

Midterm Submission

Abstract: Based on anecdotal claims, violins that are more highly priced are often preferred over those that are not, due to their supposedly higher sound quality. While this may be true when comparing instruments in the \$100 range to \$10,000 range, this study will explore whether or not high-priced violins are objectively better than others in a similar price range. Violins that are worth anywhere from \$100 to \$100,000+ will be the center of this study. Instruments in the upper echelon of the price range will be compared against each other to determine if there is any objective difference between the sound of the higher-level instruments. Digital signal processing will be applied to determine objective differences in sound quality. Sound samples that are recorded with special equipment will be digitally processed and examined in the frequency domain; analysis techniques in the frequency domain used in this study include the Fast Fourier Transform (FFT) and the Short-Term Fourier Transform (STFT). Spectrograms that can visualize the frequency domain of the instruments will be created, from which the intensities of different harmonics can be observed. Having applied this frequency domain analysis, the different harmonic intensities can be cross correlated against the different qualities of wood and methods of construction of the instruments. The result of this study will be a catalogue documenting different methods of construction and quality of material and how they impact harmonic intensities.

Review of Literature

Signal processing as a field of Computer Engineering is something that can be applied to many different disciplines. From recovering archaeological data to analyzing chess matches, the applications of signal processing are virtually boundless. These bounds also extend to encompass music, as all sound can be represented as a time-varying signal in the form of a pressure wave or voltage. The techniques of signal processing can then be applied to these signals, and information that may be imperceptible with the human ear can be extracted. Some signal processing techniques that can specifically be applied to music will be explored in this review.

The study of literature started broad, looking at recent developments in signal processing as it pertains to music. Machine learning is a big topic in signal processing lately; a paper by Therrick-Ari Anderson at Lincoln University studied musical instrument identification by using neural networks. The paper used libraries in a tool called MATLAB that took care of most of the signal processing [1]; the underlying algorithms under the libraries were also briefly introduced, mainly the Fast Fourier Transform (FFT).

In search of more recent work, we found research conducted by Atahan et. al. at Yildiz Technical University in Istanbul, Turkey. This paper focused on music genre identification using machine learning techniques such as autoencoders [2]. Again, we see the prevalence of machine learning in recent research. This study applied wavelet transforms to extract features from audio; the extracted features were then fed into the autoencoder algorithm developed by the team [2].

A common theme of time-to-frequency domain transforms appeared in the literature; both the FFT and wavelet transform take time-domain data and transform it into frequency domain information. This prompted further investigation into these algorithms. Research

conducted by the Department of Applied Electronics and Instrumentation at the Government Engineering College in Kozhikode, Kerala investigated using Spectrograms for Genre Classification [3]. A spectrogram is an application of the FFT that is very useful in analyzing sound data; it breaks a sound file into small pieces and applies the FFT to those small pieces. The study concluded that spectrograms were an accurate and reliable transformation to analyze music in the frequency domain [3].

Lastly, a more refined search was performed to locate literature that is directly pertinent to the subject matter of this study. A study conducted at the Department of Computer Science and Engineering at National Cheng-Kung University explored quantitative evaluation of violin performance by means of signal processing [4]. This study looked at both time and frequency domain data, making use of spectrograms in close relation to the goal of this project. The evaluation criteria set by the study, such as pitch variation and characteristics of vibrato [4], may be applied to our project to determine quality of instruments as opposed to quality of performance.

Signal processing is a very wide field, with applications ranging from digital communication and data transfer to fields like power generation and weather forecasting. In the real world, almost any phenomena can be viewed as a signal, which may vary both in time and frequency. When looking at music, signal processing is something that comes very naturally. In its simplest form, music can be seen as sound, which is a time-varying pressure wave measured at the ear drum. The intensity of this pressure wave is how loud the sound is, and the frequency of oscillation of this pressure wave is the pitch of the sound wave. This realization allows us to apply signal processing techniques to analyze music.

