AMATH 482 Homework 2

Shane Fretwell

February 9, 2021

Abstract

Anyone who has played the guitar or knows someone who has can tell you that tabs are important. Tabs tell you when and what notes are in a song, so that you can play it. Finding a good source of the tabs you want is essential to learning new songs. There are plenty of free online sources for popular songs, but what if you want to play a song that you found in an uncharted corner of Spotify? What if it is a song that you cannot find tabs for online? If you have experience in signal processing, you can create your own.

1 Introduction and Overview

While the signal processing methods here apply to any song, we will work with two famous clips: one from *Sweet Child O' Mine* by Guns N' Roses and one from *Comfortably Numb* by Pink Floyd – both guitar solos. We will process these clips in order to extract from them enough information to construct partial guitar tabs for *Sweet Child O' Mine* and partial guitar and bass guitar tabs for *Comfortably Numb*. In order to do so, we will need information about when and what notes are being played.

1.1 Fourier Transform – Powerful but Ignorant

The most basic and essential of signal processing tools, the Fourier Transform takes a signal and returns the frequencies present within. This gives us information about what notes are being played, but not when.

1.2 Gabor Transform – Powerful and Selectively Ignorant

Complete decomposition of a signal gives you complete information of the frequencies within the signal, at the cost of any information about the time each frequency is present. To know when, we will decompose the signal in parts and combine the information about each part to get a fuller picture of the when and where of things.

2 Theoretical Background

2.1 Fourier Series

As a short review, we discussed in lecture, the Fourier Transform is the decomposition of intensity over time and/or space into its composite frequencies. Sine and cosine waveforms are used for our purposes because this is the natural waveform of acoustic phenomena. Thus, we have the Fourier series constructed from sines and cosines of a given signal function s(x):

$$s_N(x) = \frac{a_0}{2} + \sum_{k=1}^{N} a_k cos\left(\frac{2\pi kx}{P}\right) + b_k sin\left(\frac{2\pi kx}{P}\right) \tag{1}$$

The coefficients a_k and b_k are given by inner product of the waveforms of frequency k and the signal $s(x)$.

2.2 Gabor Transform

To determine when a frequency was detected within the signal, one can make use of a Gabor transform, which essentially performs the Fourier transform on overlapping windows of time within a signal. This reveals whether a given frequency was present in the signal within each window of time.

Performing a Gabor transform is done by multiplying a signal by Gaussian filters that are centered at many different times. This removes any frequency information from outside of the time window, and thus when a Fourier transform is performed on this new signal, only frequencies within the time window will be present. The width of the filter will determine the width of the waveforms that are possible to detect, so care must be taken in choosing appropriate width for the problem at hand. A narrow filter may increase the precision of time information gleaned, but it also makes impossible the detection of frequencies wider than the filter itself (1).

2.2.1 Gabor Filter

A Gaussian filter with an L-2 norm of 1 is shown in equation 2. An L-2 norm equal to 1 ensures that the overall energy of the signal is not affected by the filtering process, and the transform is invertible, should we wish to recover the signal from the transform (1).

$$g(\tau, t) = \sqrt{\frac{2}{\sigma^2 \pi}} e^{\frac{-(t-\tau)^2}{2\sigma^2}} \tag{2}$$

To make things more concrete, consider a spectrogram (using the Gabor filter from equation 2) of the same signal using different values of sigma, the width parameter. When we use a smaller sigma, we have sharper definition in the time dimension, and we can tell when each note begins and ends. Too narrow, however, and the exact note being played will be difficult to discern due to poor resolution in the frequency dimension. This is shown in Figure 1. One is unable to discern the notes being played in the first spectrogram, and unable to resolve between strums of the same note in the final spectrogram. See subsubsection 3.2.1 for how to interpret spectrograms.

