Allison Busa

Signals and Systems Final Project : Do Birds Like Classical Music?

Abstract

In the realm of soundscape ecology, there has been much focus on auditory effect of human-created disruptive noise on bird species. For my final project, I studied the effects of a variety of sounds on the wildlife in the Parcel B forest. I identified different species of birds, which live in Parcel B. With an audio recorder and a bluetooth speaker, I measured the audio response of birds before, during and after playing classical music. By analyzing the recordings in MATLAB, I made preliminary conclusions about how Olin birds respond to music.

Background

Soundscape ecology is the study of the sounds which can be heard in a natural environment. One of the major areas of study in this field is centered around humans' effect on the natural soundscape, the auditory "landscape" of all the sounds in one area. Over the last century, anthropogenic, or human created, sounds have become increasingly prevalent in natural areas [1]. Soundscape ecologists are still studying the overall effect that anthropogenic sounds have on wild animals. Currently, most studies show that humans affect natural soundscapes negatively, by preventing animals from listening for predators and prey, as well as finding a mate.

One of the species most affected are birds, because they rely heavily on communicating with their voices. This is made worse by the fact that anthropogenic noise tends to be in the same frequency ranges as bird songs and calls [2]. One study which found that noise pollution decreased bird pairing in a forest by 15% cited that multiple studies have estimated that range to be as large as 10 – 50% decrease in pairing [3]. Another study found that there was a lower number and biodiversity in the parts of a West Bengal forest that was adjacent to loud stone mining [4]. Other studies show that birds have been changing the duration, sound frequency range, amplitude and frequency of song or call in order to adapt to human dominated environments [5],[6],[7].

Humans, however, can connect to birds in one way: music. Bird songs are thought to be the antecessor of human song and there has been many studies conducted to determine if bird song can be defined as music [8], [9]. One study found through neuroscanning that female Sparrows' brains create the same neural pathways when listening to male Sparrows' songs as humans' brains create when listening to music [8]. This study also found that the male Sparrows' brains created pathways that were similar to humans' brains when listening to unpleasant music [8]. Based on this study, as well as prior knowledge, Timothy J. DeVoogd, a professor of psychology at Cornell University, stated,"... if a bird song sounds musical to human ears, odds are that similar human music will sound songlike to the bird," [10]

Learning Objectives

My project learning goals were to understand how each part of my project contributed to the end product, how it did it, and how that connects to what we learned in class. One way through which I reached my learning goals was designing what my analysis tools would be. This allowed me to critically examine what each part of my project was actually doing. For example, by applying the concepts which I set out to learn, I realized that some components which were in my proposal did not fit my project goal (see Diagnosis). Also by engaging with the material, I also was faced with little details about how some of the processes I used work. This helped me clarify misunderstandings which I had.

Project Description

Below I highlight the steps that I took to complete my project. A general overview of my process can be seen in the block diagram below.

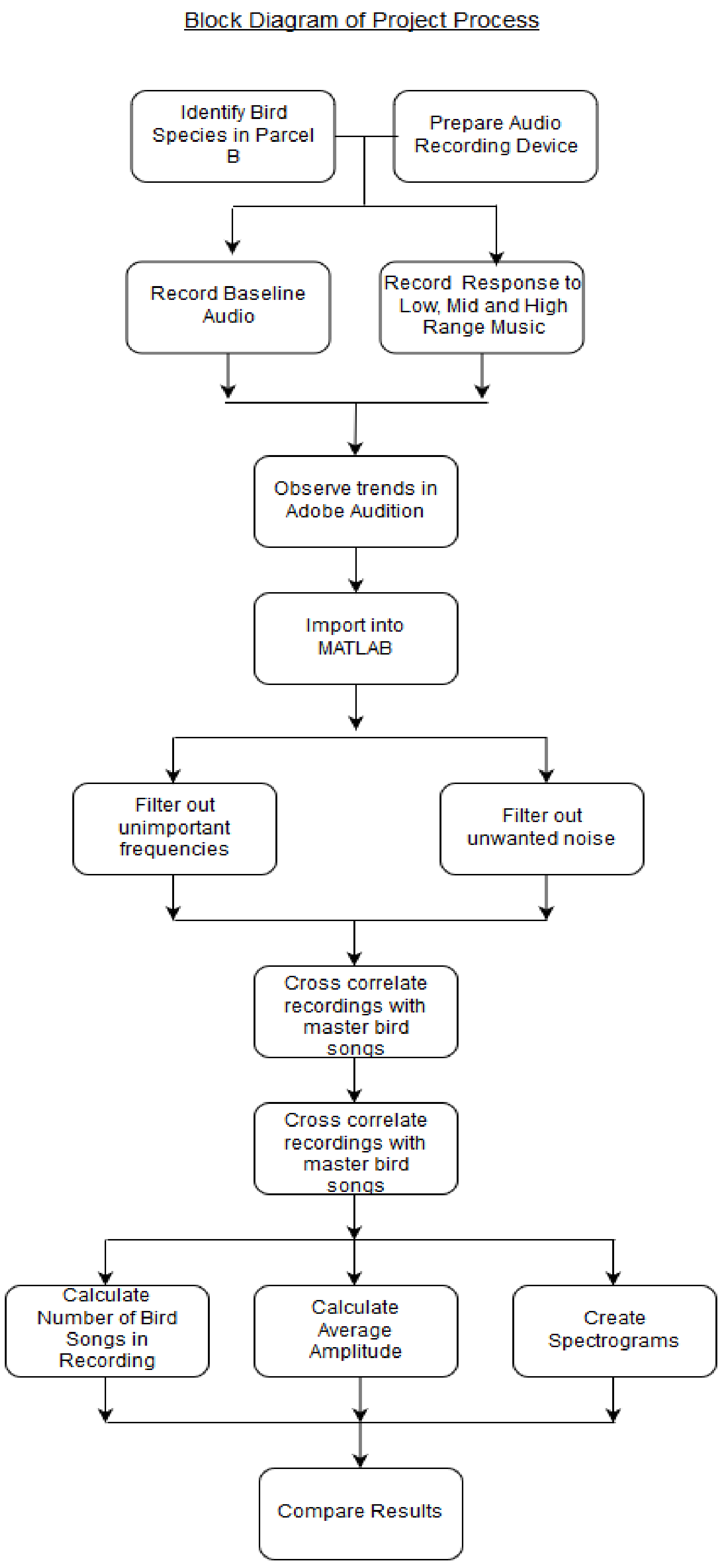

Block Diagram of Project Process
Identify Bird Species in Parcel B
Prepare Audio Recording Device
Record Baseline Audio
Record Response to Low, Mid and High Range Music
Observe trends in Adobe Audition
Import into MATLAB
Filter out unimportant frequencies
Filter out unwanted noise
Cross correlate recordings with master bird songs
Cross correlate recordings with master bird songs
Calculate Number of Bird Songs in Recording
Calculate Average Amplitude
Create Spectrograms
Compare Results

Figure 1: My General Process Diagram

I began my project by identifying birds in Parcel B and the surrounding area with Luis Zuniga. I identified birds with Luis by walking in Parcel B with cameras, taking pictures of birds we saw, and either visually identifying them on the spot or later, with the help of the pictures and the internet. Later on in the project, I identified birds by sounds with Lauren Anfenson. Lauren and I identified birds by listening to audio recordings which I took and looking at the spectrograms. Some of the birds which we found to be on Olin's campus include: Red-winged Blackbird, Song Sparrow,Chipping Sparrow, Blackbird, Chickadee, Cardinal, Goldfinch, owl, bluejay, red robin and Eastern Phoebe.

Next, I talked to David Freedman about the handheld audio recorders in the Olin Library. By weighing the information that he told me about Olin's recorders and field recorders, I chose to begin recording with Olin's Zoom H5 Handy handheld audio recorder [38]. The Zoom H5 takes all audio recordings at a sampling frequency of 44100 Hz unless stated otherwise. The word **sampling frequency** (F_s), means the number of samples taken per second. I started recording sample audio recordings of Parcel B, by finding a location in Parcel B, turning the audio recorder on and placing it on the ground and staying with it while it was recording for 10-15 minutes. I also did some initial recordings which I do not use in my analysis. These initial recordings include hour long recordings which I took with the help of Brandon Zhang and Emma Pan. In these recordings, I wait with either Brandon or Emma for about 5-10 minutes, they start playing either the violin or viola, respectively, and then I wait with the audio recorder another 30 minutes as it is recording. I also took an hour long recording in Parcel B during a Baja Drive Day. These sample recordings and initial recordings helped me tweak parameters about the recording, like the gain on the microphone, how far the audio recorder is from the sound and what to record.

My finalized set of recordings follow an experimental setup. For these recordings, I string together a set of musical compositions in a low, middle and high frequency range. In the low frequency range, I played "Bach: Solo Cello Suite No. 3 in C BWV 1009: V. Bourrée I II" four times on repeat [29]. In the mid frequency range, I played Navy Tap, Stravinsky´s trumpet solo from Petrouchka, a trumpet solo from Symphony No. 5 and a trumpet concerto [33],[34],[35],[36],[37]. In the high frequency range, I played samples of flute performances such as compositions by Tchaikovsky, Capriccio in G major and Solo Flute Concerto in A minor [30],[31],[32]. I extracted these files from the internet and sorted them into wav files based on frequency range in Adobe Premiere. I also took one reference recording, where I played music at a time when birds are not singing, and recorded the response. I conducted all four of these recordings at the grassy marsh patch adjacent to the Olin library, which I will now refer to as the marsh recordings. I also conducted recordings on high and middle frequency range recordings adjacent to the forested swamp in Parcel B, next to the footpath and Baja drive path. I will refer to these later on as the Parcel B recordings. For all of these recordings, I used a Zoom H5 audio recorder and a bluetooth speaker. I consistently placed the speaker and audio recorder down, about a foot apart. I turned the bluetooth speaker's volume to the maximum volume and connect it to my laptop. I covered both the audio recorder and speaker from the sides closely to the road, to make them look less conspicuous. I also left a note on the audio recorder which read " Audio recording in Process. Please do not disturb." I started the audio recording and stated the date and time before leaving. I went to a close by location and waited for 15 minutes, then I would turn on one of the music selections, which are all about 15 minutes long. When it ended, I would leave the recorder running anywhere between 20 to 45 minutes (sometimes I had to end early due to rain).

Once I gathered all my recordings, I moved to my analysis stage. The first part of my analysis entailed opening the recordings in Adobe Audition and inspecting them with the Spectral Frequency Display tool. The Spectral Frequency Display tool displays the spectrogram of an audio file that you input. A **spectrogram** is a graph that represents the different frequency ranges in a signal present over time (Hz), as well as how powerful each frequency is (dB/rad/sample) [12] In Audition, the Spectral Frequency Display tool can display a spectrogram for long (1+ hour) recordings and it enables the user to zoom in to see a more detailed spectrogram of a specific section. I used this tool for all of my recordings to check that there are birds in the recordings, and that they are not too faint. I also examined the quality of the recording: I checked if there was clipping or bursts of high amplitude noise. In this context, clipping refers to when a sound wave is too loud for a microphone to pick up, and the resulting recording flatlines at the maximum frequency.

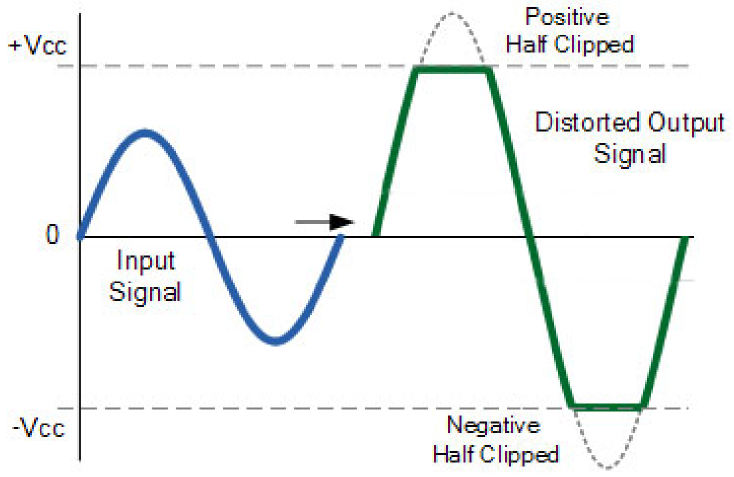


Figure 2: A visual depiction of Audio Clipping [22]

The bursts of high amplitude noise comes from the wind, which is loud and very close to the microphone. In Audition, I also extracted sections of the recording in the Spectral Frequency Display Tool and created wav files which I used as my input into matlab. I separated the recording into three separate wav files that contain the recorded audio before, during and after the music plays. I did this to keep matlab's memory from overflowing and to be able to quickly examine how bird songs changed overall from one state to the next. Lastly, I created short wav files which contained bird calls of different species from each state in the recording. The bird species which were in the recordings are Red-winged Blackbird, Song Sparrow,Chipping Sparrow, Blackbird, Chickadee, Cardinal, Goldfinch.

I completed the rest of my project in matlab. See the flow diagram below for an overview of the process and the Appendix for the raw code.

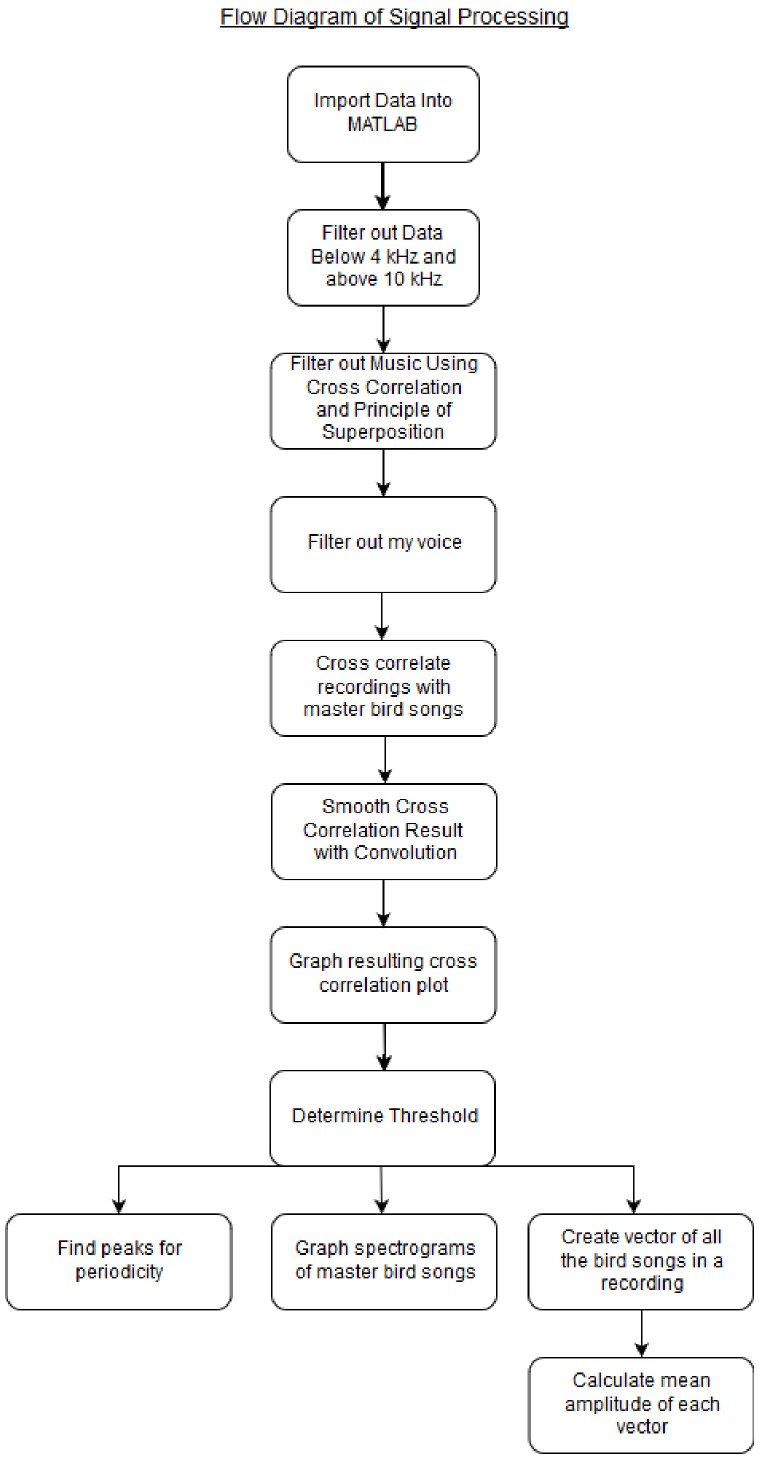

Flow Diagram of Signal Processing
Import Data Into MATLAB
Filter out Data Below 4 kHz and above 10 kHz
Filter out Music Using Cross Correlation and Principle of Superposition
Filter out my voice
Cross correlate recordings with master bird songs
Smooth Cross Correlation Result with Convolution
Graph resulting cross correlation plot
Determine Threshold
Find peaks for periodicity
Graph spectrograms of master bird songs
Create vector of all the bird songs in a recording
Calculate mean amplitude of each vector

Figure 3: A Flow Diagram Depicting the Steps I took to Execute my Code

I began with importing the files into matlab. I imported the wav file with the audioread function. Because the recordings were taken as stereo, at this point, I now convert the audio recordings to monochannel. Monochannel is another way of saying that there is only one set of data, instead of two, which come from the left and right microphones on the audio recorder. The way to convert a stereo file into mono is to take the average of the amplitudes of both channels at each point, and set that to be the new file. For this project, I reused a script that I had previously made to convert to monochannel.

