AMATH 482 Homework 2

Jessica Shi, 1762333

February 3, 2021

Abstract

In this homework, we use the Gabor Transform to process audio signals. We reproduce the music score for two music clips with different instruments. We use proper Gabor filtering to filter out overtone and get both the spectrograms and music scores for the 2 music pieces.

1 Introduction and Overview

In this homework, I have 2 music pieces from the songs Sweet Child O' Mine by Guns N' Roses and Comfortably Numb by Pink Floyd and I would use the Gabor filtering to reproduce the music scores for the guitar in the first clip and the bass for the second clip. In the first clip, since there is only one instrument, it is relatively easy to get the clean music score. On the contrary, in the second clip, there are multiple instruments, thus it would be harder to filter out the other instruments. Fortunately, the frequency range for guitar and bass, which are the main instruments in this clip, are very different, so I would first just focus on the frequency range for bass to find the music note, and then focus on the guitar solo part. In both music clips, I would use the Gaussian function as my filter.

2 Theoretical Background

The fundamental concept used in this homework is the Gabor Transform. It is a derived method from the Fourier Transform. We know that Fourier Transform could help make analysis of the signals in the frequency domain. But we will eventually lose information about the position of signals in the time domain. The Gabor Transfrom helps us to mitigate the problem.

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

This is the equation for the Gabor Transform, or the Short-Time Fourier Transform (STFT). τ is the center of the window and $g(\tau - t)$ is the filter function. It divides the original data into several windows and apply the Fourier Transform in each window so that we could keep part of the time information.
The key idea is to choose an appropriate window size. If the window size is too small, we are just looking at each time point and thus have no frequency information. On the other hand, if the window size is too big, we are just doing the normal Fourier Transform over the whole range. By choosing proper window size, we can get a good balance on both the time and frequency information.

For the filter function, I would use the Gaussian filter, which are listed below.

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

3 Algorithm Implementation and Development

The specific implementation is done using MATLAB and is divided into several steps according to the problem requirements.

3.1 Part One - GNR Clip and Guitar Note

The music note for this song is relatively easy to find since guitar is the only instrument in this clip. First, I read the audio file into MATLAB and define the domain for the Gabor Transform. Then, I start to choose the parameters for the Guassian function, which is my chosen filter function. After trying different values, I choose the window size `a` to be 1500 and the time step to be 0.2. I think these values are proper, which can be shown in the plots in the next section. Then, I loop over the time span of the song and apply the `fft` at each time step. Also, I use the `max` function to find the index of the strongest frequency at that time step in order to get a relatively accurate music score. I store this music note in the variable `guitar_note` for later plotting. Also, I store the spectrogram data separately in variable `spect_guitar`. After the loop is finished, I plot both the spectrogram and the music score for the guitar.

3.2 Part Two - Floyd Clip and Bass, Guitar Note

The basic process for this song is similar to the previous one. But the difficulty here is to filter out other instruments. This song contains several different instruments. Fortunately, we know that bass has very low sound, which means that its music notes are low and the corresponding frequencies are low as well. According to the information online (https://www.masteringthemix.com/blogs/learn/understanding-the-different-frequency-ranges), we know that the frequency range for bass is around 60-250 Hz. So our final plot will focus on this range.

Again, after defining the domain for Gabor Transform, I start to decide the parameters for the Gaussian function. For the spectrogram, since the song is long and MATLAB might be out of memory if I process the complete clip, I just choose the first 23 seconds since basically the bass part is repeating if we listen carefully. Thus, for the spectrogram, I use the window size of 200 and the time step to be 0.5.(NOTE: I think smaller time step could be better but my MATLAB have unknown error with time step smaller than 0.5, so I use 0.5).

For the music note, I am able to plot the complete clip, so here I use a window size of 1500 and a time step of 0.3. Similarly, I loop over the time span of the clip and apply the `fft` at each time step. I then use the `max` function to find the index of the strongest frequency and I store the music notes in the variable `bass_note`. I get the spectrogram variable `spect_bass` during the loop as well. After the loop is finished, I plot both the spectrogram and the music score for the bass.

For the guitar, since it has a higher range of frequency, it is clear to observe on the music score plot. Thus, to separate the bass and guitar, I plot them in different frequency range. The specific range and notes are specified in the next section.