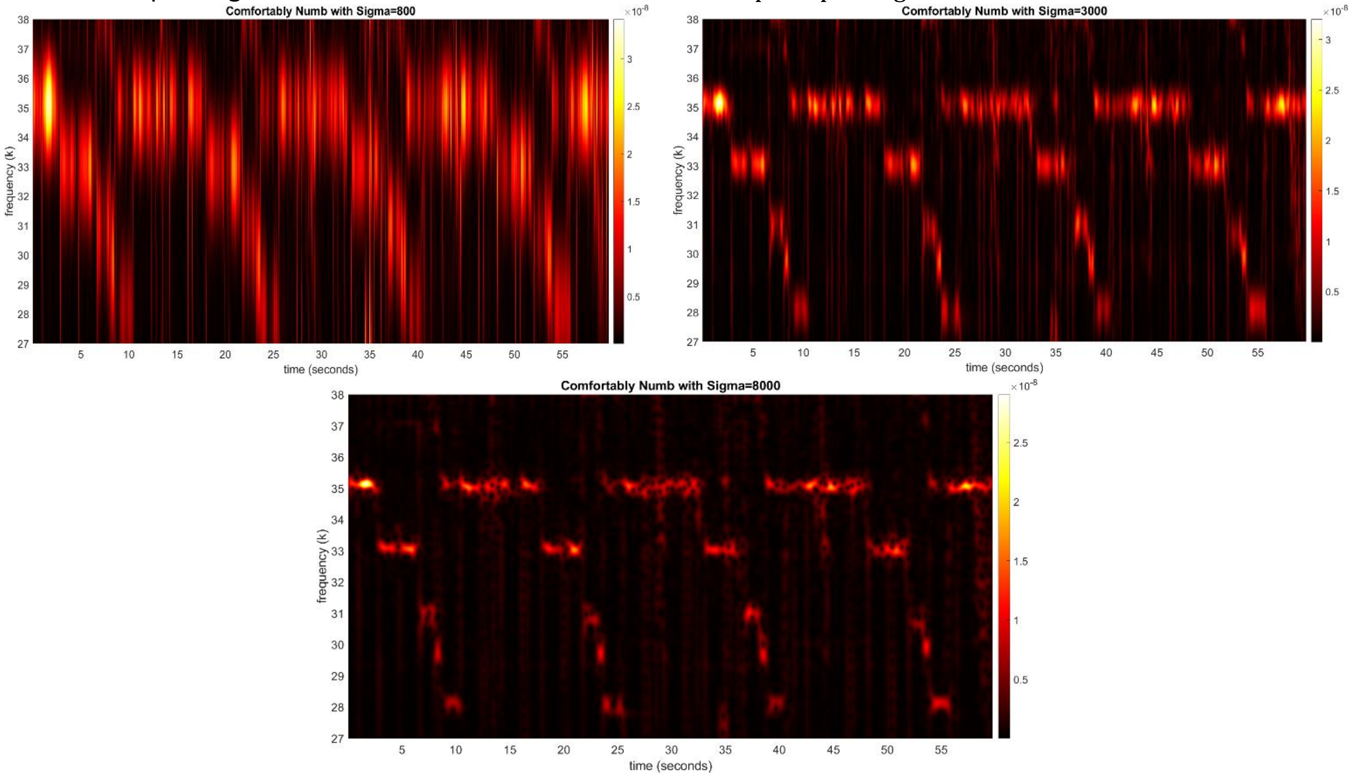

Figure 1: Spectrograms of the Gabor Transform with differing widths of filter. This demonstrates the tradeoffs associated with widening and narrowing your filter. The frequencies 27 to 38 represent notes from D2 to C#3.

3 Algorithm Implementation and Development

3.1 Applying the Gabor Transform

3.1.1 Deriving the Filter with an L-2 Norm of One

The calculation of the L-2 Norm of a filter $g(\tau, t)$ is seen in equation 3 (2). Notice that the square root can be ignored, since we intend to fix the L-2 Norm to 1, and $1^2 = 1$ anyways. The Gaussian filter comes from the Gaussian function, so we will start with the mother Gaussian as $\hat{g}(\tau, t)$ in equation 4.

$$\|g(\tau, t)\| = \sqrt{\int_{-\infty}^{\infty} \left(g(\tau, t)\right)^2 d\tau} \tag{3}$$

$$\|\hat{g}(\tau, t)\| = \int_{-\infty}^{\infty} \left(\frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(t-\tau)^2}{2\sigma^2}}\right)^2 d\tau \tag{4}$$

Through substituting $\hat{\sigma} = \sigma\sqrt{2}$ and some further algebraic manipulation, we make the integral in equation 4 look like the integral of the $Normal(t, \hat{\sigma}^2)$ probability density function over the real line, which of course evaluates to one. After this, we have equation 5, which implies that the L-2 Norm of equation 2 (our chosen Gabor Filter) is indeed equal to 1.

$$\|\hat{g}(\tau, t)\| = \sqrt{2/\pi} \tag{5}$$

3.1.2 Using a Gabor Filter

Applying this filter is simple; we multiply every point in the discrete signal by the filter at same value of *t*.