Many techniques in signal processing involve viewing time-varying signals in the frequency domain. Earlier, we arrived at the conclusion that sound can be viewed as a pressure wave that varies with respect to time. This view of sound tells us a lot of information, like how loud the sound is at any given point, how long our sound lasts in the time domain, and other similar features. While this is a helpful representation, it does not tell us what frequencies comprise our sound wave. Signal processing allows us to solve this problem, by giving us a tool known as the Fourier Transform, which takes a signal from the time domain to a signal in the frequency domain. The frequency domain can tell us what frequencies make up our sound, along with the intensities of each frequency.

Before we discuss the Fourier Transform and its many variants, it is important to discuss one factor that can hold the technique back if not applied correctly: sampling. Music exists in the continuous world, where our ears can perceive changes in sound continuously. To apply signal processing, our music must be sampled and digitized—this induces error in multiple ways. First, a digital system cannot sample continuously, meaning there is loss of information to some degree. For example, consider a digital system that samples at a frequency of 200 Hz, and a

violinist playing his open A string, which rings at 440 Hz. Since the digital system samples at a frequency lower than the fundamental of the note, it will not accurately capture the 440 Hz waveform, resulting in aliasing. Sample theory tells us that to accurately capture a signal, it must be sampled at twice the frequency of its highest frequency component. Therefore, most music is sampled at 44.1 kHz; humans cannot hear frequencies larger than 20 kHz, so the sampling frequency must be at least 40 kHz. The extra 4.1 kHz is an engineering margin to account for error. This is all relevant information for this project, as the goal is to analyze high frequency content of violins, which would require a high sampling rate. It is also relevant to the algorithms we will be working with, as sampling can impact the performance of these techniques.

Considering sample theory, the simplest form of the Fourier Transform (FT) is the Discrete-Time Fourier Transform (DTFT). This is a rudimentary form of the FT, which can take in an infinite-length sampled input and produce a frequency spectrum that is continuous in both frequency and amplitude. The algorithm does this by multiplying the signal by a sinewave of all possible frequencies and storing the amplitude of the output. It is important to note that this is partially a theoretical construct, as no computer could store a continuous spectrum.

To account for this, the Discrete Fourier Transform (DFT) is introduced, which is a sampled version of the DTFT. While the notation may be confusing here, it is important to realize that the DTFT and the DFT are two different but related things. The algorithm that implements the DFT, known as the Fast Fourier Transform (FFT), is the actual code that executes the calculations. While an algorithm for DTFT can be written, it is too computationally complex and slow to be used practically, so the FFT is used, which is an application of the DFT, which is an application of the DTFT. FFT is a couple layers removed, which is where we start to see some of its error. Because it's a sampled version of another system, the frequency resolution

is discrete. This may be a problem for us, as we need to view very small changes in very high frequencies. One way of getting around this would be to increase the sample rate significantly. If we do this, the algorithm is strong in the frequency domain; we would be able to accurately tell what frequencies occur and their strengths. The biggest drawback to this algorithm is the time-domain resolution. The FFT has no time domain resolution at all. While we may know what the frequencies are, we do not know at all when they occur. This may not be a problem for recordings that have single notes, but for anything more complex, the FFT fails.

This drawback leads us to the Wavelet Transform (WT), which is an algorithm like the Fourier Transform (FT) in its theoretical construction. Instead of multiplying the signal by a sinewave, the signal is multiplied by a waveform known as a wavelet. Sparing some technical details, using wavelets gives us a good time domain resolution at the expense of some frequency domain resolution. With this tool, we would know what frequencies occur when, and with what the intensities are. This is a good candidate option, but we sacrifice some of our frequency domain resolution when we use this.