Afterwards, I bandpass filter all the files between 4kHz and 10kHz, to only capture frequencies in which birds sing and chirp. In this frequency band, there are mostly bird sounds, but occasionally there is noise as well. Keeping the other frequencies would only skew my amplitude and cross correlation calculations (which I describe below). For this, I used the matlab bandpass function, which takes in parameters: the data that you want to filter, the lower and upper range of the filter (the lowest and highest frequencies which you want to keep), and the sampling frequency [23]. The sampling frequency is included to transfer from discrete time normalized angular frequency to **cyclic frequency** (frequency in Hertz). **Discrete time angular frequency** is $\Omega = 2\pi F$, where F is $\frac{f}{F_s}$: the frequency in Hertz over the sampling frequency. Frequency in Hertz is the standard frequency used for audio sampling, where Hertz = $\frac{samples}{second}$. For example, a signal is converted to the frequency domain using angular frequency and in discrete time is periodic with period 2π. In order to convert this set of values to frequencies in Hertz, we use the equation for discrete time angular frequency.$\Omega = 2\pi F$. Let's suppose that we want to convert π, which is half the maximum frequency, and, the **Nyquist rate**, the lowest rate at which a signal can be sampled without aliasing (without producing sampling errors and getting nonrepresentative results). With substitution and algebraic manipulation, we can solve for the frequency in Hertz.

$$\Omega = 2\pi \frac{f}{F_s}$$
$$\pi = 2\pi \frac{f}{F_s}$$
$$\frac{F_s}{2} = f$$

Converting to frequency in Hertz is convenient for audio projects, because it is standard to talk about the frequency of audio in Hertz. For example, most people describe the limit of human hearing to be about 22,000 Hz.

While processing the files, I noticed that the files that have music playing in them needed to be processed more. Specifically, the music needed to be filtered from the recording. For the marsh recordings, I had a file which contained the music being played in the same location without the birds singing. This recording represents the auditory output signal, y[k], caught of the marsh when different music is the input signal, x[k]. In this context, the input signal is an input going into a system, the marsh, and the system produces a unique response, an output signal. If the input is a **unit sample**, which in discrete time is a single input of height one at the origin, the system responds in a given way. The response to an impulse is called an **impulse response**. If we convolve our input signal with the impulse response, we are testing how the system reacts to each part of the input.(see description of convolution below). The output of that convolution is the output y[k]. My recording of just the music is an example of the marsh's response to an input of music, $x_1[k]$. However, my recording of the music and the birds is an example of the marsh's response to both the sounds of music and the sounds of birds $(x_1[k] + x_2[k])$. The marsh as a system is an **Linear Time Invariant system**. It is a time invariant system because no matter what day or time I take a recording, I will get the same result from the swamp. It is a linear system because it exhibits specific properties. Specifically, it abides to the principles of scaling and superposition. Scaling means that if we increased or decreased the input by a certain amount, the output would also change by that same amount. Superposition means that if our input is the sum of two signals, the system will act on each separately, and the output will be the sum of the system outputs of each. Now, in our marsh system, we have a recording that represents the added output of two inputs ($y_1[k] + y_2[k]$) and a recording that represents the output of only one of those inputs ($y_1[k]$). If we subtract one output from the sum of the two outputs, we'll end up with only the second output: $y_1[k] + y_2[k] - y_1[k] = y_2[k]$. This means, if we take the recording which shows the output of both music and bird inputs and subtract the output of the music, we will be left with the response to only the birds. This is what I had done to my recordings. But before I did the subtraction, I had to make sure that I was subtracting the output from the right places. My two recordings, the one with the music and the birds as well as the one with just the birds, were not perfectly aligned. In my two recordings there were differing amounts of time at the beginning before the music began. In order to account for this, I cross correlated my two signals with each other.

Cross correlation is given as $u(t) \otimes v(t) = \int_{-\infty}^{\infty} g(\tau) f \star (t+\tau) d\tau$. Here, $g(\tau)$ that is being multiplied by the **complex conjugate** (the imaginary term has a flipped sign) of $f(t+\tau)$ [21]. To paraphrase a lecture from the Imperial College London, the complex conjugate does not affect the computation if $f(t+\tau)$ is real valued, but it makes the definition work even if the function is complex valued [21]. The expression is being integrated for all values of τ. Note that this equation looks similar to convolution (see explanation of convolution below). Computationally they are similar: one function is flipped and is translated by t on the τ axis. In the case of cross correlation, the amount shifted is referred to as a **lag**. While convolution is used to calculate the output of an LTI system and create filters, cross correlation is used to compute the likeliness of two signals. Normally, when the highest value is assigns to 0 lags, it shows that the two signals are very well correlated as is, because it shows that when both signals are not shifted, they are most alike. If two signals are very well correlated, but one signal is offset by a few seconds, then the most prominent peak will occur at a lag that is equivalent to that offset. In the following example, I show how cross correlation shows offset.

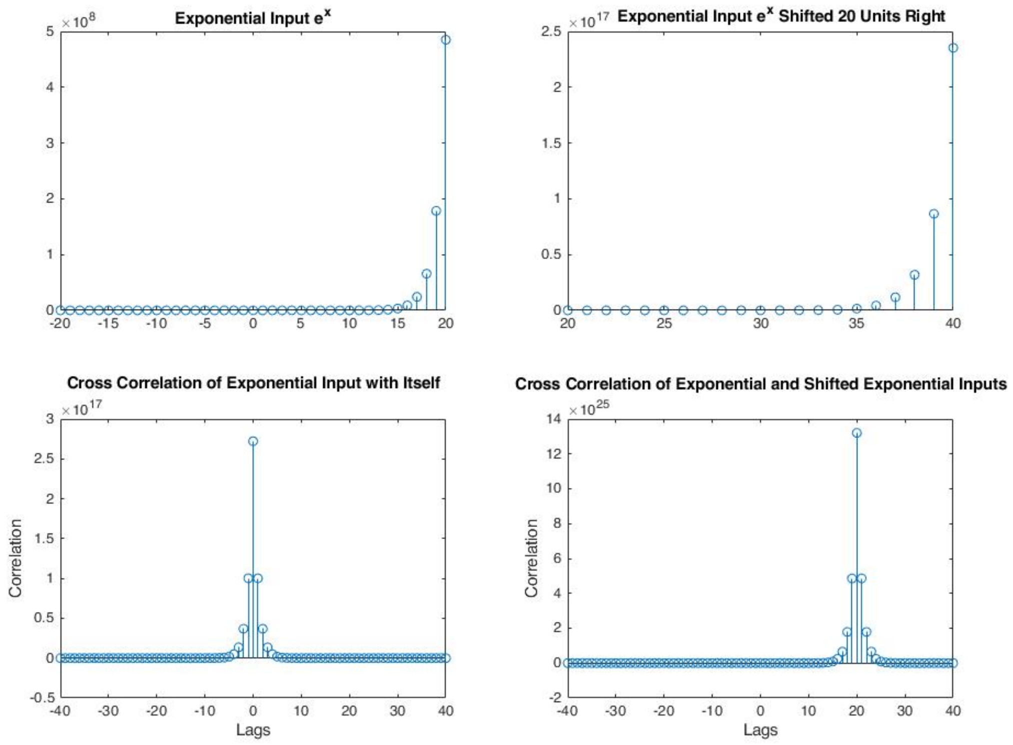


Figure 4: Example of Cross Correlation Finding a Lag

We see in the bottom left box, that if two signals are perfectly correlated, they will produce a maximum value at lag = 0. We see in the top and bottom right boxes that if one signal is shifted to the right by 20 units, the lag at which the maximum value occurs will also shift right by 20 units. I will explain the direction of the lag further later on and the code I used to create this can be found in the Appendix.

I brought these concepts and functions together to clean my recordings of the music. First, I used cross correlation to find the lags with the least time delay between the recordings. I used the matlab xcorr function which takes in the parameters two vectors, A and B, in that order. The matlab function xcorr shifts B while keeping A constant. This means that when the max correlation happens at a negative lag, B was shifted left to correlate with A and thus either A needs to be shifted right or B needs to be shifted left. The inverse is true when the lag of maximum correlation is positive: B was shifted right, so A needs to be shifted left or B needs to be shifted right. I explain this same concept visually with the graphic below:

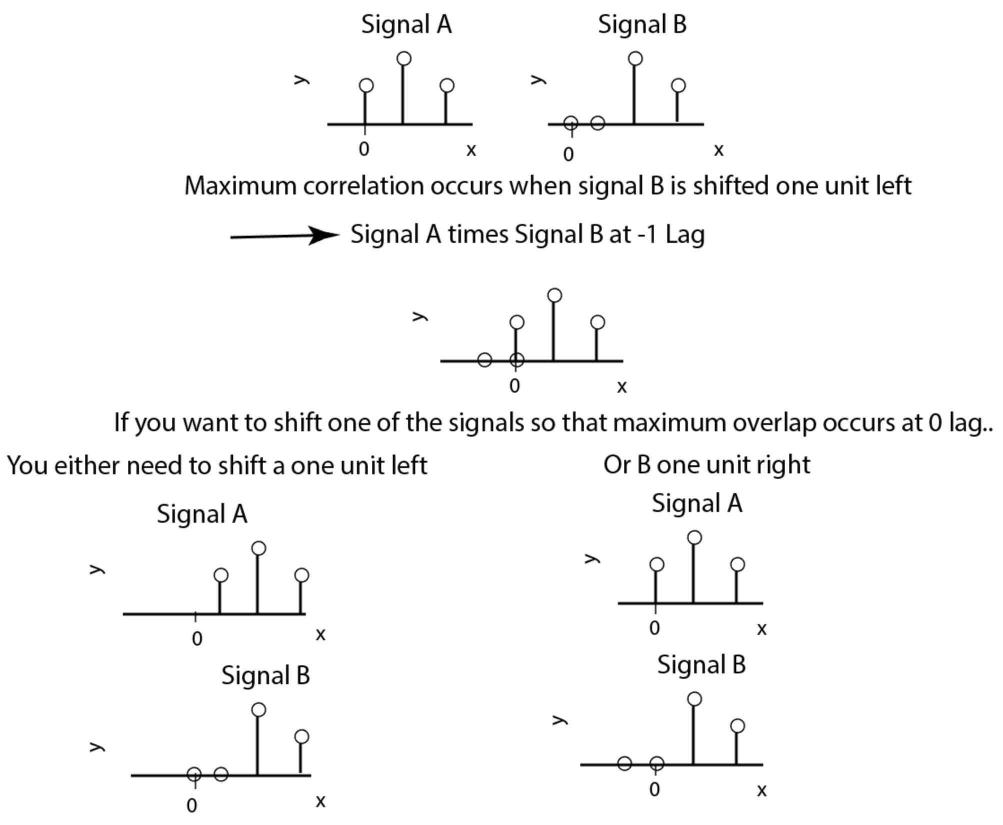

Signal A
Signal B
y
0
x
Maximum correlation occurs when signal B is shifted one unit left
Signal A times Signal B at -1 Lag
If you want to shift one of the signals so that maximum overlap occurs at 0 lag..
You either need to shift a one unit left
Or B one unit right

Figure 5: Example of Shifting to Have Maximum Overlap. Produced by Author

Once I had the lag at which maximum correlation occurs, I shifted my music and birds audio file in one of two ways. If the lag was negative, I added a vector of zeros that is the length of the lag to the beginning of the file and cut a section of the file that is the length of the vector off from the end. If the lag was positive, I cut the file to start at the lag and I add trailing zeros at the end. I add trailing zeros to either the beginning or end, because the files need to be the same length in order to subtract them. Adding the vector of zeros does not significantly affect the file because the end behavior is quieter and less important than the middle and in this scenario, because the files are fairly similar except for a few seconds delay, the lag will be small in comparison to the size of the whole file.

Lastly, I subtract the birdless recording from the shifted signal and save the result to a wav file with the matlab function audioread.

The last processing I did involved deleting the section of the recording in which I am stating the date and time. For this, I plotted a graph of the data from the beginning of the recording session. The section which I am talking is clearly discernible, because my mouth was next to the microphone and thus the amplitude of the audio recorded is high in comparison to the rest of the recording. Using the plotting tool, I find the index when I stop talking, and change by data so that it starts from that index, instead of the beginning. Below is an example of the code I used to remove my voice from the recording.

```
[swampflute_before, ~] = converttomono('file.wav');
plot(swampflute); % plot to find index
swampflute_before = swampflute_before(4718700: end); %take out the data with voice
```

After processing the data, I cross correlated each of the three wav files from one recording with the bird recordings I created. I did this because it would give me a rough estimate of many birds of one species were in a recording and where they were. As I stated earlier, cross correlation yields information about how much two signals are aligned. If they are very similar or very similar but differ by an offset, then the cross correlation will yield one large spike. However, if there is a repeating pattern in one of the signals, then there will be prominent spikes for multiple values of lags. Specifically, there will be prominent peaks at all the time steps at which the same pattern appears. In this case, if we cross correlate a bird song with a recording filled with bird songs, the prominent spikes will occur at all the times in which a bird songs in the recording. In my project, I take the absolute value of the cross correlation at this point in order to make taking the threshold easier. In this case, having strong negative correlation is just as significant as positive correlation because the sign of the cross correlation shows if two variables are directly or inversely proportional, which does not matter in my project. An example of cross correlation graph I made can be seen below.