Algorithm 1: Applying the Filter

```
for each tau
  filter = sqrt(2/(sigma^2*pi))*exp(-0.5*(((1:L) - tau)/sigma).^2); % L is the length of the signal
  tframe = y.*filter; % y is the signal

  freqsGT = fft(tframe); % Perform Fourier transform on the Gabor window
  allfreq = abs(freqsGT/L); % Divide by L to obtain frequencies in Hz
  posfreq = allfreq(1:(L/2 + 1));
  posfreq(2:(end - 1)) = 2*posfreq(2:(end - 1)); % Retrieve only positive frequencies

  specf(:, tau) = posfreq; % Store the spectrum of every Gabor window
end
```

3.2 Determining a Tenable Filter Width

To obtain the best sigma for use in our filter, we must try many different values and adjust based on the results of each until we arrive upon a useful filter.

3.2.1 Displaying the Results of a Gabor Transform

Once we have our Gabor transform in a matrix as results from algorithm 1, this matrix is the spectrum of each Gabor window, lined up one after another by increasing tau values. A spectrogram simply takes that matrix and makes large magnitude frequencies in each spectrum appear brighter than other, lower magnitude values. For specifics on the generation of the spectrograms, see appendix A, 'pcolor.' Thus, the spectra being aligned by frequency, we can see the evolution of the spectrum of the sliding Gabor window over time. This is how we can see that, at about four seconds into the *Comfortably Numb* clip, the 33rd note, representing G#2, is played.

3.3 Comfortably Numb Lead Guitar

While the spectrograms for the instrumentals so far have been relatively clean, the same is not true for those for the guitar in *Comfortably Numb*. There are frequency artifacts from the bass and rhythm guitars that make it difficult to single out the notes being played by the lead guitar.

3.3.1 Overtones

In reality, guitars do not play only a single frequency with each strum; they also play overtones that are integer multiples of the fundamental frequency of the note, due to the nature of vibrating strings. So, the note F#2 being played by the bass guitarist is made up mostly of the frequency 92.5Hz, but it also contains 185Hz, 277.5Hz, and 555Hz, the last of which corresponding closely to the fundamental frequency of the note C#5. This means that whenever the bass guitarist is playing a F#2, the note C#5 in the spectrogram is also lit up, giving a potentially false positive for the guitarist playing the C#5. Thus, when a note N in the spectrogram of the guitar part of *Comfortably Numb* is lit up, we can only be sure that the lead guitar played that note if the bass guitar did not play a note with an overtone frequency equal to the fundamental frequency of N.

4 Computational Results

Having the tools to generate useful spectrograms and understanding how to interpret them as an efficient method of representing the Gabor Transform, we can now produce musical scores from each spectrogram.

4.1 Sweet Child O' Mine Guitar

The notes in the *Sweet Child O' Mine* clip range from about D#3 to A5, so this is the window of frequencies that we set the spectrogram to encompass. Experimentation led to finding the sigma value of 4000, which gives the spectrogram in figure 2. From this spectrogram we can visually extract the following bars.

49/C4	**61/C5**	**56/G4**	**54/F4**	**66/F5**	**56/G4**	**65/E5**	**56/G4**
49/C4	61/C5	56/G4	54/F4	66/F5	56/G4	65/E5	56/G4
51/D4	61/C5	56/G4	54/F4	66/F5	56/G4	65/E5	56/G4
51/D4	61/C5	56/G4	54/F4	66/F5	56/G4	65/E5	56/G4
54/F4	61/C5	56/G4	54/F4	66/F5	56/G4	65/E5	56/G4
54/F4	61/C5	56/G4	54/F4	66/F5	56/G4	65/E5	56/G4
49/C4	61/C5	56/G4	54/F4	66/F5	56/G4	65/E5	56/G4
49/C4							

