

```
ks = fftshift(k);
St=fft(S);
subplot(2,2,2)

plot(ks,abs(fftshift(St))/abs(max(St)))
xlim([0,2000])
xlabel('Frequency [Hz]');
ylabel('fft(S)')
title('Origin Frequency Distribution')
%%
%this part filter the overtones
tau=1e-5;
k0=400;
filter = exp(-tau*(k - k0).^2);
St=St'.*filter;
S=ifft(St);
%use a gaussian filter to fliter out overtones
%from the instruction, instruments' frequency is around middle C,
%so we set the filter's center frequency as 300Hz.
subplot(2,2,4)
plot(ks,abs(fftshift(St))/abs(max(St)))
xlim([0,2000])
xlabel('Frequency [Hz]');
ylabel('fft(S)')
title('Filtered Frequency Distribution')

%%
%use gaussian window function to scale over the song
%to get time-related frequency domain
a=1000;%after comparing different setting of parameter a,
%value 2000 can have a relative promissing result in both time and frequency
resolution
tau=0:0.1:tr_real;
i=1;
for j=1:length(tau)

    g = exp(-a*(t - tau(j)).^2);
    sg=g.*S;
    sgt=fft(sg);
    sgt_spec(:,j)= fftshift(abs(sgt));
    i=i+1;
    disp(['scaling',num2str(j),'/',num2str(length(tau))])
    figure(2)
    sgtf=fftshift(sgt);
    plot(ks,abs(sgtf)/max(abs(sgtf)))
    xlim([0,2000])

    xlabel('Frequency [Hz]');
    ylabel('fft(S)')
    title('Frequency spectrum during scaling')
    pause(0.001)
end


%%
%this part plot the spectrogram and add lines of tones
figure('Position', [10 10 2000 800])
```

```
% index=find(200<ks<1000);%to find guitar solo,truncate the low frequency
% sgt_spec=sgt_spec()
sgt_spec=log(abs(sgt_spec)+1);
spec=1-(abs(sgt_spec)./max(abs(sgt_spec)));
pcolor(tau,ks(1,0.5*length(ks):length(ks)),spec(0.5*length(ks):length(ks),:))

%truncate the frequency domain in half to save plotting time
ylim([0,1000])
shading interp
colormap(gray)
hold on

%draw line of frequencies in staff system
ctone=[130.8 146.8 164.8 174.8 196.0 220.0 246.9 261.63 293.66 329.63 349.23
392.00 440.00 493.88 523.55 587 659 698 784 880 988];
%frequency of tones from C1
tonename=['C','D','E','F','G','A','B','C','D','E','F','G','A','B','C','D','E'
,'F','G','A','B','C','D','E','F','G'];
for i=1:length(ctone)
    plot([tau(1),tau(end)],[ctone(i),ctone(i)],'-r')
    text(tau(end),ctone(i),tonename(i),'FontSize',10)
    hold on
end

semitone=[277 311 370 415 466 554 622 740 831 932];
%frequency of semitones C1#
tonename2={'C#','D#','F#','G#','A#','C#','D#','F#','G#','A#','C#','D#','F#','
G#','A#'};
for i=1:length(semitone)
    plot([tau(1),tau(end)],[semitone(i),semitone(i)],'-b')
    text(tau(end-5),semitone(i),string(tonename2(i)),'FontSize',9)
    hold on
end
title('music score of Comfortably Numb','Fontsize',20)
ylabel('Frequency [Hz]','Fontsize',20);
xlabel('Time [s]','Fontsize',20)

%%
%%
%this part mean to filter around low frequency to issolate the base
S=y(1:length(k));
St=fft(S);
figure(4)
subplot(2,1,1)
plot(ks,abs(fftshift(St))/abs(max(St)))
xlim([0,2000])
xlabel('Frequency [Hz]');
ylabel('fft(S)')
title('Origin Frequency Distribution')
%setting signal, frequency domain in same array size, and fft signal
%plot spectrum in signal domain(Hz)

tau=1e-4;
k0=100;
filter = exp(-tau*(k - k0).^2);
St=St'.*filter;
```

```
S=ifft(St);
%as we can see
subplot(2,1,2)
plot(ks,abs(fftshift(St))/abs(max(St)))
xlim([0,2000])
xlabel('Frequency [Hz]');
ylabel('fft(S)')
title('Isolated Base frequnce')

a=1000;%after comparing different setting of parameter a,
%value 2000 can have a relative promissing result in both time and frequency
resolution
tau=0:0.1:tr_real;
i=1;
for j=1:length(tau)

    g = exp(-a*(t - tau(j)).^2);
    sg=g.*S;
    sgt=fft(sg);
    sgt_spec(:,j)= fftshift(abs(sgt));
    i=i+1;
    disp(['2scaling',num2str(j),'/',num2str(length(tau))])
    figure(11)
    sgtf=fftshift(sgt);
    plot(ks,abs(sgtf)/max(abs(sgtf)))
    xlim([0,2000])

    xlabel('Frequency [Hz]');
    ylabel('fft(S)')
    title('Frequency spectrum of isolated base during scaling')
    pause(0.002)
end
figure( 'Position', [10 10 2000 400])
% pcolor(tau,ks,sgt_spec)
spec=1-(abs(sgt_spec)./max(abs(sgt_spec)));
pcolor(tau,ks(1,0.5*length(ks):length(ks)),spec(0.5*length(ks):length(ks),:))
%truncate the frequency domain in half to save plotting time
title('isolated base of Comfortably Numb')
ylabel('Frequency [Hz]');
xlabel('Time [s]')
ylim([0,400])
shading interp
colormap(gray)
hold on


%%
disp(['please wait for the plot of spectrogram'])
disp(['finish'])
```