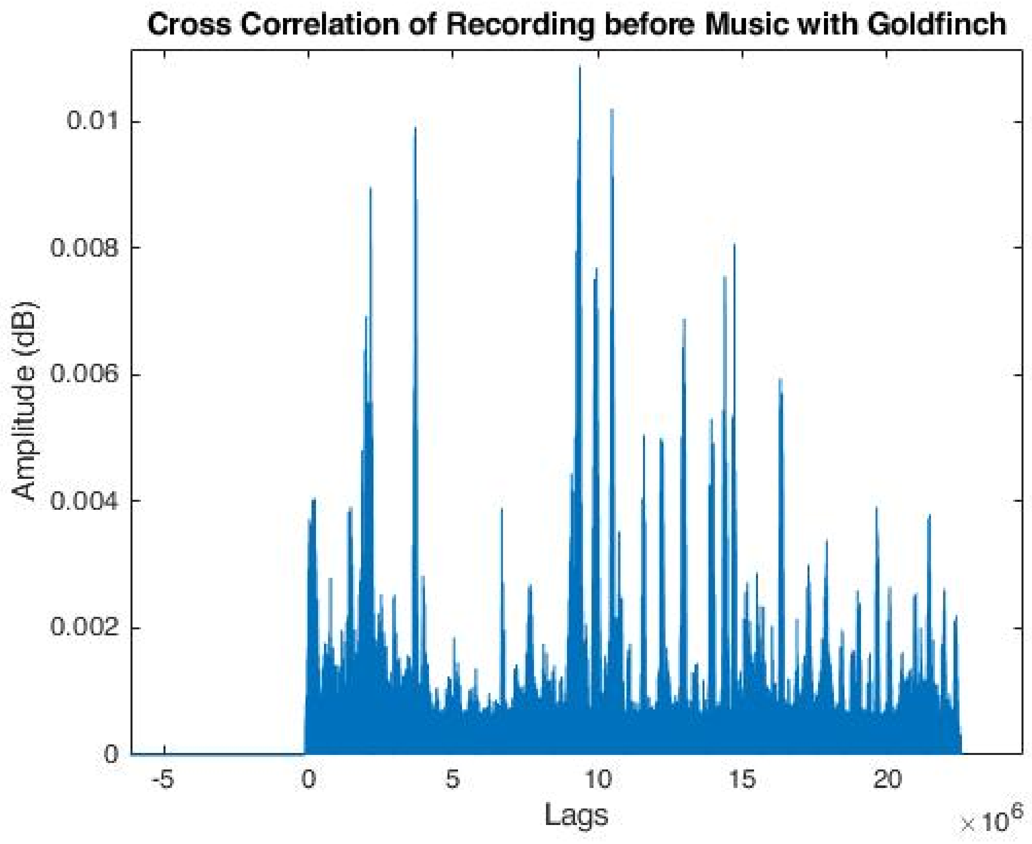


Figure 6: Cross Correlation of a Recording While the Music Plays and a Goldfinch Song

Computing only the cross correlation of a bird song and a recording will produce noisy results, however. There are little spikes in the cross correlation graph which are the result of correlation with random noise. In order to smooth out the correlation, and filter the correlations to only yield the most prominent ones,I processed the cross correlation output further with convolution. As stated above, convolution is very similar to cross correlation. In the discrete time domain (in the case of discrete data points indexed over a range of time), **convolution** is given as: $y[k] = x[k] \star h[k] = \sum_{m=-\infty}^{\infty} x[m]h[-m+k]$. One function is held constant while the other shifts in position by a value of k from negative infinity to infinity. For every shift, every value of k, we receive an output. We see by the equation that if one of the functions is zero, the product of the functions and thus the convolution are also zero. Thus, convolution only produces a non-zero result when the functions are overlapping. I convolved my cross correlation output with a box of impulses that was 5000 samples long. I chose 5000 samples, which was about a fourth of the length of a bird song, because I wanted to pick a length which would not be too small to smooth out the cross correlation spikes, and not too big that it would distort the actual prominent peaks. Another way to step through convolution in this case would be to think about taking the sum of the product of a box that is 5000 samples long and some other signal. The result will be the sum of 5000 samples of the signal multiplied by one, which is akin to averaging out the value. When the convolution summation is shifted by some k, the result will again be the sum of the 5000 adjacent samples. Recall that it is the sum of only the nearest 5000 samples because the box evaluates to zero. In order to visually see this, I have included a plot below of convolution between a box and a ramp function with spikes. The code to this plot can be found in the Appendix.

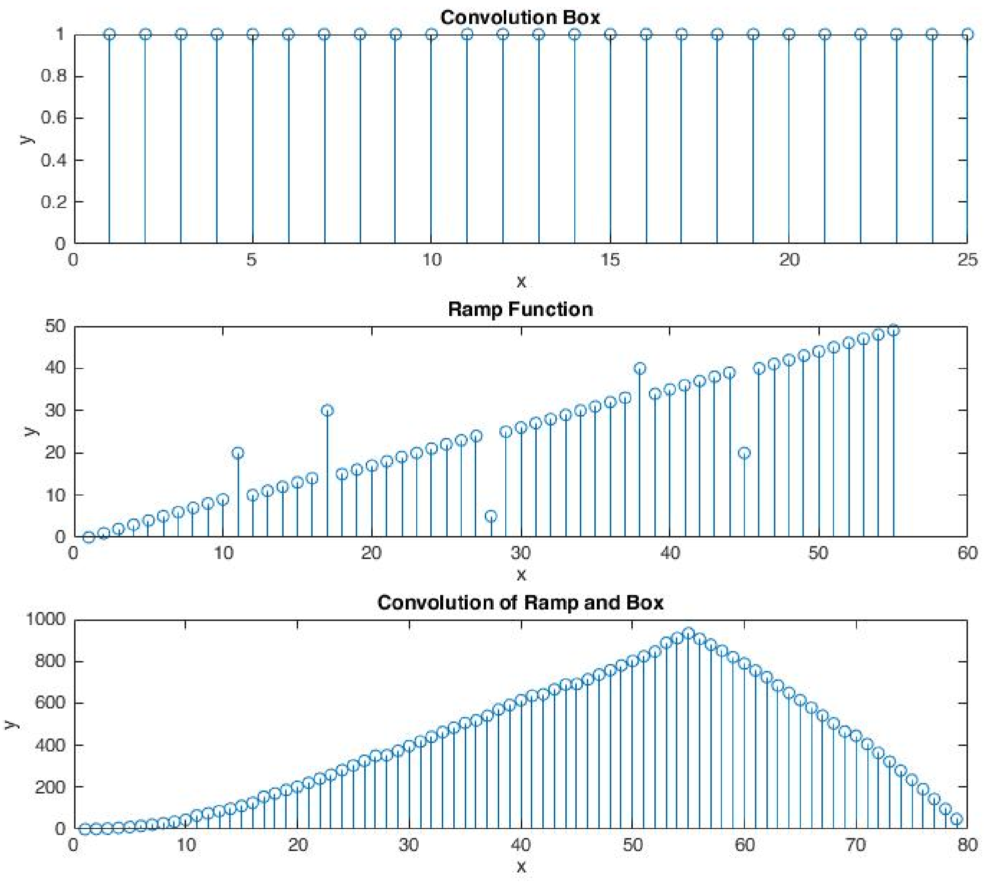
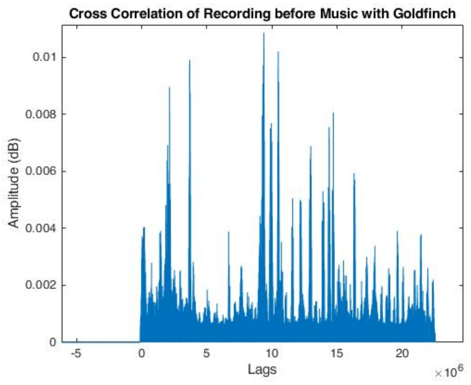


Figure 7: Example of Convolution Amplifying a Signal

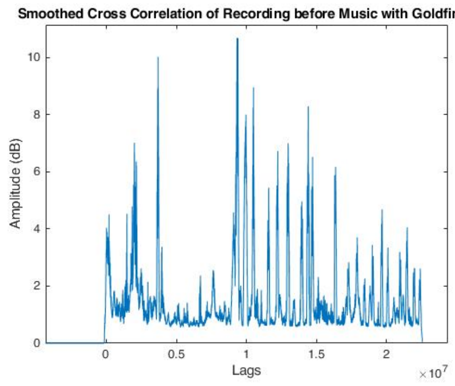
We see in the convolution output graph that the lower values are attenuated more and the higher values are increased more. The spikes in the graph are smoothed out, because they no longer represent the original value at those points on x, but the average of the points around those points, when those x values are set as the parameter [k] in convolution. Similarly, in my project, I also smooth out the spikes.

Note that convolution produces a function, or vector, whose length is the sum of the lengths of the convolved signals. In order to get my final result, I subtracted half of the box length from both sides.

A final result of the convolution smoothing on my data can be seen below. Compared to the graph I showed before, the smoothed graph looks cleaner.



(a) The Cross Correlation Graph from Above. A goldfinch song cross correlated with a recording.



(b) The same Cross Correlation graph smoothed with Convolution

An example function which I used to create one of these graphs can be seen below.

```
box = ones(5000,1); % create a box of impulses
[cardinalcc, lagscl] = xcorr(swampflute_during, cardinal);
% compute cross correlation
cardinalcc = abs(cardinalcc); %compute absolute value of cross correlation
Mcl = conv(cardinalcc, box); %convolve the box and the signal
Mcl = Mcl(2500:end-2500); %remove half of the box from the beginning and end
%plot figure
figure;
plot(lagscl, Mcl); %plot result as a function of lags
```

Having a more reliable cross correlation output enabled me to extract more information about my data. For instance, I used the output of the convolution, the smoothed cross correlation data, to create a vector which was metaphorically a string of all the instances of one bird song tied together. For this, the first thing I did was create a binary vector from the cross correlation data. A binary vector is created by setting a threshold and making everything above it one, and everything below it zero. I picked my threshold visually for each bird call, but for most of them the threshold ended up being about the same value in the context of each recording file. The exception is the chickadee call, which is one order of magnitude smaller. I suspect that this is the case because the chickadee call is very short (only two tones), making it harder to compare to other calls.

First, I used the thresholds as well as the smoothed data to find the number of prominent peaks, which roughly correlates with the number of bird songs of the same species in that recording. I used the matlab function findpeaks to automatically count the number of prominent peaks in the cross correlation graph. Findpeaks finds the local maxima in a vector and takes the input data as the only required parameter. It has a large number of optional parameters that help find the peaks, such as 'NPeaks', the maximum number of peaks to find, and 'MinPeakHeight', the minimum height which peaks can have. The parameters which I chose where 'MinPeakHeight' and 'MinPeakDistance', the minimum distance between peaks. I chose 'MinPeakHeight' as the threshold I set, so that I could pick out only the prominent peaks, and I chose 'MinPeakDistance' because the data was still very noisy and without this parameter, findpeaks would return peaks right next to each other. The 'MinPeakDistance' I chose was 50, which I found from trial and error [24]. One graph showing the result of the findpeaks function can be seen below.

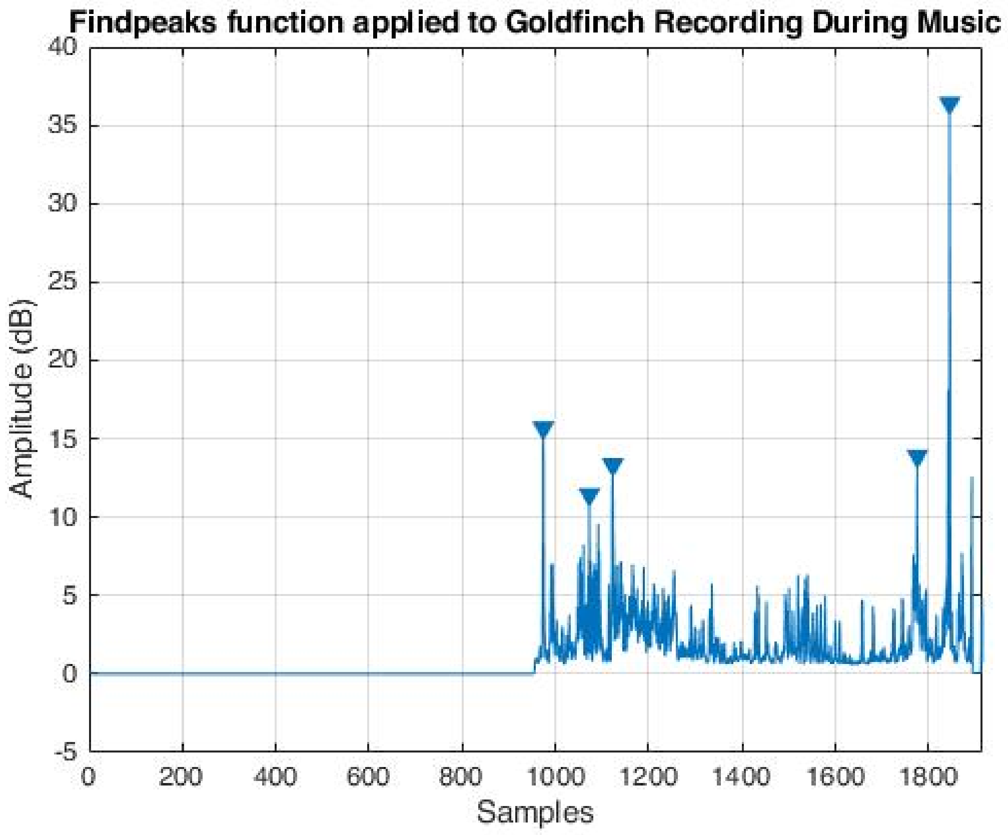


Figure 9: Example of FindPeaks Function

This is the output of the findpeaks function, which will automatically produce a graph, if an output vector is not specified. The blue triangles represent the peaks found.

After the findpeaks function, I cut the cross correlation vector to encompass only positive values of lag. The negative values of lag evaluate to zero because of matlab's implementation of cross correlation. Matlab pads shorter signals (like my master bird songs) to be the same length as the signal it is being cross correlated with. Then, I multiplied this vector by the original signal. All of the signal that goes above the threshold stays at the original value, and everything below goes to zero. I take out all the zeros in the vector to create a vector which consists of only the most prominent cross correlation results. An example of how I did this can be seen below:

```
halfwaypoint = length(swampflute_before);
```

```
% used for calculating the one sided result
cardinalthres = Mcl > 6;
%create a binary vector where every value greater than 6 is set to one
%and every value lower than 6 is set to zero
filteredcl = cardinalthres(halfwaypoint:end) .* swampflute_before;
%Because the binary vector comes from the cross correlation vector,
%it's length is the length of the %longer cross correlation input signal
%in the positive and negative directions
```

I was able to create one of these vectors for each recording part (before, during and after music) as well as for all the bird songs I decided to test for. I used these vectors by checking the average amplitude of a type of bird song in each recording part. For this, I input the vectors into matlab's mean function [25]. Because the vectors roughly consist only of a certain species' song, I should have gotten a fair estimate of the mean amplitude of the songs. Interestingly, amplitude can be both negative and positive. A microphone measures changes in pressure from an ambient state (a zero condition). A sound wave is a mechanical phenomenon that oscillates the pressure in the area. When the pressure changes, it can either be lower or higher than the ambient pressure. When the pressure is lower than the ambient pressure, the microphone reads the sound wave as being negative and when the pressure is higher than the ambient pressure, the microphone reads the sound wave as being positive [26].