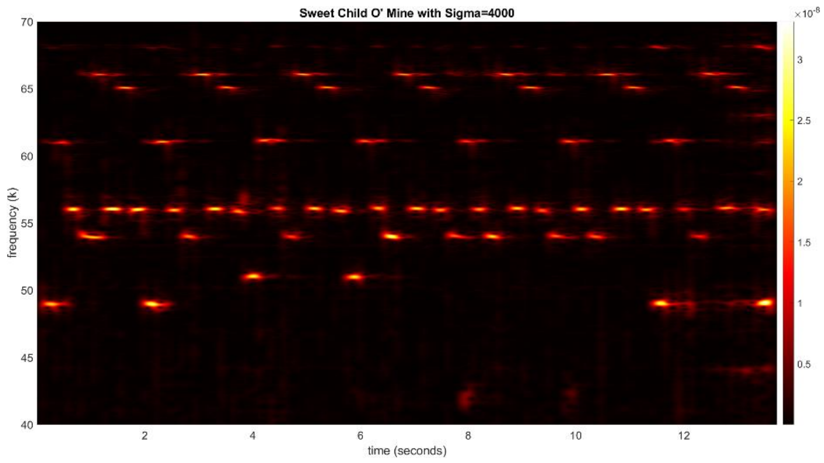

Figure 2: Spectrogram of the Gabor Transform of *Sweet Child O' Mine*.

4.2 Comfortably Numb Bass

The notes the bass plays in the *Comfortably Numb* clip range from about D2 to C#3, so the spectrogram is restricted to these notes. Once again, experimentation led to a sigma of 3000, which gives the spectrogram in figure 3. From this spectrogram and its sister spectrograms from figure 1, we can visually extract the following bars. Specifically, we can cross-reference the clear start and end of notes from the narrow filter spectrogram with the frequencies of these notes that are more easily discernable in the middle and wide filter spectrograms.

		35/A#2	**33/G#2**	**33/G#2**	**31/F#2**	**30/F2**	**28/D#2**
35/A#2	35/A#2	35/A#2	33/G#2	33/G#2	31/F#2	30/F2	28/D#2
35/A#2	35/A#2	35/A#2	33/G#2	33/G#2	31/F#2	30/F2	28/D#2
35/A#2	35/A#2	35/A#2	33/G#2	33/G#2	31/F#2	30/F2	28/D#2
35/A#2	35/A#2						

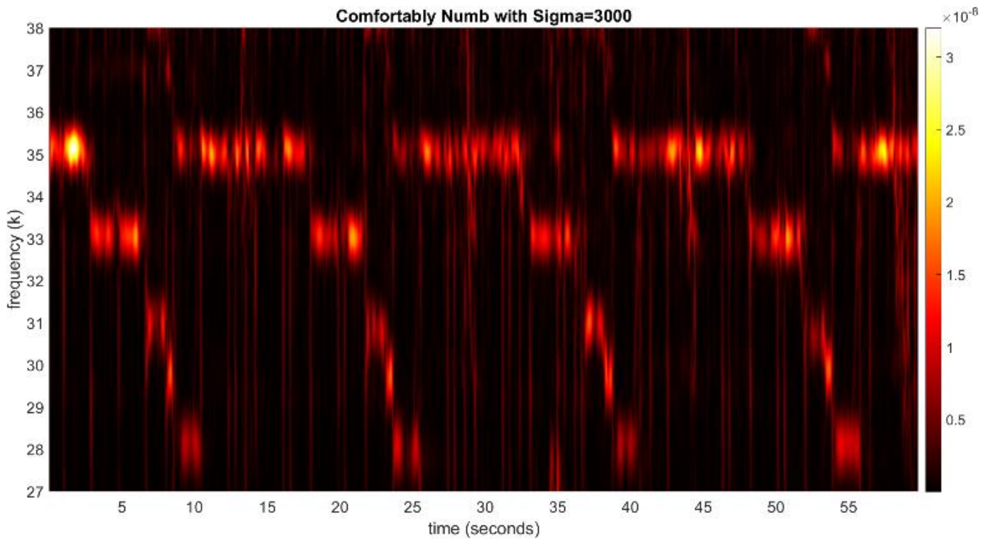

Figure 3: Spectrogram of the Gabor Transform of *Comfortably Numb*.

