Coding Project 2: Parsing musical frequency signatures

Lucas Liu

Abstract

In this project, spectral analysis on an audio file was done to isolate specific instrument in the track. Specifically, the Gabor Transform was applied to create a spectrogram of the audio file. A Gaussian filter centered about the peak frequencies was then used to isolate these frequencies, which were then further filtered by a make-shift Shannon filter to single out the bass and guitar portions of the audio.

1 Introduction

Musical notes are sounds that have a specific pitch and frequency. Each note is associated with a particular frequency, which determines its position on the musical scale. The frequency of a note is measured in hertz (Hz), which is the number of cycles per second. These frequencies have been used for centuries to create music, and have played a crucial role in the development of musical traditions and cultures throughout history.

The concept of musical notes and their frequencies has been studied and refined over time, with various tuning systems and instruments being developed to produce these sounds accurately. These systems and instruments have allowed musicians to express their creativity and emotions through sound, leading to the creation of countless musical masterpieces.

The remainder of this report summarizes the theoretical background, numerical methods, and results used in the project which conducted spectral

analysis and filtering on an audio file to single out instruments at specific frequencies. The instruments chosen were the most prominent ones, indicated by a higher value in frequency space.

2 Theoretical Background

This section briefly covers the mathematical techniques used in the project.

2.1 The Gabor Transform

The Gabor Transform is a mathematical tool used for analyzing signals and images in the time-frequency domain. While the Fourier Transform provides information on the structure of frequencies that make up a signal as a whole, the Gabor transform shows how a signals frequencies vary over time. It was first introduced by Hungarian physicist and engineer Dennis Gabor in 1946. The idea behind the Gabor Transform is to decompose a signal or image into a set of simple building blocks, called "Gabor atoms," that are defined by their time and frequency characteristics.

The derivation of the Gabor Transform starts with the idea of a window function, which is used to isolate a portion of the signal or image.

For a filter centered at T, the filter takes the form:

$$\text{filter } = g(t - T)$$

A Gabor atom is then defined as the product of this window function and a complex exponential function that depends on both time and frequency. The Gabor Transform is obtained by computing the inner product of the signal or image with each Gabor atom, over a range of time and frequency scales.

$$\tilde{f}(T, k) = \int_{-\infty}^{\infty} f(t)g(t - T)e^{-2\pi ikt}dt$$

This results in a time-frequency representation of the signal or image, where the strength of the representation at each time-frequency point reflects the presence of the corresponding frequency components in the signal or image. The Gabor Transform has proven to be a powerful tool for signal processing

and image analysis, with applications in areas such as speech recognition, image compression, and feature extraction.

2.2 The Discrete Gabor Transform

The Discrete Gabor Transform (DGT) is a variant of the Gabor Transform that is designed to handle discrete signals and images. The main difference between the DGT and the continuous Gabor Transform is that the DGT operates on a discrete set of time and frequency points, rather than on continuous functions. The DGT is used to analyze signals and images in the time-frequency domain by approximating the continuous Gabor Transform with a finite set of Gabor atoms.

The derivation of the DGT starts by discretizing the time and frequency domains into a set of evenly spaced points. As such,

$$k = mw_0, T = nt_0$$

where t_0 is the stepsize and n is the number of steps, m and w_0 are associate values.

A set of Gabor atoms, also called basis functions, is then generated for each frequency component of interest. The Gabor atoms are created by sampling the window function and the exponential function at the discrete time points, and then normalizing the resulting functions to have unit energy. Finally, the DGT is obtained by computing the inner product between the signal or image and each Gabor atom.

$$\tilde{f}_g(T,k) \approx \int_{-\infty}^{\infty} f(t)g(t-nt_0)e^{-2\pi imw_0 t}dt$$

The result is a time-frequency representation of the signal or image, where each element of the representation represents the strength of the corresponding frequency component in the signal or image. The DGT is widely used in signal processing and image analysis applications, as it provides a convenient and efficient way to analyze signals and images in the time-frequency domain.

3 Numerical Methods

The MATLAB code used in the project is provided below:

```
   clear all; close all; clc;
load 'CP2_SoundClip.mat'

Fs = 44100; %sample rate of the sound clip

S = y'; % transposes in order to have consistent dimen-
sions
w = length(y)/4; % break the spectrogram up into four time
windows, otherwise it
% will be too big for MATLAB/autograder to run.

S1 = S((1-1)*w+1:1*w); % this will isolate the correct win-
dow
S2 = S((2-1)*w+1:2*w); % this will isolate the correct win-
dow
S3 = S((3-1)*w+1:3*w); % this will isolate the correct win-
dow
S4 = S((4-1)*w+1:4*w); % this will isolate the correct win-
dow

L = length(S1)/Fs; % length of each window in seconds
n = length(S1); % number of elements in each window
t = [0:1/Fs:L - 1/Fs]; % t in sec.  relative to the start
of the window
tau = 0:0.1:L; % discretization for the Gabor transform
k = 2*pi*(1/L/2)*[0:n/2-1 -n/2:-1]; % discretization in
frequency space
ks = fftshift(k); % gotta shift them freqs.

Sgt_spec = zeros(length(ks),length(tau)); % initializing
the function for the spectrogram

%Gabor Transform Parameters
a = 400; %this will give you the correct width, so exp(-
a(...))
```

```
range = [1:1800]; %use this when finding the max and in-
dex of the transformed Gabor filtered signal
```