The last thing that I tested was the spectrograms. For this part, I went back to Audition and extracted one recognizable bird song from each species. I imported them all into matlab and ran them through matlab's spectrogram function with the same parameters. As I stated above, a spectrogram is a graph that represents both temporal and frequency information about an audio recording. Below in Figure 10 is an example of a spectrogram created in Audition:

Annotated Spectrogram

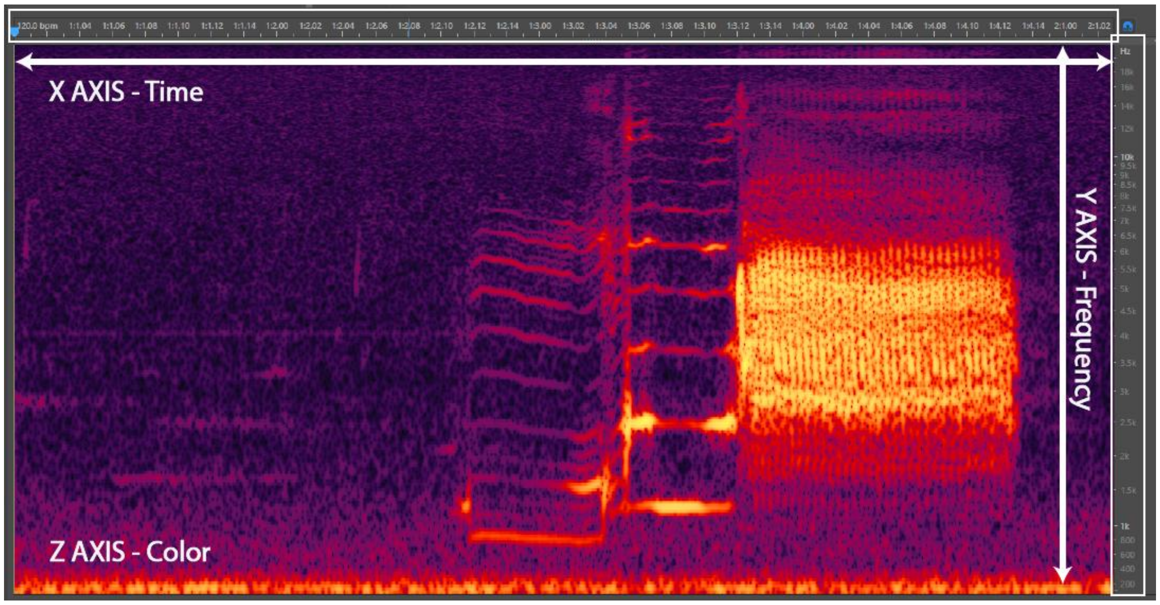


Figure 10: An Annotated Spectrogram of a Red-Winged Blackbird. Produced by Author

The core of the spectrogram function in MATLAB is the **Discrete Fourier Transform (DFT)**, the Fourier Transform for discrete frequencies [13]. It can be expressed as $X(\Omega) = \sum_{k=-\infty}^{\infty} x[k]e^{-j\Omega k}$, as we saw in class. According to this function, for a given frequency Ω, we take a discrete signal in the time domain, x[k], and multiply it by a complex exponential. We shift the position of the function x[k] as well as the angle of the complex exponential. The DFT can be used to map a function from the time domain to the frequency domain.

The Discrete Fourier Transform is the basis for the **Short-time Fourier Transform**, an algorithm in which the Discrete Fourier Transform is computed for a given interval of time, recorded and then recalculated for the following intervals of time. The reason why the Short-time Fourier Transform is calculated in this way is so the results can be brought together to show how the frequencies in a recording change over time. A spectrogram shows exactly the result of the Short-time Fourier Transform: the outputs of the Discrete Fourier Transform for all those intervals of time are put next to each other, to show how prominent frequencies change over time. The intervals in which the Discrete Fourier Transform are calculated are called **windows**, and they can be overlapping to various degrees. In fact, the degree to which windows overlap is a design choice: a smaller window will produce more accurate timing, but less accurate frequency representation. On the other hand, a larger window will be more precise in frequency representation but less precise in timing. This is due to the fact that a long interval contains more information about the signal's frequency. In order to compute a Short-time Fourier Transform, one needs to also specify the windowing function, a mathematical process that accounts for discontinuities that result from a real set of data not being composed of an integer number of cycles [13], [15].

All of these concepts come together in MATLAB's spectrogram function. The spectrogram function takes in five parameters: X, window,noverlap,f,fs . X represents the data [12]. Window represents the size in samples of the segment that the DTFT is being taken of. A large sampling window means that there is a longer time for frequency to change. Thus, a larger window means that the DTFT will be less accurate about which frequencies are actually present in that chunk of time. An example of this could be a signal which at first has a very low frequency, then a very high frequency. If you took the DTFT of the whole signal, you would see high and low frequency equally represented on the DTFT graph, but you would not know by looking at the graph if the low frequency or high frequency part came first, or even that the two frequencies don't occur at the same time. The third parameter, noverlap, represents the number of samples the programmer chooses to overlap in each window. Because the DTFT is taken on finite chunks of samples, there is the risk that the DTFT will be taken at an intersection where the frequency increases suddenly or produce inaccurate results because of another temporal shift. In order to negate this issue, the DTFT windows can overlap. For example, let's suppose there is a signal that is 201 samples in length and the window size is 100 samples. If there was no overlap, there would be one DTFT measurement from 0 to 100 samples and another from 101 to the 201. With 50 samples of overlap, the resulting measurements indices would start at 0 and go to 100 samples, then from the 50th sample to 150th sample, then from the 100th sample to the 200th sample. f represents the cyclical frequency, which describes cycles per second. In this case, specifying the cyclic frequency specifies how many frequencies are tested for. Again, this DTFT calculation happens discrete and it would be computationally heavy to find a result for many different frequencies. The number of frequencies which it will test for will be half of the number of frequencies that you give it. For example, if you specify f to be 128, then the spectrogram evaluates at $\frac{128}{2} = 65$ frequencies [12]. F_s is the sampling frequency, and it is included here for the same reason that it is included in the bandpass function (see above) [12].

After trying a few different variations, I found a matlab answers forum response which describes an algorithm which worked well for me [29]. The algorithm reads:

```
% Window duration (in seconds):
dur = 0.5;
% Spectrogram settings (in samples):
winSize = round(fs*dur);
overlap = round(winSize/2);
fftSize = winSize;
```

In this code, the windowsize, overlap and cyclical frequency are all dictated by a segment in time. The user decides how long they want their window size to be in seconds. Then, that windowsize is converted to samples. This comes from the equation

$$samples_{total} = F_s(\frac{samples}{sec}).seconds_{total}$$

where F_s is the sampling frequency, as mentioned before. The overlap is always half of the window size, which means that every point will contribute to two DTFT measurements. The cyclical frequency is the same size as the window. This is a common thing to do [12].

I went through all of this process which I describe above, for the swamp recording for high frequency music. I describe the results of the frequency and amplitude testing below.

Table of Average Amplitude for all Birds Identified at the Swamp Responding to High Frequency Music

Measurements given in dB

Bird	Before Music	During Music	After Music
Goldfinch	$2.0566*10^{-8}$	$2.2055*10^{-8}$	$-1.2463*10^{-6}$
Cardinal	$9.3424*10^{-9}$	$1.5157*10^{-7}$	$3.3320*10^{-6}$
Red-Winged Blackbird	$-1.167*10^{-8}$	$-2.9722*10^{-8}$	$6.7010*10^{-8}$
Chipping Sparrow	$-1.167*10^{-8}$	$-2.9722*10^{-8}$	$6.7010*10^{-8}$
Song Sparrow	$-1.5696*10^{-9}$	$8.2473*10^{-8}$	$-3.2899*10^{-8}$
Chickadee	$-2.0612*10^{-8}$	$-1.2155*10^{-8}$	$-1.3427*10^{-6}$

From this, we see that most amplitudes do not change significantly. Because of the small scale, even those amplitudes will change more can be written off to random noise. One interesting result is that the amplitudes for the chipping sparrow and the red-winged blackbird are exactly the same. By looking at the spectrograms, it seems that the songs for red-winged blackbirds and chipping sparrows are very similar. I suspect that my program was picking up the two songs as the same signal.

Table of Number of Bird Songs Detected for all Birds Identified at the Swamp Responding to High Frequency Music

Measurements given in number of songs detected

Bird	Before Music	During Music	After Music
Goldfinch	6	5	2
Cardinal	7	5	3
Red-Winged Blackbird	7	5	2
Chipping Sparrow	6	5	2
Song Sparrow	4	7	2
Chickadee	4	4	2

In this table, we also see that the chipping sparrow and red-winged blackbird results are the same, which validates my assumption above. There are less chickadee songs found, which was expected because the song is so short. I think these results are preliminary and not very representative of the actual number of songs. My leading reason for thinking this is because I manually checked the file of the time before the music played, and I did not find any song sparrows. My program falsely find four chickadee songs, which, compared to the maximum songs found (7) is significant.

Lastly, I imported the extracted songs from the Audition files and create spectrograms of them with matlab. Below are a set of spectrograms for before, during and after music recordings at the swamp when high frequency music was played. The rest of the spectrograms which I created can be found in the Appendix.

My first spectrogram is not zoomed in. It highlights the fact that the frequencies outside of the 4kHz to 10kHz range are attenuated. This proves that the bandpass filter worked.

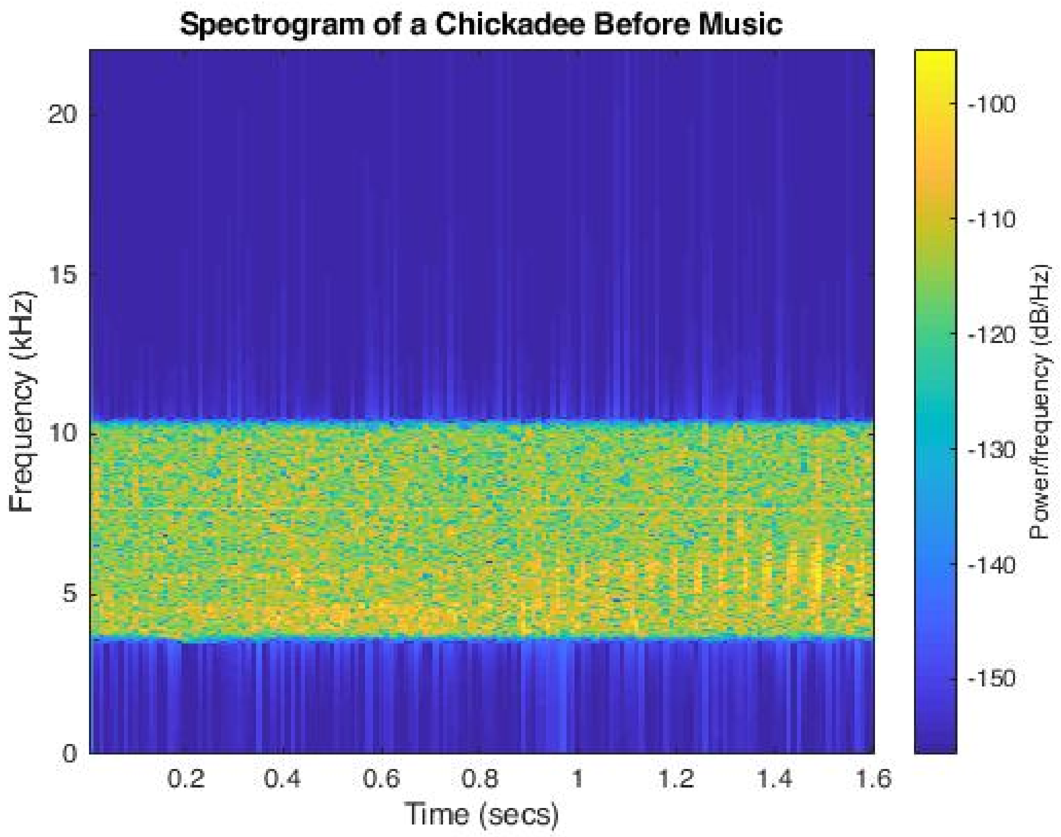


Figure 11: Unzoomed Spectrogram of Chickadee

These next three spectrograms show different bird songs of a chipping sparrow.