4.3 Comfortably Numb Lead Guitar

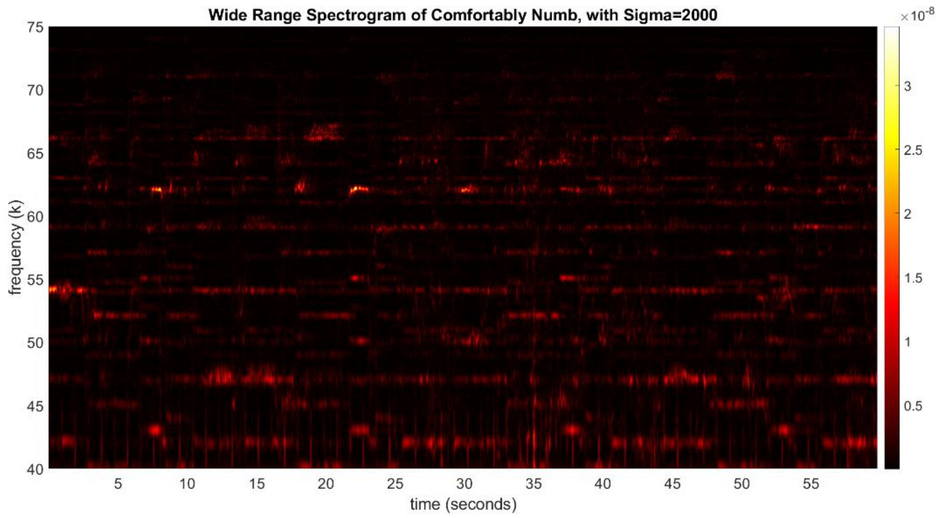

Figure 4: Wide range spectrogram of *Comfortably Numb*. There are far more possible notes detected than in the guitar spectrogram from *Sweet Child O' Mine* or the bass spectrogram.

To get an idea as to the cleanest parts of the spectrogram, it was useful to have a wide frequency view of the song first. Inspection of this spectrogram reveals the highest intensity frequencies, which will be the easiest to label as part of the lead guitarist's instrumental. This is the 62nd note, C#5. As mentioned in subsubsection 3.3.1, when the bass guitarist plays F#2, it will appear in the spectrogram as if the lead guitar is playing C#5. So, we look to figure 3 to see that F#2 is played at roughly 7.4, 22.5, 37.4, and 52.7 seconds. We cannot say whether the guitarist was playing C#5 at these times because the positive reading could be attributed to overtones. We can, however, say that the positive hits at about 17.5 seconds and 30 seconds (figure 5) are not due to the overtones of the bass guitar, because the bass does not play F#2 at these times and no other note the bass plays during the clip has overtones resembling C#5. See appendix A for the calculation of overtones for a set of notes.

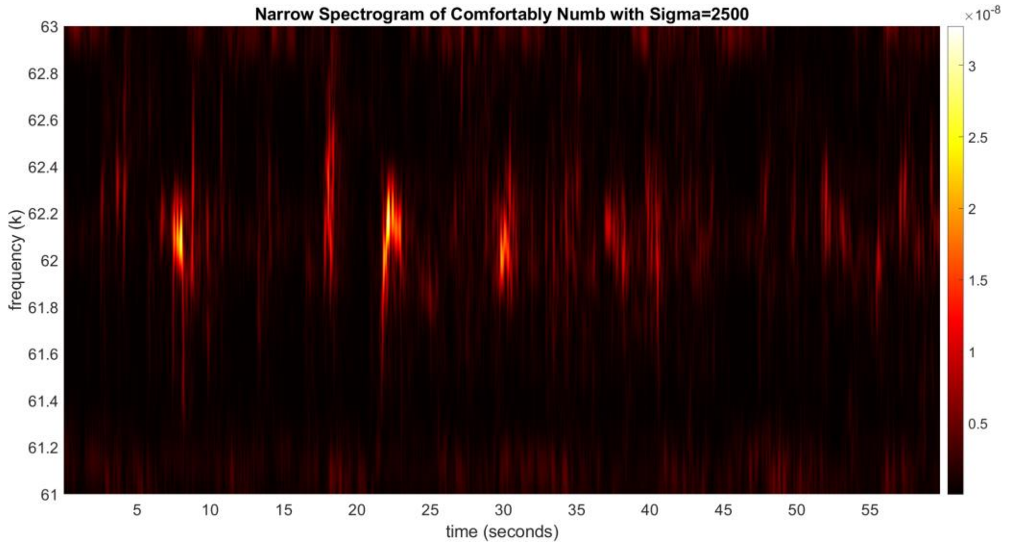

Figure 5: Spectrogram of *Comfortably Numb* zoomed into the highest intensity frequencies around 62, or C#5.