The first section of code above initializes the variables and objects necessary to conduct the spectral analysis. Most importantly, the audio file named 'CP2_SoundClip.mat' is loaded in and variables representing the Sample rate, time domain, frequency domain, and spectrogram of the sound clip are all instantiated. The sound clip was split into four portions ($S1, S2, S3, S4$) for feasibility.

```
   % apply Gabor Transform to achieve spectrogram, only
keeping the peak % frequencies
   for j = 1:length(tau)
   g = exp(-a*(t - tau(j))^2); % Window function apply gauss
filter to reading (time amp space)
   Sg = g.*S1;
   % apply FFT (k, amp space)
   Sgt = fft(Sg);
   % find index of peak frequency
   [maxVal,maxIndex] = max(Sgt(range));
   % create gauss filter in frequency space centered around
peak frequency
   g2 = exp((-1/L)*(abs(k)-k(maxIndex)).^2);
   % apply filter to FFT
   Sgt_filt2 = g2.*Sgt;
   Sgt_spec(:,j) = fftshift(abs(Sgt_filt2)); % We don't want
to scale it end
   A1 = Sgt_spec;
```

Afterwards, the Gabor transform, using a Gaussian filter, is conducted on the sound clip. However, only the peak frequencies are kept. This is done by finding the location and value of the max frequencies and applying a Gaussian filter on the spectrogram centered about the max frequencies. This process was conducted for portions $S1, S2, S3$, and $S4$ and the corresponding spectrograms were stored in variables $A1, A2, A3$, and $A4$, repectively.

After inspecting the spectrogram, it was estimated that the frequencies produced by the bass lay in the range of 250-300 Hz and the frequencies produced by the guitar lay in the range of 460-480 Hz. Consequently, a

makeshift Shannon filter was applied to the original sound clip to isolate these frequencies, setting the value of all other frequencies to 0.

```
   % Isolating Bass
   S = y'; % transposes in order to have consistent di-
mensions
   kt = 2*pi*(1/L/2)*[0:n/2-1 -n/2:-1]; % freq discretiza-
tion
   ks = fftshift(kt); % shift freq discretization so from
least to greatest
   % take fourier transf of S
   St = fftshift(fft(S));
   % filter frequencies, set all values for frequenices
not in 250-300 to be 0
   % find index of frequenicies (0 is located at size/2+1)
   f1_index = size(ks,2)/2+1+250;
   f2_index = size(ks,2)/2+1+300;
   % create new matrix of all zeroes
   Sfilt = zeros(size(St));
   % copy over values in found range in St to new matrix
   Sfilt(1,f1_index:f2_index) = St(1,f1_index:f2_index);
   A5 = ifft(Sfilt)'; %Shape:  1938240x1 double
```