4 Computational Results

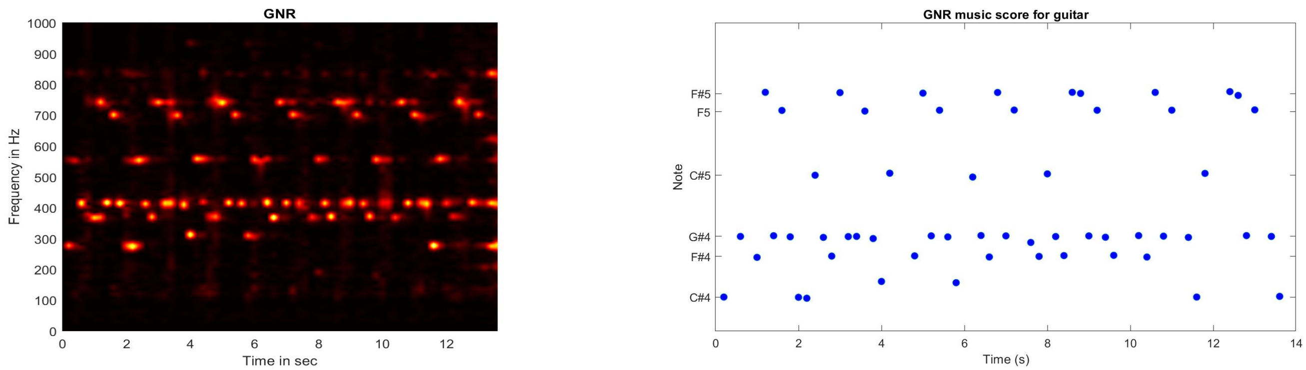
4.1 GNR - Guitar's Music Score

Below is the spectrogram for the guitar in GNR clip and the music score for it. Figure 1(a) is the spectrogram for the guitar in GNR clip. We could see that it has a regular pattern. In fact, it is repeated if we listen to the clip. Also, even though we use the Gaussian filter function, there are still some overtones.

Figure 1(b) is the music score for the guitar in GNR clip. Comparing to the spectrogram, the pattern is clearer. Also, I put the corresponding music notes on the y-axis. From the plot, we know that the main notes for this clip is C#4, F#4, G#4, C#5, F5 and F#5.

4.2 Floyd - Bass's and Guitar's Music Score

Figure 2(a) is the spectrogram for the bass in Floyd's clip. Since the music clip is long and the notes are actually repeating, I choose the first 23 seconds in the clip and plot it. Also, from the previous section, we know that the frequency range for bass is about 60-250 Hz, thus, I choose a range of 0 - 300Hz to show the



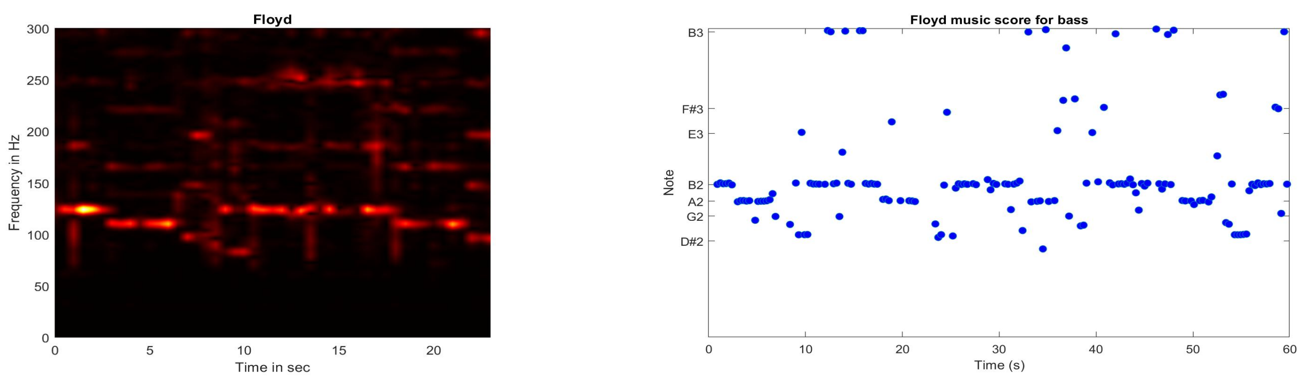
(a) The Spectrogram for Guitar in GNR Clip

(b) The Music Score for Guitar in GNR Clip

Figure 1: GNR

bass's spectrogram.