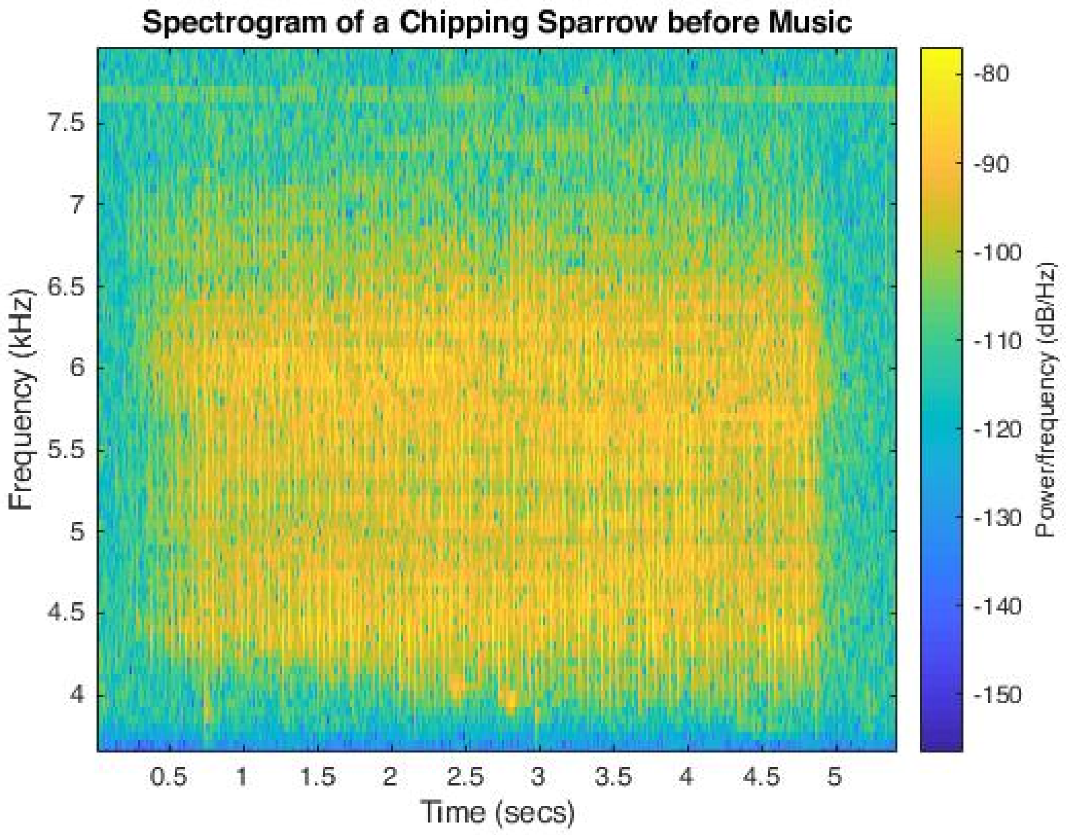


Figure 12: Spectrogram of Sparrow Before Music

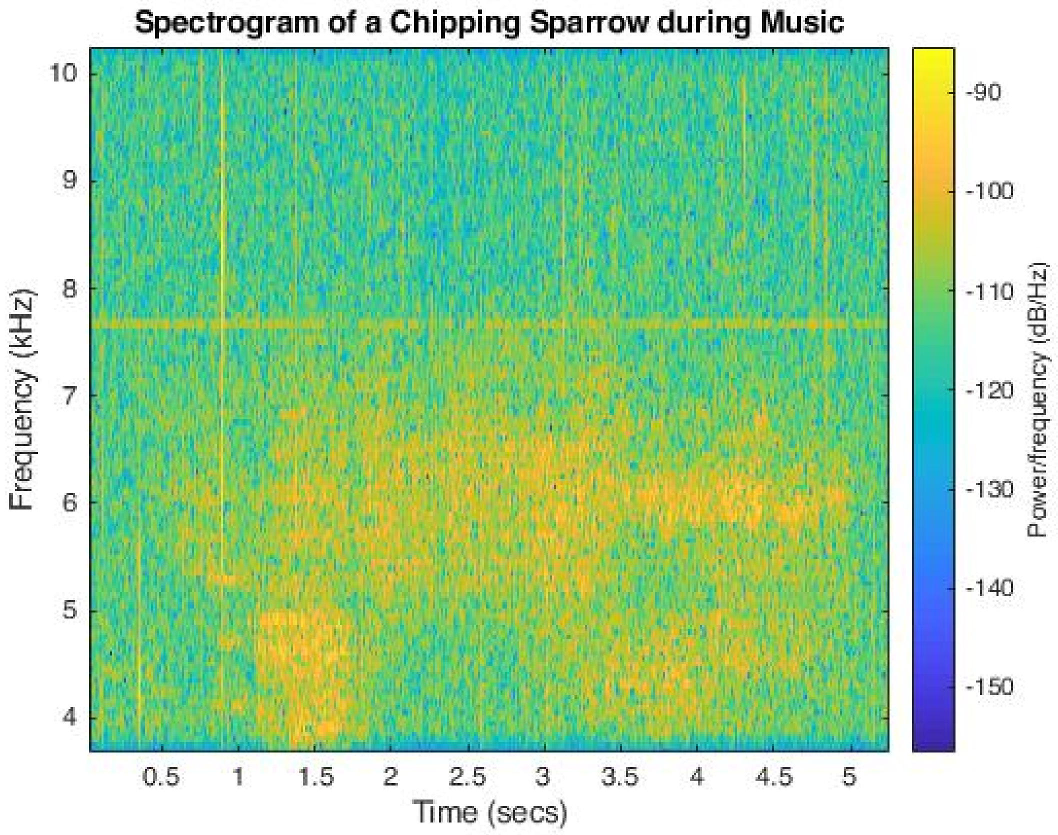


Figure 13: Spectrogram of Sparrow During Music

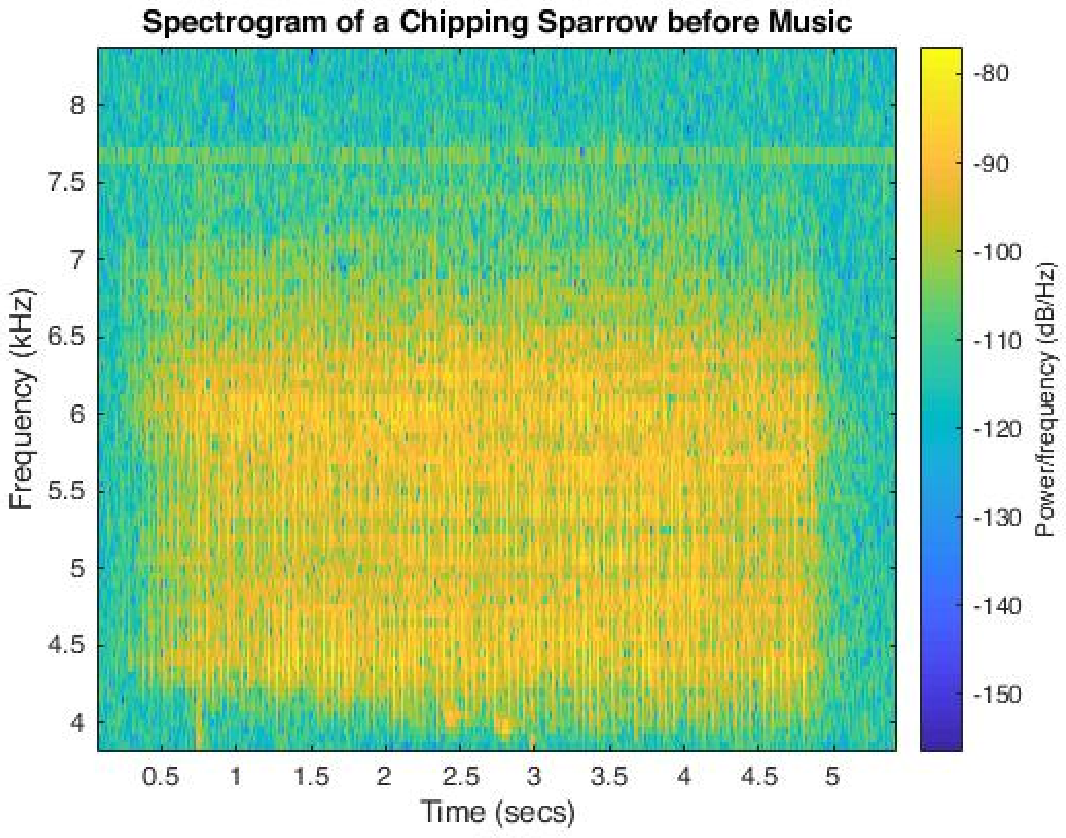


Figure 14: Spectrogram of Sparrow After Music

The before and after spectrograms are very similar, and the third is fainter and less defined. Likely, the first two spectrograms show the song of the same chipping sparrow who is a constant distance from the microphone. I suppose this because the pattern looks the same, and two different birds of the same species have personal voice, so their spectrograms would not look identical. I also saw that the bird is at a constant distance because the color of the spectrogram is consistent.

I was not able to process more data, because of my timing restraints and the long time it takes to run the analysis programs.

Diagnosis

Thinking back on what went wrong in this project, I can categorize the problems into problems with recording and problems with analysis.

There were a lot of things on the recording side which went wrong and which I learned to deal with on the spot. For one, in my original proposal, I had wanted to test the difference between live and recorded audio as well as different times of music. This proposal was very overscoped, and asking people to play was time consuming and made getting recordings harder. After talking with Diana Dabby, I managed to narrow my project down to test the response of three frequency ranges played by a speaker. Even though this project was more manageable, I was still not able to get all of the recordings analyzed. Furthermore, the ten hours of recordings that I did take were for the most part not used due to long processing time. See more about why in the Improvement section. Another problem with the recordings which I encountered was that it was hard to find a part of Parcel B in which birds were very chattery. It would often sound that a location further away was chattery, but I would walk towards it and the sound would move somewhere else. I attributed this to birds being frightened by my presence and moving away. I had also discussed this issue with Diana Dabby and she suggested that I find a way to start the recording, wait and turn the music on remotely from somewhere else. The method by which I chose to do this, by using a bluetooth speaker, was a suggestion from my classmate Daniel Connolly and I selected it because it was the device which was easiest to obtain and set up. I also ran into a response to my recordings which I was not expecting. I hadn't realized that birds would be so active while the recording was going on and that it would be valuable to extract information from the period while the music was playing. I decided to use this part of the recording in my analysis as well, by filtering out the music, as I described in the Project Description. Overall, I think this recording experience was very valuable to me in the future, even though I did generate a lot of content which I did not end up using.

The nature of the problems which I encountered on the analysis side of the project were problems which pertained to open-endedness and uncertainty. Some of the analysis which I had proposed to do originally did not make sense to do in this context, so I took it out. I had proposed algorithms that would cross correlate individual bird songs, return Pearson's coefficient of two songs, return the time in seconds of each bird song and rank length of each bird song. Functions that compared individual signals was too involved and computationally heavy for this project. This is because there were too many bird calls to extract by hand. In order to extract all the bird calls, I would have to do so computationally, which leads to problems like not all of the songs being recognized and sounds that are not a match being recognized. I would also have to dynamically create vectors for recognized sounds, which is an algorithm that would take me a substantial amount of time to learn. Because I was no longer measuring duration, I did not need to create a vector of the time in seconds at which each datapoint was taken. I had also proposed algorithms that I realized by working on the project would not yield substantially more information. I took these out as well. The DFT plot I proposed would not yield more information because frequency domain information is embedded in spectrograms already. The Pearson's coefficient would also not be particularly useful on short signals, because it is a function which works better when there is more data. I believe that there is also a lot on the analysis side which could be improved, but going through this process helped me learn about the scale of these types of projects and the importance of incremental testing.

Improvement

The roads I see to improvement are also distinct for recording and analysis.

For recordings, I would want a longer time scale and access to better resources. Doing this experiment over a couple of months would give better results. For one thing, there is a lot of variability which could be made clearer with more data.For example, a bird can change its song over the course of the recording because of other factors like time of day. For another thing, it would be easier to take the recordings of only the music and no animal sounds in the dead of winter. Also, having a really expensive recorder that has a windscreen and that you can leave out in the woods would decrease the time needed to spend and also would allow me to get better sound quality.

For analysis, I believe with more time, I could make improvement to the process as well as the components themselves. As I mentioned before, incremental testing would help me gauge how attainable certain analysis goals are. Also, focusing on making functions time efficient and automated would allow me to get more analysis done. Things that could be automated include naming the functions correctly and choosing a threshold for the prominent cross correlated outputs of recordings and master bird songs. Now, it takes about twenty minutes to get amplitude, frequency and spectrograms for each section of a recording session. This includes opening the file, extracting and saving the bird songs. If I could automate these two things, then I could have used a lot more wrapper functions, and my code would be a lot easier to understand and use. One idea which was reinforced for me through doing this project is that data from the natural world is messy and irregular. The functions and processes that are applied to study the natural world should be dynamic to deal with the irregularity. I believe this because I found in this project that some of the functions that I did use did not work as effectively as was needed. Cross correlating a long file with a master bird song identifies places where there is another bird call of the same species about half of the time. Frequently, the prominent peaks in these cross correlations are bird songs of different species. It makes sense that this is happening because, compared to white noise, sounds with a similar amplitude and number of oscillations would be more highly correlated. Similarly, the findpeaks algorithm doesn't find all the peaks above a threshold, because the data is noisy and the peaks are spaced irregularly.

Also, I learned that matlab has trouble dealing with the quantity of data that I gave it. At first, I had imagined that I would split the data into three wav files in matlab. However, I immediately got errors about memory when I tried applying any function to it. In the future, I would like to bring back some of the functions which I had to remove from this project. I would like to bring back functions which compare multiple bird songs, because I believe that it is best to test if a bird song has changed by comparing the results directly. In order to do this, I would need to create a function that finds the birds and creates vectors dynamically as well as improve the algorithm which finds bird songs One starting step to accomplishing the latter goal could be cross correlating the signal with more master bird songs of the same species, and taking the average of the cross correlations.

If I have the chance, I would like to improve and iterate on this project in the future. However, if I go about doing that, I would plan it to be on a longer time scale and for a larger time commitment.

Collaboration

I worked by myself on this project, but as I mentioned earlier in my paper, I had help from a variety of people. These people are Lauren Anfenson, Luis Zuniga, David Freedman, Diana Dabby and Joseph Lee.

References

[1] B.C. Pijanowski, A. Farina, S.H. Gage, S.L. Dumyahn, and B.L. Krause, “What is soundscape ecology? An introduction and overview of an emerging new science,” *Springer Link*, vol.26, Nov. 2011.

[2] M. Brittingham. “Noise impacts to wildlife – a review of pertinent studies, Pennslyvania State University Department of Ecosystem Science and Management”, [Online]. Available: http://www.docs.dcnr.pa.gov/cs/groups/public/ documents/document/dcnr_20028837.pdf

[3] L. Habib, E.M. Bayne and S. Boutin, “Chronic industrial noise affects pairing success and age structure of ovenbirds Seiurus aurocapilla,” *Journal of Applied Ecology*, vol.44, no.1, pp. 176-184, Feb. 2007.

[4] D.C. Saha and P.K. Padhy, “Effect of air and noise pollution on species diversity and population density of forest birds at Lalpahari, West Bengal, India” *Science of the Total Environment*, vol. 409, no. 24, pp. 5328-5336, Nov. 2011

[5] C.D. Francis, C.P. Ortega and A. Cruz. “Different behavioural responses to anthropogenic noise by two closely related passerine birds” *Biology Letters*, vol. 7, no. 6, May 2011.

[6] F.E. Rheindt. “The impact of roads on birds: Does song frequency play a role in determining susceptibility to noise pollution?”, *Journal für Ornithologie*, vol. 144, no. 3, pp. 295-306, July 2003.

[7] C.E. Ortega. “Effects of Noise Pollution on Birds: A Brief Review of Our Knowledge”, *Ornithological Monographs*, vol. 74, no. 1, pp. 6-22, July 2012.

[8] S.E. Earp and D.L.Maney, “Birdsong: is it music to their ears?”, *frontiers in Evolutionary Neuroscience*, 2012.

[9] C. Hartshorne, "THE RELATION OF BIRD SONG TO MUSIC*," International Journal of Avian Science*, vol.100, no.3, pp. 421-445, July 1958.

[10] C.C. Ray. (2017, June, 19), Do Birds Listen When You Play Music? [Online]. Available: https://www.nytimes.com/2017/06/19/science/do-birds-listen-when-you-play-music.html

[11] [Online]. Available: https://www.xeno-canto.org/

[12] Mathworks. "Spectrogram" [Online]. Available:https://www.mathworks.com/help/signal/ref/spectrogram.html

[13] CSCE 666 Pattern Analysis : Fourier analysis. [Online]. Available: http://research.cs.tamu.edu/prism/lectures/pr/pr_l29.pdf

[14] Mrinal Mandal and Amir Asif. *Continuous and Discrete Time Signals and Systems*. Cambridge, UK: Cambridge University Press, 2007.

[15] CSCE 666 Pattern Analysis :Short-time Fourier analysis and synthesis. [Online]. Available: http://research.cs.tamu.edu/prism/lectures/sp/l6.pdf

[16] Wolfram MathWorld. "Cross Correlation". [Online]. Available: http://mathworld.wolfram.com/Cross-Correlation.html

[17] Mathworks. "Xcorr". [Online] Available: https://www.mathworks.com/help/matlab/ref/xcorr.html#mw_ff426c84-793b-4341-86f0-077eba46eb22

[18] DSP Guide. "Chapter 12: The Fast Fourier Transform". [Online]. Available:https://www.dspguide.com/ch12.htm

[19] L. Bodis. "Quantification of spectral similarity towards automatic spectra verification," PhD. thesis, ETH Zurich, Zurich, DE, 2007.

[20] Mathworks. “corrcoef”. [Online] Available: https://www.mathworks.com/help/matlab/ref/xcorr.html;jsessionid=7971df4f4133 444521e662d5cd51#mw_ff426c84-793b-4341-86f0-077eba46eb22

[21] Imperial College London. “8: Correlation”. [Online] Available: http://www.ee.ic.ac.uk/hp/staff/dmb/courses/e1fourier/00800_correlation.pdf

[22] Wade Desouza. “What is Distortion”. [Online] Available: https://wadedesouza.wordpress.com/2016/05/04/tube-distortion/

[23] Mathworks. “bandpass”. [Online] Available: https://www.mathworks.com/help/signal/ref/bandpass.html

[24] Mathworks. “findpeaks”. [Online] Available: https://www.mathworks.com/help/signal/ref/findpeaks.html

[25] Mathworks. “mean”. [Online] Available: https://www.mathworks.com/help/matlab/ref/mean.html

[26] StackOverflow. “Meaning of negative values in audio waveforms”.[Online] Available: https://stackoverflow.com/questions/1380692/meaning-of-negative-values-in-audio-waveforms

[27] Mathworks. “spectrogram”. [Online] Available: https://www.mathworks.com/help/signal/ref/spectrogram.html

[28] MATLAB Answers. “High quality spectrogram with a few seconds signal”. [Online] Available: https://www.mathworks.com/matlabcentral/answers/267003-high-quality-spectrogram-with-a-few-seconds-signal

[29] BRS Music. (2015, Nov. 1). JS Bach: Solo Cello Suite No. 3 in C BWV 1009: V. Bourrée I & II (& I again) [Video file]. Available: https://www.youtube.com/watch?v=6uRfSAytVYI

[30] Alicanto Studio. (2013, Sep. 24). Eugène Oneguine" paraphrase, Tchaikovsky - Emmanuel Pahud (flûte & piano) [Video file]. Available: https://www.youtube.com/watch?v=HsUB38k_nmU&list=RDYEyKM13yf_4&index=2

[31] AbsolutelyPahud. (2013, Oct. 9). EMMANUEL PAHUD | J.J. Quantz - Capriccio in G major [Video file]. Available: https://www.youtube.com/watch?v=rQMWi2nGWLA&list=RDYEyKM13yf_4&index=12

[32] P. Bugarchich. (2012, May 12). C.P.E. Bach ~ Solo Flute Concerto in A minor H 56 (Jean Pierre Rampal) [Video file]. Available: https://www.youtube.com/watch?v=ULJjNpHun9g

[33] Unites States Navy Band. (2012, May 25). Taps [Video file]. Available: https://www.youtube.com/watch?v=WChTqYlDjtI

[34] Play With A Pro Music Academy. (2016, Aug. 6). David Bilger plays Stravinsky´s trumpet solo from Petrouchka [Video file]. Available:https://www.youtube.com/watch?v=sm7rDpWDhLI

[35] Boston Philharmonic. (2018, Mar. 10). Interpretation Class: Mahler - Trumpet Solo from Symphony No. 5 [Video file]. Available: https://www.youtube.com/watch?v=gPzDh2ypK1U