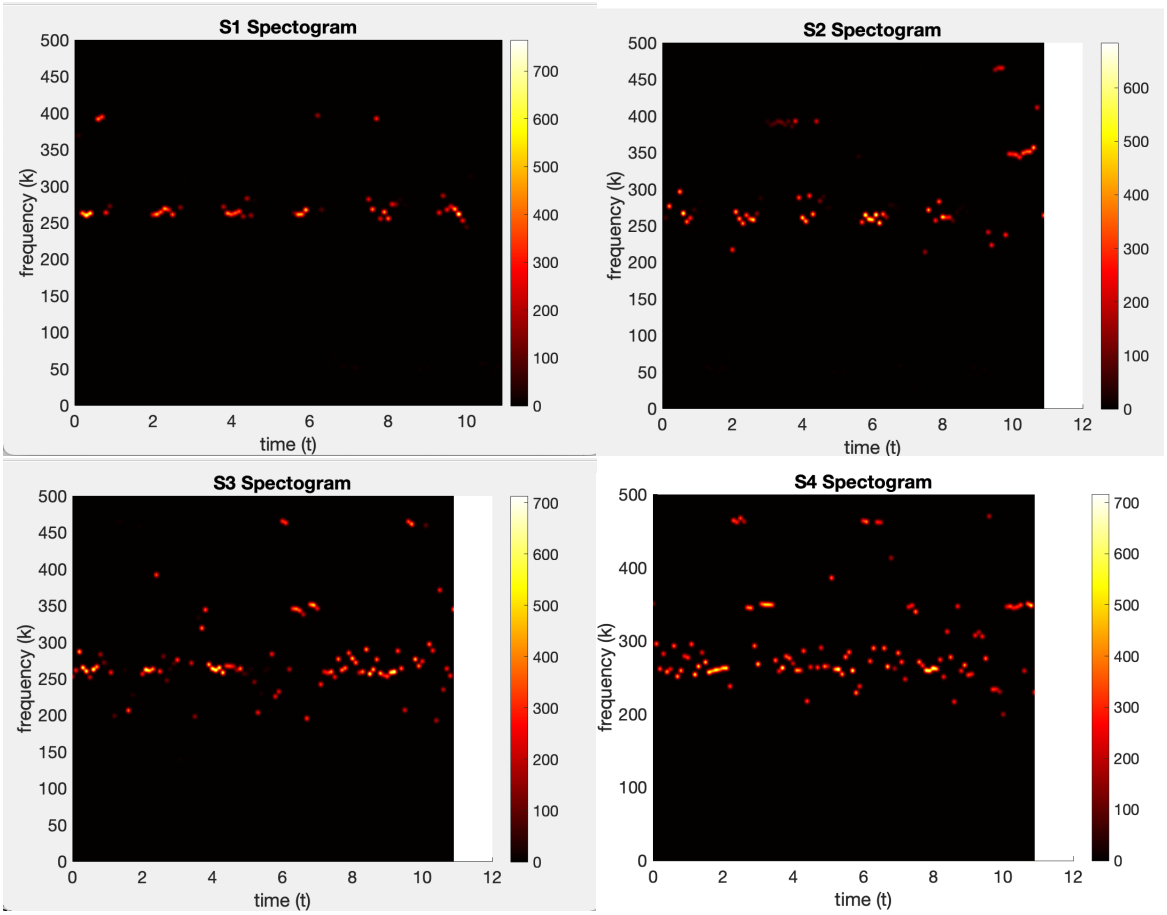
Inverting the filtered spectrogram yields a sound clip with only the bass instrument. The same process was done to isolate the guitar:

```
   % Isolating Guitar
   S = y'; %reinitialize the S from the previous part above.
   % take fourier transf of S
   St = fftshift(fft(S));
   %frequencies that should be part of bass around 275 Hz
   % filter frequencies, set all values for frequenices
not in 260-480 to be 0
   % find index of frequenicies (0 is ocated at size/2+1)
   f1_index = size(ks,2)/2+1+460;
   f2_index = size(ks,2)/2+1+480;
   % create new matrix of all zeroes
   Sfilt = zeros(size(St));
   % copy over values in found range in St to new matrix
```

```
Sfilt(1,f1_index:f2_index) = St(1,f1_index:f2_index);
A6 = ifft(Sfilt)'; %Shape:  1938240x1 double
```

4 Results

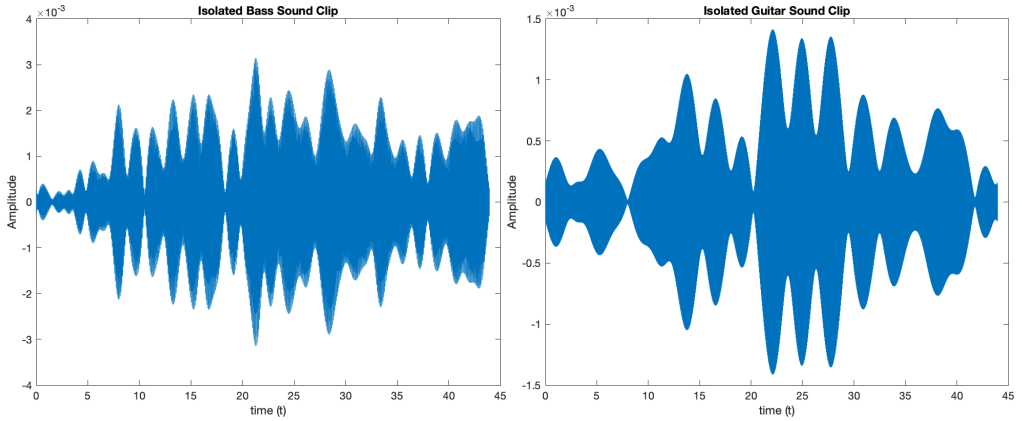
Below are the spectrograms for each quarter of the sound clip:



The spectrograms reveal a visual representation of how the max frequencies in the sound clip vary over time. The dots reveal when these frequencies were the most dominant and the colors of the dots reveal how prominent they were with white being the most and black being the least. From the

spectrograms, we can see that frequencies in the 250-300 Hz range were very prominent throughout the sound clip as well as frequencies 460-480 Hz in the higher range. Due to the nature of the instruments, the peak frequencies sin the lower range are most likely from the bass while the frequencies in the higher range are resultant from the guitar.

After filtering about these frequency ranges for the whole sound clip, the resultant bass isolated sound clip and guitar isolated sound clip are:



The plots show how the amplitude of these instrument varied over time. It can be seen that both instruments hit their peak amplitudes towards the middle of the track.

5 Conclusion

In summary, through the use of the Gabor Transform and Gaussian filtering, we are able to create a spectrogram of the peak frequencies in a sound clip. By inspecting the spectrogram, we are able to deduce the frequencies ranges where the most prominent instruments reside, allowing us to filter the original sound clip to isolate these instruments.

One drawback of the Gabor Transform that became evident during this project is the high computational costs. Applying the transform and plotting the resulting spectrogram took very long times and made my whole system very slow. Nonetheless, this project was still very enlightening.

Acknowledgment

Stackoverflow, offical MATLAB pages, and the Canvas Discussion page were referenced during this project.