Similarly, for the music score in Figure 2(b), I choose the range of 0 - 250 Hz, which covers the range of bass. And the pattern is clearer in this plot. We could see that the main notes are around D#2, G2, A2, B2, E3 and F#3. This is reasonable since bass always has a very low sound.



(a) The Spectrogram for Bass in Floyd Clip

(b) The Music Score for Bass in Floyd Clip

Figure 2: Floyd Bass

In Figure 3, I plot the music score for the guitar in the Floyd clip. We know that guitar has a very different frequency range from bass. Also, this clip is mainly consisted of guitar and bass sounds. Thus, it is reasonable to say that the frequency higher than 250 is created by the guitar. Therefore, I plot the frequency from 250 to 800Hz to show the music score for guitar. Also, I put the corresponding note on the y-axis. We then could see from the plot that the main notes are around C4, E4, F#4, B4, D5, E5, F#5 and G5. Actually, if we listen to the clip, we could know that there is no obvious pattern for this guitar solo. So, in the plot, the notes are very spread out.

5 Summary and Conclusions

To conclude, in order to get the music score for an instrument from audio files, first we need to understand the frequency range of the instrument. Also, there are often some overtones in the music clips, so we need to use a filter function. In this homework, I choose to use the Gabor Transform and the Gaussian function as the filter function. By choosing proper window size and time step, we are able to create the spectrogram and music score for a specific instrument. In the first clip, since there is only guitar sound, we just filter out

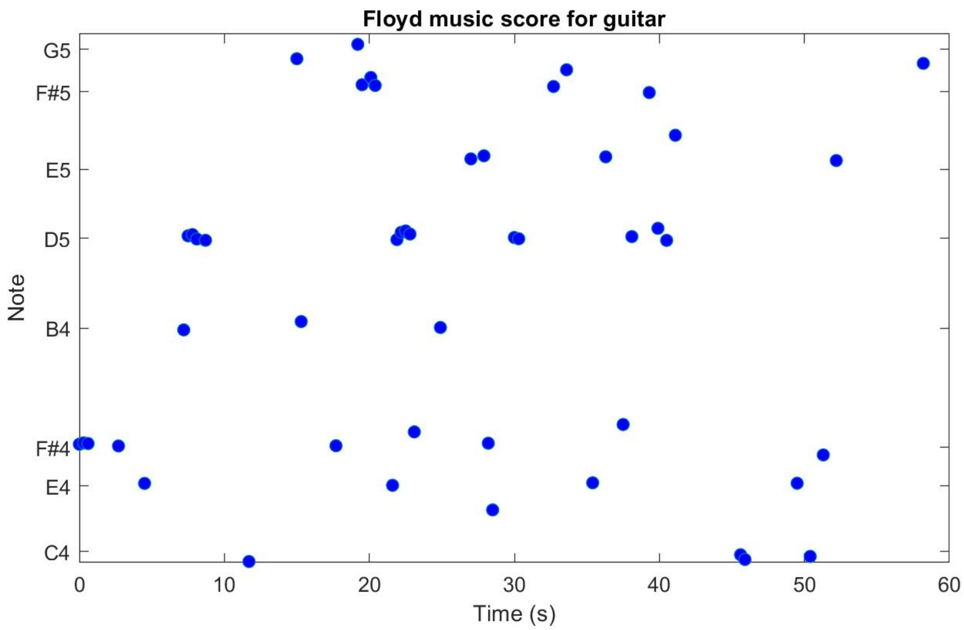


Figure 3: The Music Score for Guitar in Floyd Clip

overtones and get the music score for guitar. But in the second clip, there are both sounds from guitar and bass. Therefore, in order to distinguish between these two sounds, we find the proper range of frequency for each instrument and then plot the corresponding music score for each of them.