5 Summary and Conclusions

Given acoustic two clips of famous guitar instrumentals, we set out to reproduce what notes were being played and when. Using the Gabor Transform and spectrogram visualization, we were able to do so for the lead guitar in *Sweet Child O' Mine* and the bass in *Comfortably Numb*. While the lead guitar proved to be more difficult to pin down in *Comfortably Numb*, we were able to cross-reference what we knew about the bass guitar's instrumental to better understand when the lead guitarist was playing a certain note. Given enough time, we could use the same line of reasoning to determine for each possible positive detection of a note whether that is likely to be an overtone of the bass guitar note being played at that time in the clip. With that information, we could eliminate potential false positives and be left with only notes being played by the lead guitarist.

Thus, we were able to extract tab-like data on the guitar instrumentals from both clips using techniques that can be applied to similar clips that may not have easily accessible tabs online. One could even see the potential of automating this entire process to take clips of music and information about the instruments being played and outputting tabs for each instrument right then and there!

References

[1] Jose Nathan Kutz. *Data-driven modeling & scientific computation: methods for complex systems & big data*. Oxford University Press, 2013.

[2] Weisstein, Eric W. "L^2-Norm." From *MathWorld*--A Wolfram Web Resource. https://mathworld.wolfram.com/L2-Norm.html

Appendix A MATLAB Functions

- y = linspace(x1,x2,n) returns a row vector of n evenly spaced points between x1 and x2.
- decomposed = fft(signal) performs a discretized version of the Fourier transform and returns the coefficients of the frequencies present in *signal* ordered by positive indices before negative (starting with 0).
- pcolor(m, n, data) displays a spectrogram on the current figure using the vectors m and n as axis tick labels. data is an m by n matrix of Fourier decomposition coefficients.
- overtones(notes,levels) Returns a matrix of the overtones of the given vector of notes up to the levels-th overtone. Matrix returned is of size n by levels, where notes has length n.

Appendix B MATLAB Code

main.m - Guns and Roses

```
%{
02-04-2021
Shane Fretwell
AMATH 482 Assignment 2, Music Isolation
%}
%%
clear; close all;
%%

[y, Fs] = audioread('GNR.m4a'); % y := intensity, Fs := number of measurements per
second
y = transpose(y);
tr_gnr = length(y)/Fs; % record time in seconds

T = 1/Fs; % Seconds between samples
L = length(y); % Number of samples
t = (0:L-1)*T; % Time of each sample in seconds

Tau = 42; % Number of time points per second
% The Gabor Transform will be centered at each of the following points
taustep = floor(linspace(1, length(y), floor(Tau * tr_gnr)));
k = Fs*(0:(L/2))/L; % Frequencies in Hz

%% Defining note frequencies
%{
Formula for calculating these values comes from
https://pages.mtu.edu/~suits/NoteFreqCalcs.html
%}
f0 = 16.35;
% a = 2^(1/12);
% notes = f0*a.^(0:107); % Frequency in Hz of all notes from C_0 to B_8

%% Take Gabor Transform over time
% 10538 is the length required to get notes from D#3 to A5, because D#3 is
% at index 12801 in the positive frequency space, A5 at index 2263, and
% 12800-2263=10537
specf = zeros(10538, length(taustep));
sigma = 4*10^3;
a = 5*10^-5;

for tau=1:length(taustep)
    % Define the gabor filter with L-2 norm equal to one
    filter = sqrt(2/(sigma^2*pi))*exp(-0.5*(((1:L) - taustep(tau))/sigma).^2);
    tframe = y.*filter;

    freqsGT = fft(tframe);
    allfreq = abs(freqsGT/L);
    posfreq = allfreq(1:(L/2 + 1));
    posfreq(2:(end - 1)) = 2*posfreq(2:(end - 1));

    specf(:, tau) = posfreq(2263:12800); % D#3 to A5
end

%% Plot spectrogram
figure(3)
pcolor(taustep / Fs, log(k(2263:12800) / f0) / log(2^(1/12)), specf)
shading interp
set(gca,'ylim',[40 70],'Fontsize',16)
colormap(hot)
colorbar
xlabel('time (seconds)'), ylabel('frequency (k)')
```