The middle ground between the FFT and WT for our application is the Short-Term Fourier Transform (STFT), often known as a spectrogram. This algorithm functions by taking the full time-domain signal and slicing it into a set of smaller time domain signals. Then, the FFT is applied to each of the smaller time domain signals, producing a set of small frequency domain signals. The frequency domain signals are then all stitched together to produce a spectrogram, which tells us how frequency content varies with respect to time. This is the ideal algorithm for this project, as it gives us as much control as we want over time-domain resolution, and theoretically infinite precision in the frequency domain.

Short Term Fourier Transform

After exploring a few different algorithms and techniques in the previous section, we have settled on using the Short-Term Fourier Transform (STFT). This algorithm takes a sound file and splits it up into smaller sound files, often into time steps of one to two milliseconds. Recall from our discussion of sampling that audio is typically sampled at around 44.1 kHz; this means that a one millisecond audio clip will have 44 samples. A Fast Fourier Transform (FFT) algorithm is then applied to these smaller sound files; each FFT is combined into a plot known as a spectrogram. A plot of a spectrogram can be seen in Figure 1.

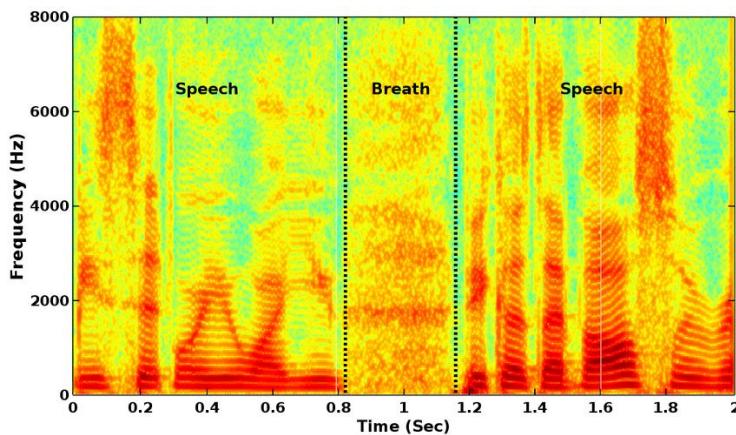


Figure 1: Spectrogram Plot

The color of the plot indicates the intensity of the frequency. For this plot, darker colors indicate a larger presence of the frequency, while lighter colors indicate a smaller presence. This plot can also be viewed as a three-dimensional construct, with peaks in the Z-axis representing intensity of sound. This can be seen in Figure 2.

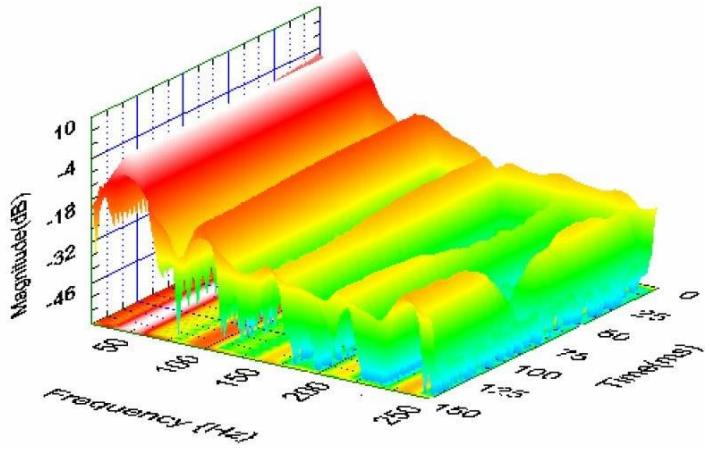


Figure 2: 3D Spectrogram Plot

These representations offer a good intuition of how the spectrogram works; specifically, it shows how much control we have over both frequency resolution and time resolution. Applying this to a sound file of a musical piece, we will know what frequencies occur with what intensity, and when they occur. Both two dimensional and three dimensional plot will be generated for different qualities of violins; from this, we will be able to discern patterns in the upper harmonics.