[36] P. Reiser. (2017, Dec. 10). Joseph Haydn Trumpet Concerto in Eb, 3rd movement - Staatskapelle Dresden - Christian Thielemann [Video file]. Available: https://www.youtube.com/watch?v=s8Obtk0HZmM

[37] Boston Philharmonic. (2015, Apr. 24). Boston Philharmonic Youth Orchestra: Hummel - Trumpet Concerto (feat. Elmer Churampi, trumpet) [Video file]. Available: https://www.youtube.com/watch?v=Jhz3Mo1kA4M

[38] Zoom. “The Zoom H5”. [Online]. Available: https://www.zoom-na.com/products/field-video-recording/field-recording/zoom-h5-handy-recorder

Appendix 1 : Final Code

My file that converts stereo to mono

function [datamono, Fs] = converttomono(filename) if nargin == 2 [y,Fs] = audioread(filename, 'native'); else [y, Fs] = audioread(filename); end if size(y,2) > 1 [,n] = size(abs(y)); datamono = sum(y, n) / size(y, n); else datamono = y; end datamono = bandpass(datamono, [4000 10000],Fs); datamono = bandpass(datamono, [4000 10000],Fs); end

My file that calculates all the functions for a given recording

The code is the same for the other recordings, but I copied and pasted them to other m files so that I could change the threshold values and the variable names. This is a problem that I mentioned in my Improvement section.

```
% Loading all of the signals

[swampflute_after, ~] = converttomono('...\longswamprecording_after.wav');
%%
[cardinal, ~] = converttomono('...\cardinal.wav');
[goldfinch, ~] = converttomono('...\goldfinch.wav');
[chickadee, ~] = converttomono('...\chickadee.wav');
[sparrow, ~] = converttomono('...\sparrow.wav');
[rwblackbird, ~] = converttomono('...\rwblackbird.wav');
[songsparrow, ~] = converttomono('...\songsparrow_after.wav');
%%
% Taking the cross correlation of the signal with the master bird calls
box = ones(5000,1);

[cardinalcc, lagscl] = xcorr(swampflute_after, cardinal);
cardinalcc = abs(cardinalcc);
```

```
Mcl = conv(cardinalcc, box);
Mcl = Mcl(2500:end-2500);
figure;
plot(lagscl, Mcl);
title('Cross Correlation of Recording after Music with Cardinal');
xlabel('lags');

[goldfinchcc, lagsgf] = xcorr(swampflute_after, goldfinch);
goldfinchcc = abs(goldfinchcc);
Mgf = conv(goldfinchcc, box);
Mgf =  Mgf(2500:end-2500);
figure;
plot(lagsgf, Mgf);
title('  Cross Correlation of Recording after Music with Goldfinch');
xlabel('lags');

[chickadeecc, lagscd] = xcorr(swampflute_after, chickadee);
chickadeecc = abs(chickadeecc);
% Mcd = movmean(chickadeecc,6000);
Mcd = conv(chickadeecc, box);
Mcd =  Mcd(2500:end-2500);
figure;
plot(lagscd, Mcd);
title('  Cross Correlation of Recording after Music with Chickadee');
xlabel('lags');

[sparrowcc, lagssp] = xcorr(swampflute_after, sparrow);
sparrowcc = abs(sparrowcc);
```

```
Msp = conv(sparrowcc, box);
Msp =  Msp(2500:end-2500);
% Msp = movmean(sparrowcc,6000);
figure;
plot(lagssp, Msp);
title('  Cross Correlation of Recording after Music with Sparrow');
xlabel('lags');

[rwbbcc, lagsrw] = xcorr(swampflute_after, rwblackbird);
rwbbcc = abs(rwbbcc);
% Mrw = movmean(rwbbcc,6000);
Mrw = conv(rwbbcc, box);
Mrw = Mrw(2500:end-2500);
figure;
plot(lagsrw, Mrw);
title('  Cross Correlation of Recording after Music with Red-WingedBlackBird');
xlabel('lags');

[rwbbss, lagsss] = xcorr(swampflute_after, songsparrow);
rwbbss = abs(rwbbss);
Mss = conv(rwbbss, box);
Mss = Mss(2500:end-2500);
figure;
plot(lagsss, Mss);
title('Cross Correlation of Recording Before Music with Song Sparrow');
xlabel('lags');
```

```
%%
halfwaypoint = length(swampflute_after); % used for calculating the one sided result

%% Extracting all the sounds above a correlation threshold
cardinalthres = Mcl > 90;
filteredcl = cardinalthres(halfwaypoint:end) .* swampflute_after;
onlyfilteredcl = filteredcl(filteredcl ~= 0);
% figure;plot(filteredcl);
% xlabel('lags');
% ylabel('amplitude');
% title('Binary Vector of Filtered Cardinal Song');
% audiowrite('...\cardinalfilteredagain.wav', filteredcl, 44100)
%%
goldfinchthres =Mgf >90;
filteredgf = goldfinchthres(halfwaypoint:end).* (swampflute_after);
onlyfilteredgf = filteredgf(filteredgf ~= 0);
% audiowrite('...\goldfinchfilteredagain.wav', filtered4, 44100)
%%
chickadeethres = Mcd > 9; %the correlation for this song is one order of
%magnitude smaller
filteredcd = chickadeethres(halfwaypoint:end) .* swampflute_after;
onlyfilteredcd = filteredcd(filteredcd ~= 0);
figure;plot(filteredcd);
% audiowrite('...\chickadeefilteredagain.wav', filtered5, 44100)
%%

sparrowthres = Msp > 100;
filteredsp = sparrowthres(halfwaypoint:end) .* swampflute_after;
```

```
onlyfilteredsp = filteredsp(filteredsp ~= 0);
figure;plot(filteredsp);
% audiowrite('...\sparrowfilteredagain.wav', filteredsp, 44100)
%%
rwbbthres = Mrw > 100;
filteredrw = sparrowthres(halfwaypoint:end) .* swampflute_after;
onlyfilteredrw = filteredrw(filteredrw ~= 0);
figure;plot(filteredrw);
% audiowrite('...\rwbbfilteredagain.wav', filteredrw, 44100)
%%
ssthres = Mss > 4000;
filteredss = ssthres(halfwaypoint:end) .* swampflute_after;
onlyfilteredss = filteredss(filteredss ~= 0);
% figure;plot(filteredss);
% audiowrite('...\ssfilteredagain.wav', filteredss, 44100)
%% Taking the average of the extracted sounds
goldfinch_after_average = mean(onlyfilteredgf);
chickadee_after_average = mean(onlyfilteredcd);
cardinal_after_average = mean(onlyfilteredcl);
rwbb_after_average = mean(onlyfilteredrw);
sparrow_after_average = mean(onlyfilteredsp);
ss_before_average = mean(onlyfilteredss);
%% Checking the peaks and number of peaks
pks_goldfinch = findpeaks(Mgf,44100,'MinPeakHeight',90, 'MinPeakDistance', 50);
pks_chickadee = findpeaks(Mcd,44100,'MinPeakHeight',9, 'MinPeakDistance', 50);
%changed the order of magnitude here too
pks_cardinal = findpeaks(Mcl,44100,'MinPeakHeight',90, 'MinPeakDistance', 50);
pks_rwbb = findpeaks(Mrw,44100,'MinPeakHeight',100, 'MinPeakDistance', 50);
```

```
pks_sparrow = findpeaks(Msp,44100,'MinPeakHeight',100, 'MinPeakDistance', 50);
pks_ss = findpeaks(Mss,44100,'MinPeakHeight',4000, 'MinPeakDistance', 50);

%%
len_pks_goldfinch = length(pks_goldfinch);
len_pks_chickadee = length(pks_chickadee);
len_pks_cardinal = length(pks_cardinal);
len_pks_rwbb = length(pks_rwbb);
len_pks_sparrow = length(pks_sparrow);
len_pks_ss = length(pks_ss);
```

File to Create Spectrograms

```
%Loading the Files
[cardinalB, ~] = converttomono('...\cardinal.wav');

[goldfinchB, ~] = converttomono('...\goldfinch.wav');
[goldfinchA, ~] = converttomono('...\goldfinch_after.wav');

[chickadeeB, ~] = converttomono('...\chickadee.wav');

[sparrowB, ~] = converttomono('...\sparrow.wav');
[sparrowA, ~] = converttomono('...\sparrow_after.wav');
[sparrowD, ~] = converttomono('...\sparrow_during.wav');

[rwblackbirdB, ~] = converttomono('...\rwblackbird.wav');
[rwblackbirdA, ~] = converttomono('...\rwbb_after.wav');
[rwblackbirdD, ~] = converttomono('...\rwbb_during.wav');
```

```
[songsparrowA, ~] = converttomono('...\songsparrow_after.wav');
[songsparrowD, ~] = converttomono('...\songsparrow_during.wav');

%%

% Window duration (in seconds):
dur = 0.02;
% Spectrogram settings (in samples):
winSize = round(44100*dur);
overlap = round(winSize/2);
fftSize = winSize;
% Plot the spectrogram:
figure;spectrogram(sparrowB,winSize,overlap,fftSize,44100,'yaxis');
title('Spectrogram of a Chipping Sparrow before Music');
figure;spectrogram(sparrowA,winSize,overlap,fftSize,44100,'yaxis');
title('Spectrogram of a Chipping Sparrow after Music');
figure;spectrogram(sparrowD,winSize,overlap,fftSize,44100,'yaxis');
title('Spectrogram of a Chipping Sparrow during Music');

figure;spectrogram(cardinalB,winSize,overlap,fftSize,44100,'yaxis');
title('Spectrogram of a Cardinal after Music');

figure;spectrogram(goldfinchB,winSize,overlap,fftSize,44100,'yaxis');
title('Spectrogram of a Goldfinch before Music');
figure;spectrogram(goldfinchA,winSize,overlap,fftSize,44100,'yaxis');
title('Spectrogram of a Goldfinch after Music');

figure;spectrogram(chickadeeB,winSize,overlap,fftSize,44100,'yaxis');
```

```
title('Spectrogram of a Chickadee before Music');

figure;spectrogram(rwblackbirdA,winSize,overlap,fftSize,44100,'yaxis');
title('Spectrogram of a Red-WingedBlackBird after Music');
figure;spectrogram(rwblackbirdD,winSize,overlap,fftSize,44100,'yaxis');
title('Spectrogram of a Red-WingedBlackBird during Music');
figure;spectrogram(rwblackbirdB,winSize,overlap,fftSize,44100,'yaxis');
title('Spectrogram of a Red-WingedBlackBird before Music');

figure;spectrogram(songsparrowA,winSize,overlap,fftSize,44100,'yaxis');
title('Spectrogram of a Song Sparrow before Music');
figure;spectrogram(songsparrowD,winSize,overlap,fftSize,44100,'yaxis');
title('Spectrogram of a Song Sparrow after Music');
```

Function which Filters out Music

```
function [cleanedfile] = filteringoutmusic(recordingwithbird, ...
recordingwithoutbird, name)
% cross correlate the signals
[cc, lags] = xcorr(recordingwithbird, recordingwithoutbird);

figure;
plot(lags, cc);
title('Cross Correlation of ');
xlabel('Lags');
ylabel('Correlation');
%find the lag for greatest correlation
[~, index] = max(cc);
max_lag = lags(index);
```

```
% shift the signals so they line up
if max_lag<0
%     shift some signal
% shift back
    shifted_signal = [recordingwithbird(-1*max_lag:end); zeros(-1*max_lag,1)];

%     shifted_signal = delayseq(bird,-1*max_lag, Fs);
elseif max_lag > 0
%     shifted_signal = delayseq(bird,max_lag, Fs);
    shifted_signal = [zeros(max_lag,1); recordingwithbird(1:end-max_lag) ];
end

% soundsc(shifted_signal,44100)
% subtract the song one from the bird one

if length( recordingwithoutbird)> length(shifted_signal)
    cleaned = shifted_signal -  recordingwithoutbird(1:length(shifted_signal));
elseif  length( recordingwithoutbird)< length(shifted_signal)
    cleaned = shifted_signal(1:length( recordingwithoutbird)) - ...
    recordingwithoutbird;
end

audiowrite(char('cleaned_during_'+string(name)+'.wav'),cleaned,44100);
[cleanedfile, ~] = converttomono(char('cleaned_during_'+string(name)+'.wav'));
end
```

Code for Cross Correlation Example

```
% Create a finite impulse train

x = -20:1:20;

signal = exp(x);

figure;

subplot(2,2,1)

stem(x, signal);

title('Exponential Input e^{x}');


% Create Shifted Cosine

x2 = 20:1:40;

shifted_signal =  exp(x2);

subplot(2,2,2)

stem(x2, shifted_signal);

title('Exponential Input e^{x} Shifted 20 Units Right');

%Graph of AutoCorrelation

[auto, autolags] = xcorr(signal, signal);

subplot(2,2,3)

stem(autolags, auto);

title('Cross Correlation of Exponential Input with Itself');

xlabel('Lags');

ylabel('Correlation');

%Graph of Cross Correlation with a Delayed Signal

[cc, cclags] = xcorr(signal, shifted_signal);

subplot(2,2,4)

stem(cclags, cc);

title('Cross Correlation of Exponential and Shifted Exponential Inputs');

xlabel('Lags');
```

```
ylabel('Correlation');
```

Code for Convolution Example

```
% Create Box of Ones
box = ones(1,25);
% Create Cosine
t = 0:1:50;
signal = [t(1:10),20,t(11:15),30,t(16:25),5,t(26:34),40, t(35:40),20,t(41:50)];
% ramp function with noise
% convolve the two
output = conv(box, signal);
figure;
subplot(3,1,1);
stem(box);
title('Convolution Box');
ylabel('y');
xlabel('x');
subplot(3,1,2);
stem(signal);
title('Ramp Function');
ylabel('y');
xlabel('x');
subplot(3,1,3);
stem(output);
title('Convolution of Ramp and Box');
ylabel('y');
xlabel('x');
```

Appendix 2: Unused Code

The code in this section ended up being irrelevant, not the optimal approach, or difficult to use. See the Diagnosis and Improvement Sections for more information.