bass.m

```
%%
clear; close all;
%%
[y, Fs] = audioread('Floyd.m4a'); % y := intensity, Fs := number of measurements
per second
y = transpose(y);
% Making the length of y even makes analysis easier without changing results in any
significant way
y = y(1:end - 1);
tr_floyd = length(y)/Fs; % record time in seconds

T = 1/Fs; % Seconds between samples
L = length(y); % Number of samples

Tau = 42; % Number of time points per second
% The Gabor Transform will be centered at each of the following points
taustep = floor(linspace(1, length(y), floor(Tau * tr_floyd)));
k = Fs*(0:(L/2))/L; % Frequencies in Hz

%%
% 4127 is the length required to get notes from D2 to C#3, because D2 is
% at index 8775 in the positive frequency space, C#3 at index 4649, and
% 8775-4649=4126
specf = zeros(4127, length(taustep));
sigma = 4*10^3;
```

```
for tau=1:length(taustep)
    % Define the gabor filter with L-2 norm equal to one
    filter = sqrt(2/(sigma^2*pi))*exp(-0.5*(((1:L) - taustep(tau))/sigma).^2);
    tframe = y.*filter;

    freqsGT = fft(tframe);
    allfreq = abs(freqsGT/L);
    posfreq = allfreq(1:(L/2 + 1));
    posfreq(2:(end - 1)) = 2*posfreq(2:(end - 1));

    specf(:, tau) = posfreq(4649:8775); % D2 to C#3
end
%}
%% Plot spectrogram
figure(3)
pcolor(taustep / Fs, log(k(4649:8775) / 16.35) / log(2^(1/12)), specf)
shading interp
set(gca,'ylim',[27 38],'Fontsize',16)
colormap(hot)
colorbar
xlabel('time (seconds)'), ylabel('frequency (k)')
```

guitar.m

```
%%
clear; close all;
%%

[y, Fs] = audioread('Floyd.m4a'); % y := intensity, Fs := number of measurements
per second
y = transpose(y);
% Making the length of y even makes analysis easier without changing results in any
significant way
y = y(1:end - 1);
tr_floyd = length(y)/Fs; % record time in seconds

T = 1/Fs; % Seconds between samples
L = length(y); % Number of samples

Tau = 42; % Number of time points per second
% The Gabor Transform will be centered at each of the following points
taustep = floor(linspace(1, length(y), floor(Tau * tr_floyd)));
k = Fs*(0:(L/2))/L; % Frequencies in Hz

%%
an = 61; bn = 63; % Note range as note numbers
af = floor(16.35*L/Fs*(2^(1/12))^an); % Indices of note range in frequency domain
bf = floor(16.35*L/Fs*(2^(1/12))^bn);
specf = zeros(bf - af + 1, length(taustep));
sigma = 2.5*10^3;

for tau=1:length(taustep)
    % Define the gabor filter with L-2 norm equal to one
    filter = sqrt(2/(sigma^2*pi))*exp(-0.5*(((1:L) - taustep(tau))/sigma).^2);
    tframe = y.*filter;
    freqsGT = fft(tframe);
    allfreq = abs(freqsGT/L);
    posfreq = allfreq(1:(L/2 + 1));
    posfreq(2:(end - 1)) = 2*posfreq(2:(end - 1));

    specf(:, tau) = posfreq(af:bf); % D2 to C#3
end
%% Plot spectrogram
figure(3)
pcolor(taustep / Fs, log(k(af:bf) / 16.35) / log(2^(1/12)), specf)
shading interp
set(gca,'ylim',[an bn],'Fontsize',16)
colormap(hot)
colorbar
xlabel('time (seconds)'), ylabel('frequency (k)')
```

overtones.m

```
function [overtones] = overtones(notes,levels)
%overtones Returns a matrix of the overtones of the given vector of notes
%up to the levels-th overtone. Matrix returned is of size n by levels, where notes
has
%length n.
A = transpose(16.35*(2^(1/12)).^notes);
B = 1:levels;
C = A*B;
overtones = log(C/16.35) / log(2^(1/12));
end
```