Initial Data Collection and Processing

The initial batch of instruments under study comprises of two instruments in the low-price range, and two instruments in the middle price range. The two low price range instruments are Kutz 100 violins, valued at \$300.00; the two middle price range instruments are a John Cheng and Kono, both valued at around \$3,000.00. It is important to note that the middle range instruments are about 10 times as expensive as the low range instruments; one focus of this project will be to determine if the more expensive instruments are truly better by a factor of 10.

For each instrument, a recording was taken of the open strings, a G major scale bowed, the harmonics on each open string, the G string plucked, and a G major scale plucked. This resulted in a total of 12 recordings for every instrument, and 48 recordings in total for this first batch of instruments.

For this project, the entire data collection process had to be done in a very controlled environment. The recordings for all of the violins were done in the same session and same room, using the same bow and same player. This methodology ensured that the unit under test was the violin and helped reduce variations in the data. The data was collected using the Zoom H1N recorder and sampled at 96 kHz. This specific recorder and sampling rate are important to the project, and steps will be taken in the data processing to preserve the sample rate.

MATLAB is a data processing language that is often used at engineering firms for data science and signal processing tasks. It is designed to be used to deal with very large amounts of data efficiently, both in terms of developing the software and using computational resources. The language functions through scripts, which are written to perform small dedicated tasks. Multiple

scripts can be combined for more complex operations. In this project, MATLAB will be used for all stages of data processing.

Recalling our discussion about sample theory, the sampling rate is one of the defining qualities of a signal, and drastically impact the accuracy and range of our frequency domain analyses. It is important that we preserve the sample rate through the preprocessing stages. Each recording was taken with information about the recording spoken at the beginning (e.g. “Kutz 100 #1 Open G”), followed by the open G being played in this case. This information was not necessary for the data processing, and needed to be edited out.

Using audio processing software like Audacity was not effective for this, as it threw samples out of the data when exporting the edited audio, effectively reducing the sample rate. To get around this, a small MATLAB script was written to trim the data by hand. While this was more tedious and time consuming, the sampling rate of the data was not affected by the trimming process, resulting in no loss of essential data.

After preprocessing the data, I began my analysis by looking at the open string recordings and writing a script that can process and extract information from them. The first thing I looked at was the Fast Fourier Transform (FFT) of the open string recordings. Recall, the FFT is an implementation of the Fourier Transform that can perform quickly on large data sets, and gives good frequency resolution but poor time resolution. Since the pitch is not changing significantly with respect to time on the open string recordings, the FFT is a good algorithm to use here.

The FFT’s frequency resolution can be calculated by dividing the sampling frequency of our signal by the length of the signal in samples. For a 1 second recording sampled at 96 kHz, the frequency spacing would be 1 Hz, as 96,000 samples would be recorded in the one second

time period by definition. Since our open string recordings are about 15-20 seconds each, the frequency resolution is very good for the open string recordings. The FFT of the open string recording of one of the Kutz 100 violins can be seen in Figure 1.