An m file containing a variation of functions which I tested to check their relevancy

```
% Loading Data
[datamonoduring, Fs] = converttomono('/home/alli/longswampyrecording_music.wav');
[background, ~] = converttomono('/home/alli/background_flute_swamp.wav');
% Filter out Data Below 100 Hz and above 15kHz
datamonoduring = bandpass(datamonoduring,[4000 10000],Fs);
background = bandpass(background,[4000 10000],Fs);
% calculate time vector

samplenum = 1:1:length(datamonoduring);
timevector = samplenum .* (1./Fs);
minutevector = timevector./60;
%% Preprocessing for Filtering Out Music During Music Playing

% Take recording of same song in the same places
% Upload both with and without birds
bird = datamonoduring;
% nobird
% cross correlate them
[cc, lags] = xcorr(bird, background);
%%
figure;
plot(lags, cc);
```

```
title('Cross Correlation of Background and Recording');
xlabel('Lags');
ylabel('Correlation');
% figure out the lag for greatest correlation
%%
[~, index] = max(cc);
max_lag = lags(index);
%%
% shift one of them so they have greatest
if max_lag<0
%     shift some signal
% shift back
    shifted_signal = [bird(-1*max_lag:end); ones(-1*max_lag,1)];

%     shifted_signal = delayseq(bird,-1*max_lag, Fs);
elseif max_lag > 0
%     shifted_signal = delayseq(bird,max_lag, Fs);
    shifted_signal = [ones(max_lag,1); bird(max_lag:end) ];
end
% what is a lag?
% soundsc(shifted_signal,Fs)
% subtract the song one from the bird one
%%
if length(background)> length(shifted_signal)
    cleaned = shifted_signal - background(1:length(shifted_signal));
elseif  length(background)< length(shifted_signal)
    cleaned = shifted_signal(1:length(background)) - background;
end
```

```
% soundsc(cleaned)
audiowrite('/home/alli/cleaned.wav',cleaned,Fs);
```