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`.
- `y = linspace(x1,x2,n)` returns a row vector of `n` evenly spaced points between `x1` and `x2`.
- `Y = fftshift(X)` rearranges a Fourier transform `X` by shifting the zero-frequency component to the center of the array.
- `Y = fft(X)` computes the discrete Fourier transform (DFT) of `X` using a fast Fourier transform (FFT) algorithm.
- `[M,I] = max(A)` returns the index into the operating dimension that corresponds to the maximum value of A.
- `plot(X,Y)` creates a 2-D line plot of the data in `Y` versus the corresponding values in `X`.
- `pcolor(X,Y,C)` creates a pseudocolor plot using the values in matrix `C` and 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

```
clear; close all; clc

% figure(1)
```

```
% [y, Fs] = audioread('GNR.m4a');
% tr_gnr = 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);



% Guitar part for GNR -----------------------------------------------------
% Define domain
[y, Fs] = audioread('GNR.m4a');
tr_gnr = length(y)/Fs; % record time in seconds
v = y';
n = length(v);
t2 = linspace(0, tr_gnr, n+1);
t = t2(1:n);
k = (2*pi/tr_gnr)*[0:n/2-1 -n/2:-1];
ks = fftshift(k);

a = 1500;
d_tau = 0.2;

tspan = 0:d_tau:tr_gnr;
spect_guitar = zeros(length(tspan), n);
guitar_note = zeros(1, length(tspan));


for i = 1:length(tspan)
    f = exp(-a*(t - tspan(i)).^2);
    window = f .* v;
    window_f = fft(window);
    [M, I] = max(window_f);  % find the maximum frequency
    guitar_note(1, i) = abs(k(I))/(2*pi);
    spect_guitar(i, :) = fftshift(abs(window_f));
end

figure(1)
pcolor(tspan, (ks/(2*pi)), spect_guitar.'),
shading interp
ylim([0 1000])
colormap hot
title("GNR")
xlabel('Time in sec'), ylabel('Frequency in Hz')


figure(2)
plot(tspan, guitar_note, 'o', 'MarkerFaceColor', 'b');
yticks([277.18,369.99, 415.30, 554.37, 698.46, 739.99]);
yticklabels({'C#4','F#4','G#4', 'C#5', 'F5', 'F#5'});
ylim([200 900])
title('GNR music score for guitar');
xlabel('Time (s)');
ylabel("Note");

% Bass part for Floyd -----------------------------------------------------
% Define domain
[y, Fs] = audioread('Floyd.m4a');
tr_floyd = length(y)/Fs; % record time in seconds
v = y';
n = length(v);
t2 = linspace(0, tr_floyd, n+1);
t = t2(1:n);
k = (2*pi/tr_floyd)*[0:(n-1)/2 -(n-1)/2:-1];
ks = fftshift(k);

a = 1500;
d_tau = 0.3;
```

```
tspan = 0:d_tau:tr_floyd;

% For smaller range spectrogram
% a = 200;
% d_tau = 0.5;
% tspan = 0:d_tau:23;

spect_bass = zeros(length(tspan), n);
bass_note = zeros(1, length(tspan));

for i = 1:length(tspan)
    f = exp(-a*(t - tspan(i)).^2);
    window = f .* v;
    window_f = fft(window);
    [M, I] = max(window_f);  % find the maximum frequency
    bass_note(1, i) = abs(k(I))/(2*pi);
    spect_bass(i, :) = fftshift(abs(window_f));
end

figure(3)
pcolor(tspan, (ks/(2*pi)), spect_bass.'),
shading interp
ylim([0 300])
colormap hot
title("Floyd")
xlabel('Time in sec'), ylabel('Frequency in Hz')

figure(4)
plot(tspan, bass_note, 'o', 'MarkerFaceColor', 'b');
yticks([77.782, 97.999, 110.00, 123.47, 164.81, 185.00, 246.94]);
yticklabels({'D#2','G2','A2', 'B2', 'E3', 'F#3', 'B3'});
ylim([0 250])
title('Floyd music score for bass');
xlabel('Time (s)');
ylabel("Note");

figure(5)
plot(tspan, bass_note, 'o', 'MarkerFaceColor', 'b');
yticks([261.63, 329.63, 369.99, 493.88, 587.33, 659.26, 739.99, 783.99]);
yticklabels({'C4','E4','F#4', 'B4', 'D5', 'E5', 'F#5', 'G5'});
ylim([250 800])
title('Floyd music score for guitar');
xlabel('Time (s)');
ylabel("Note");
```