Appendix 3: Additional Spectrogram Files

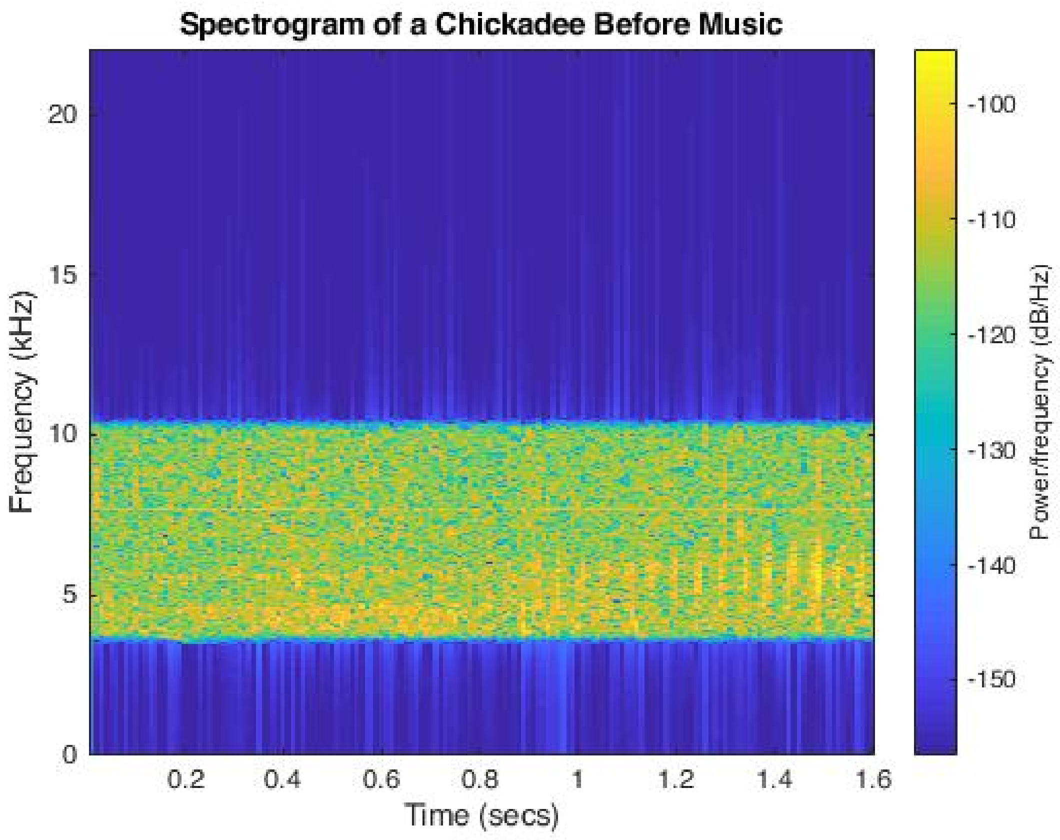


Figure 15: Spectrogram of Chickadee

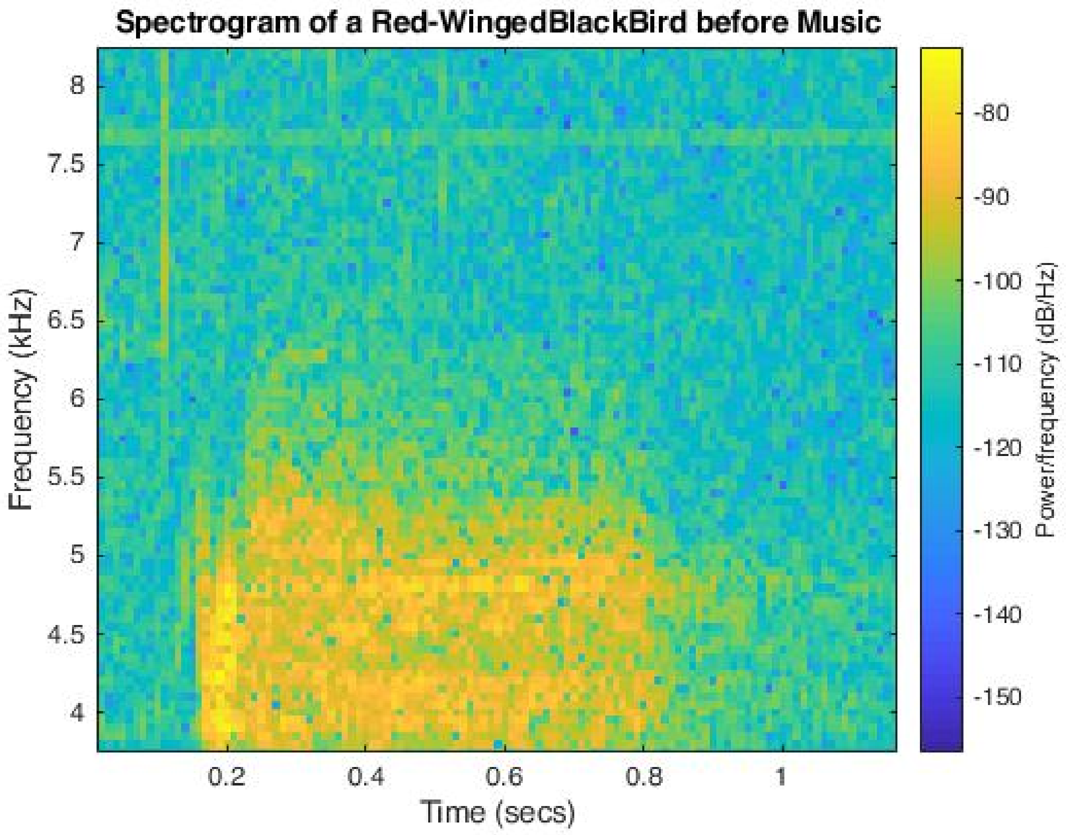


Figure 16: Spectrogram of Red-Winged Blackbird Before Music

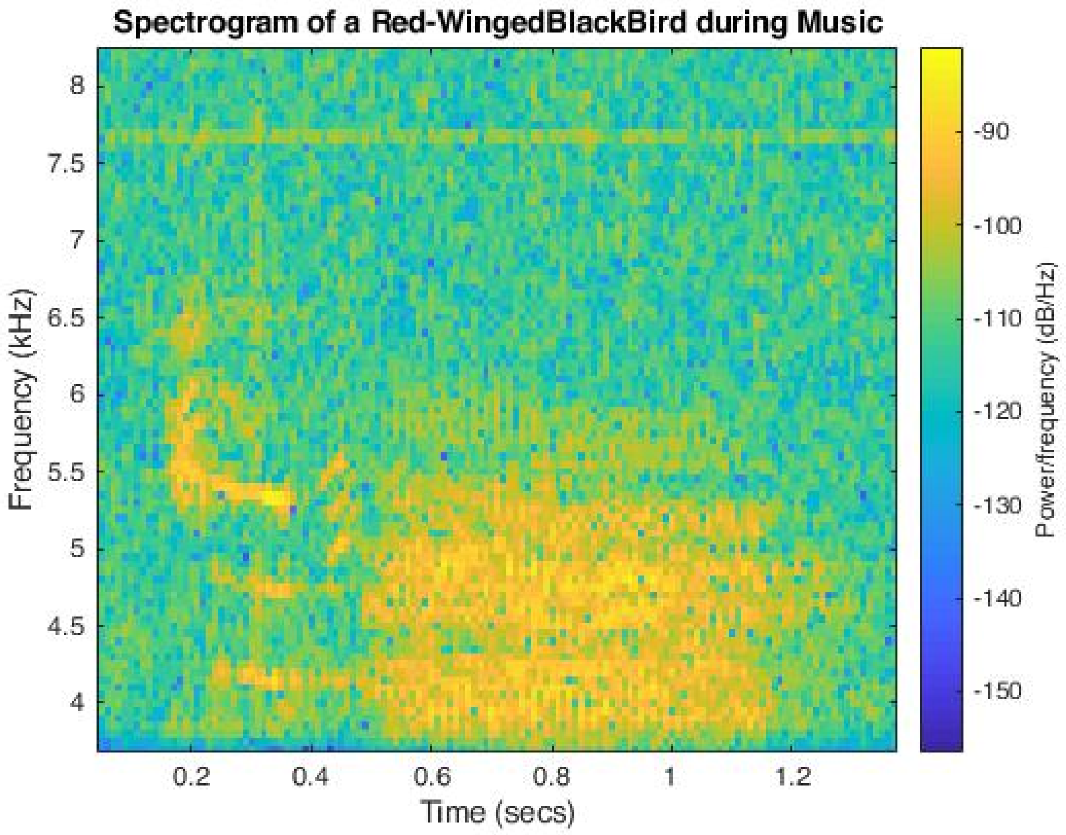


Figure 17: Spectrogram of Red-Winged Blackbird During Music

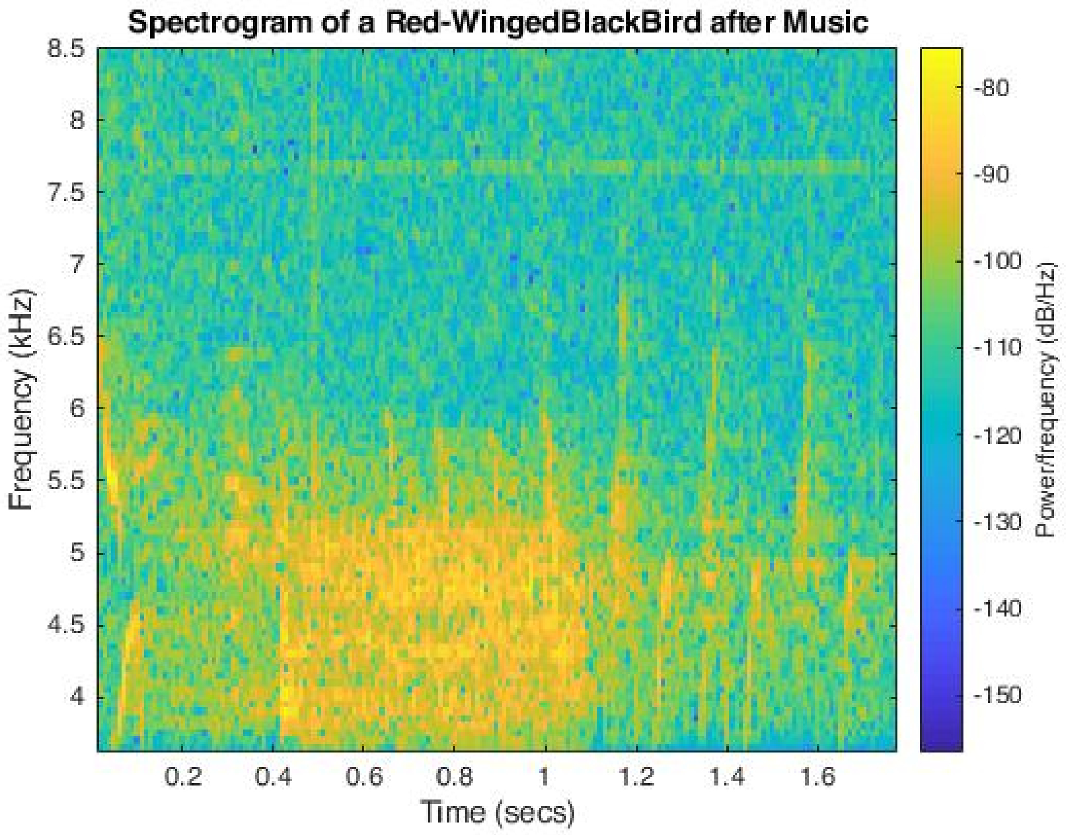


Figure 18: Spectrogram of Red-Winged Blackbird After Music

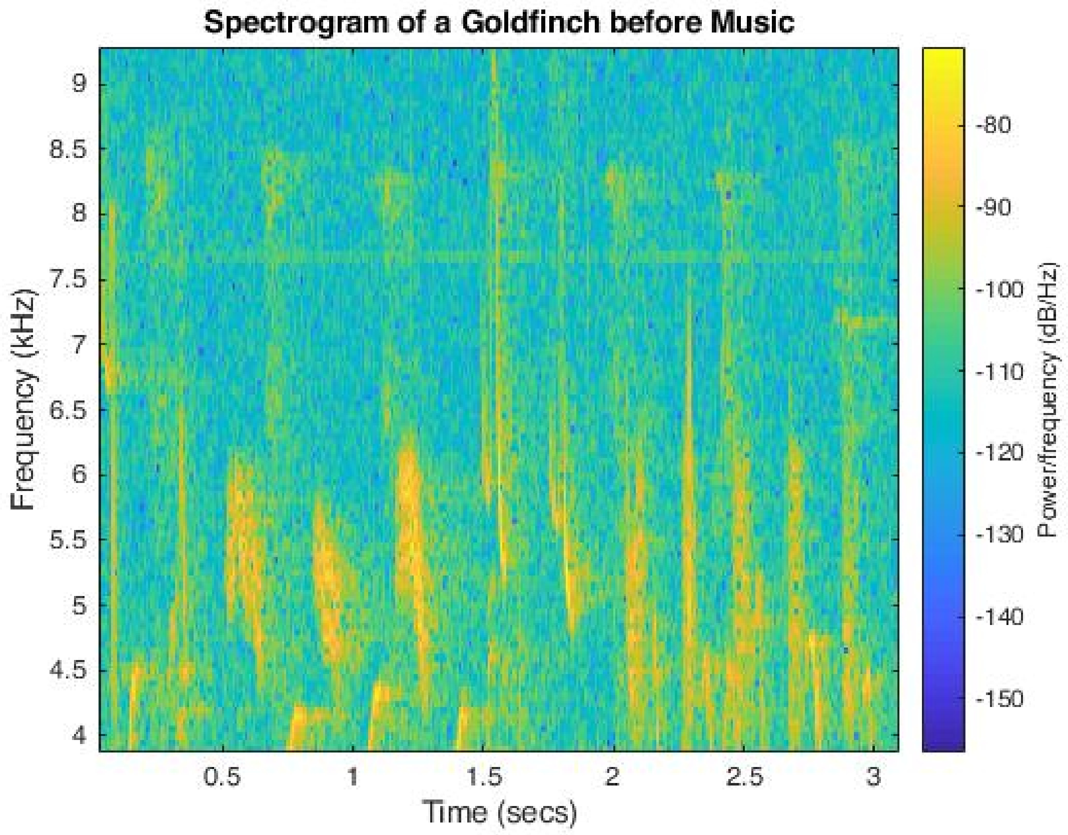


Figure 19: Spectrogram of Goldfinch Before Music

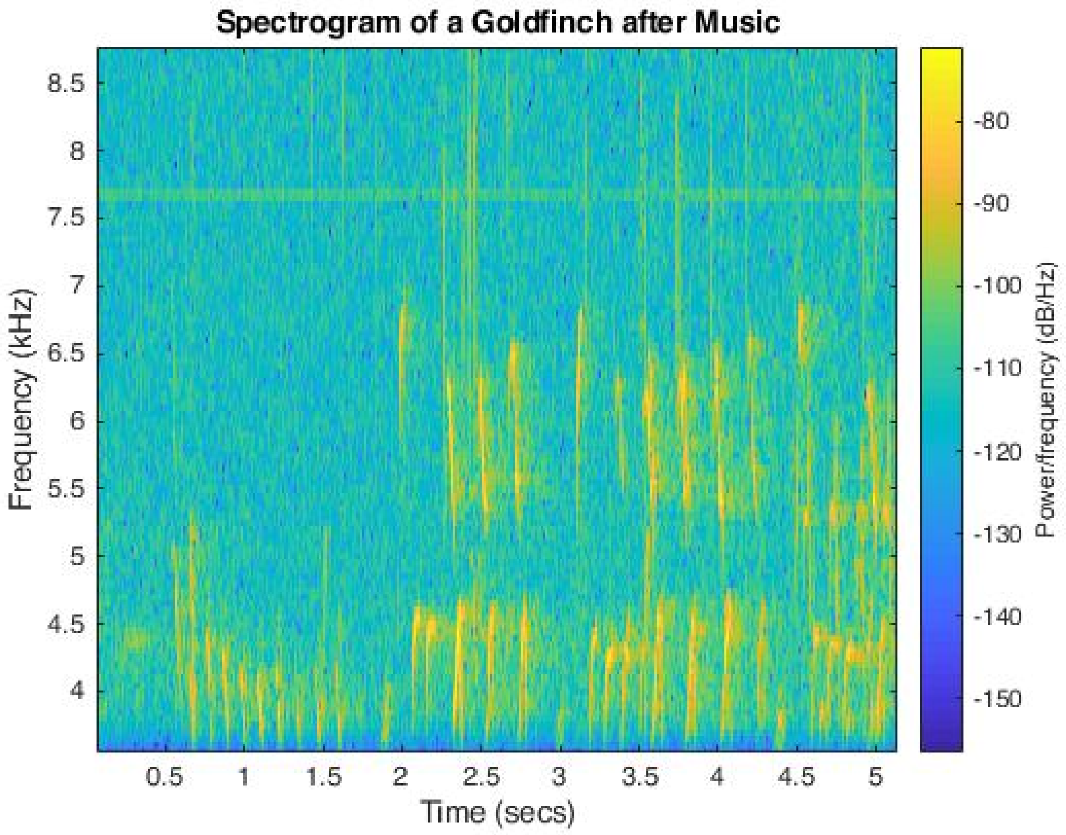


Figure 20: Spectrogram of Goldfinch After Music

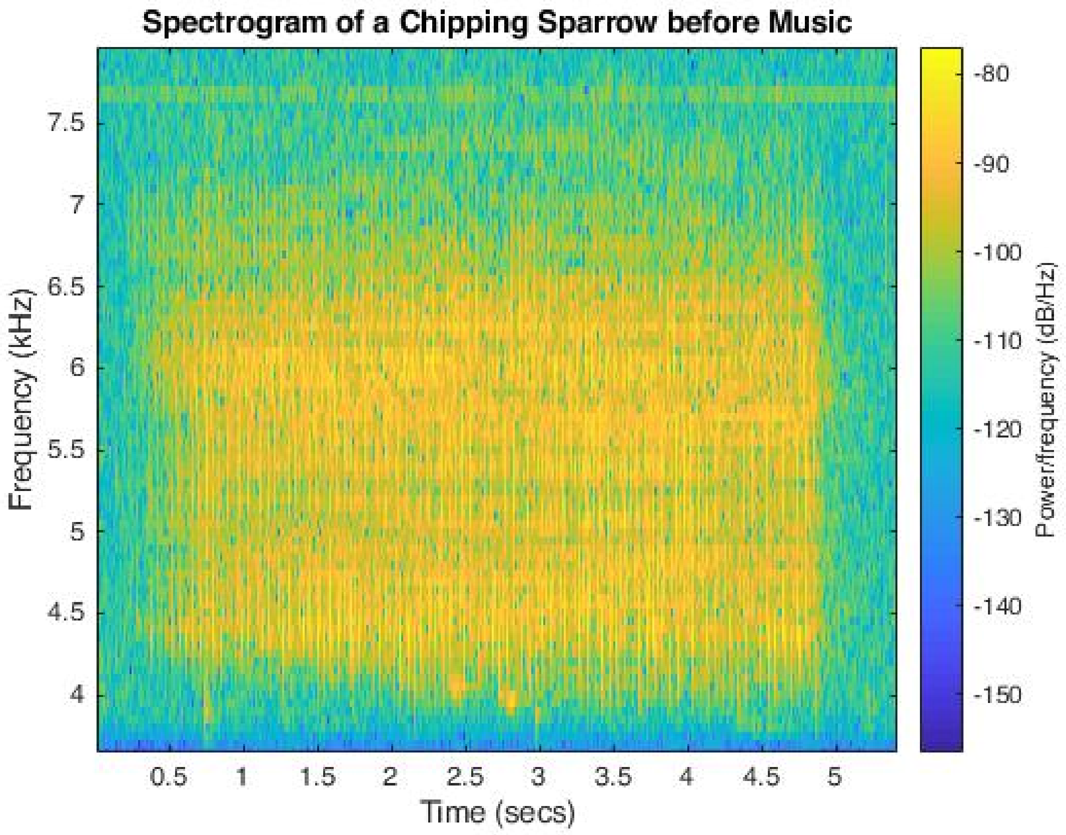


Figure 21: Spectrogram of Chipping Sparrow Before Music

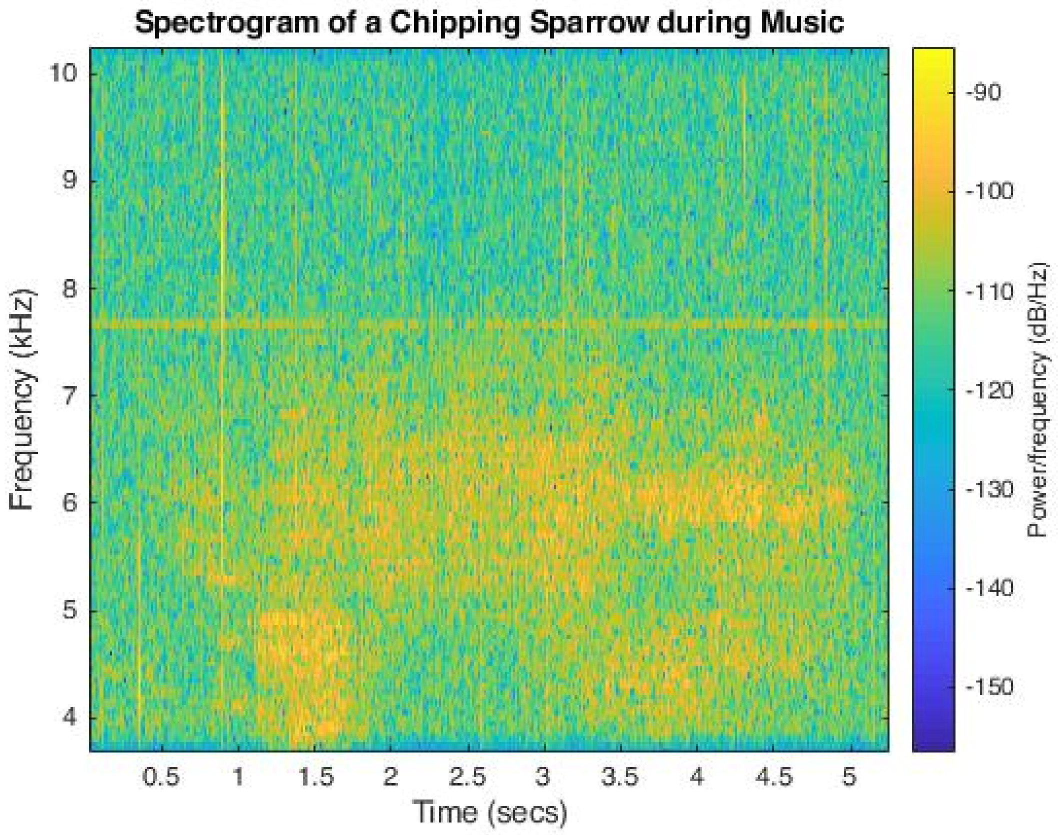


Figure 22: Spectrogram of Chipping Sparrow During Music

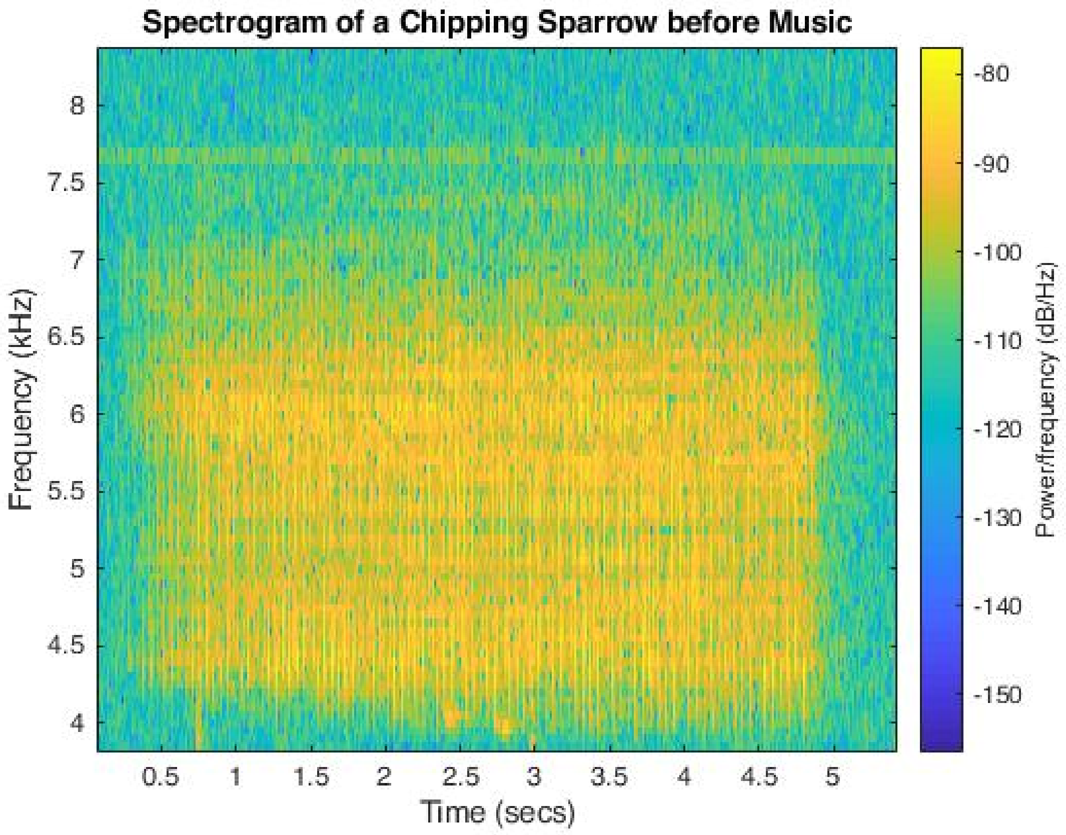


Figure 23: Spectrogram of Chipping Sparrow After Music

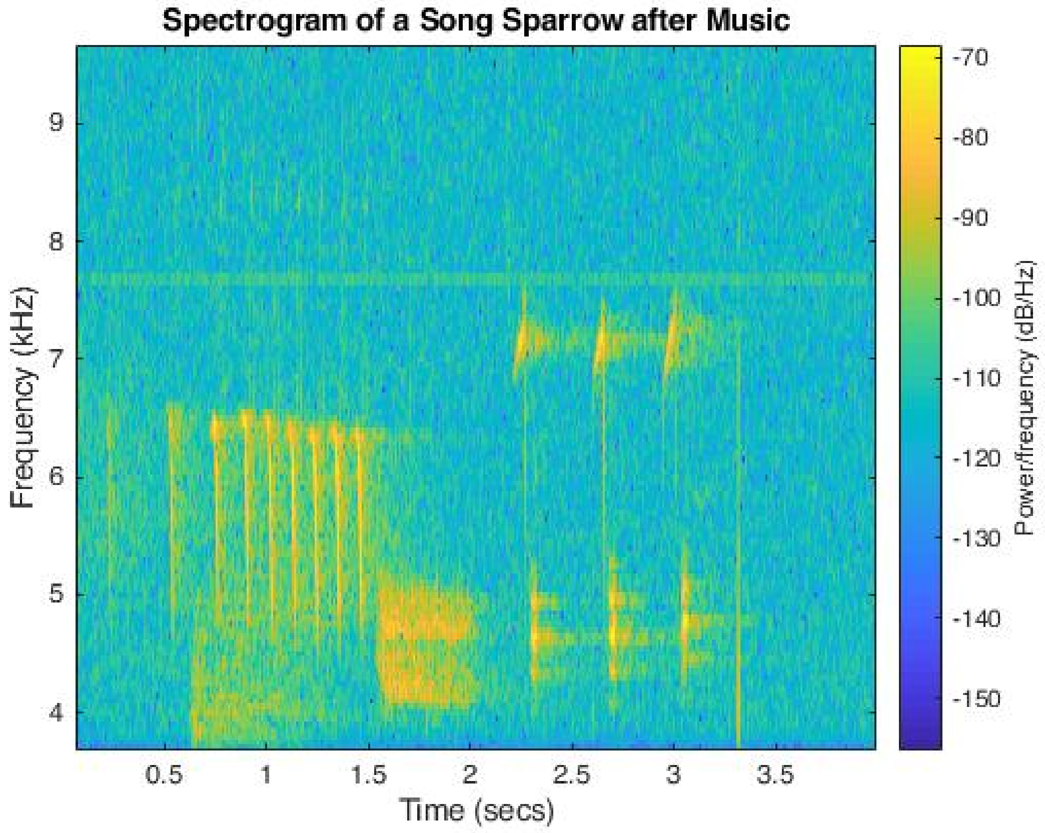


Figure 24: Spectrogram of Singing Sparrow After Music

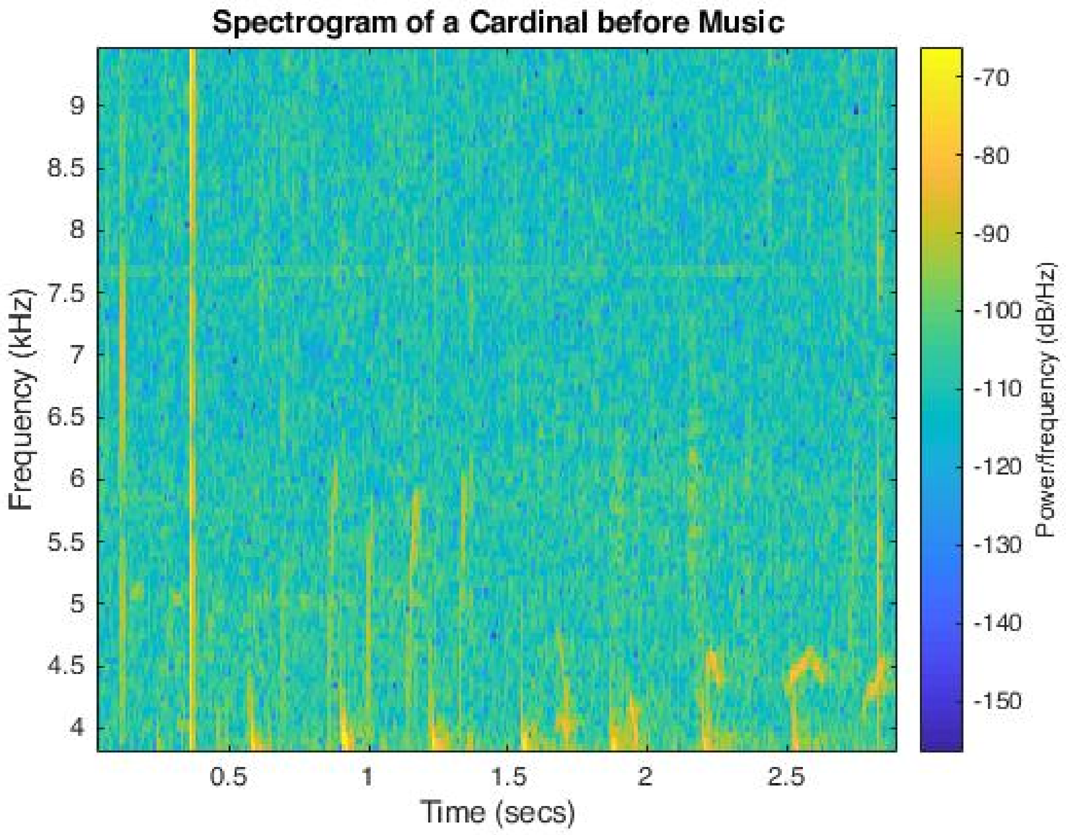


Figure 25: Spectrogram of Cardinal Before Music