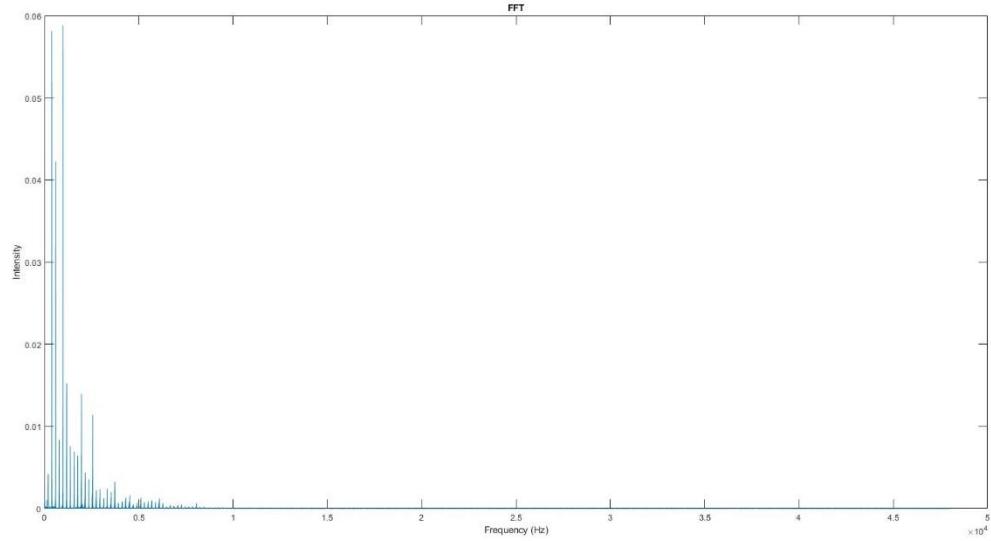


Figure 1: Kutz 100 Open G FFT

Note that the FFT will have a bandwidth of 48 kHz, meaning it is able to detect frequencies of up to 48 kHz. Most of the activity for all the open strings looks like this—the peaks are the harmonics of the fundamental. In this case, the fundamental frequency is the 196 Hz peak shown at the very left of the plot. While it may seem like the majority of the plot is zero or very close to it, zooming in and analyzing further shows that this is not the case. Due to the extreme differences in activity between the upper and lower frequency ranges, the two will have to be analyzed in different ways. Figure 2 shows a more enhanced version of the lower frequency range.

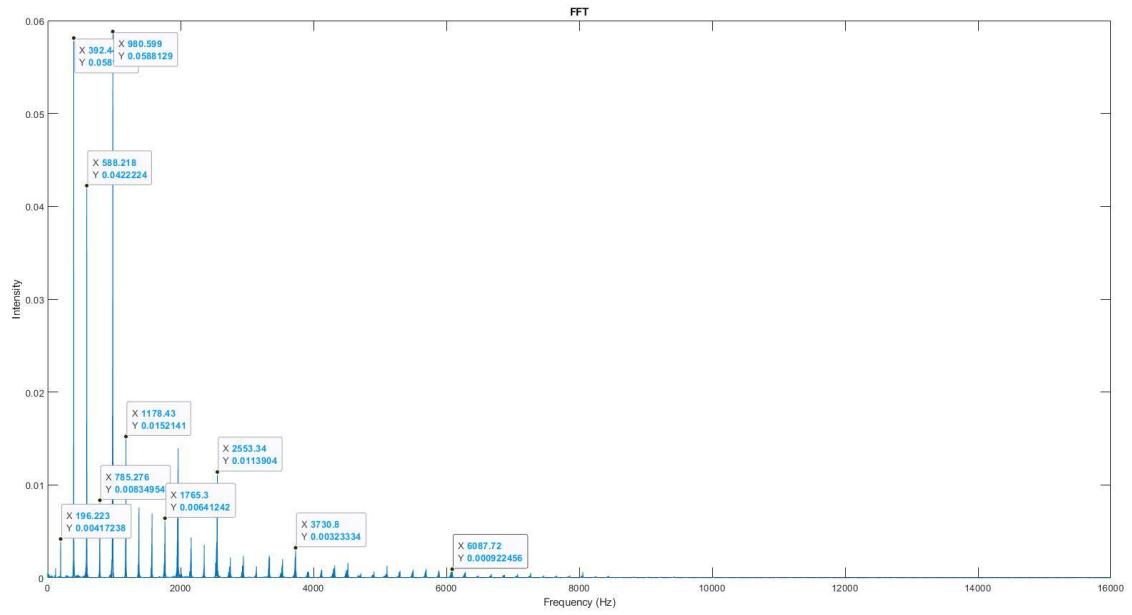


Figure 2: Kutz 100 Lower Frequency Range

Note that the peak of the fundamental on this instrument is very weak; in fact, this analysis tells us that the second, third, and fourth harmonics of the open G on the Kutz are at least 10 times stronger than the fundamental. Also note that the peaks on the harmonics of the lower frequency follow an approximately exponential trend, decreasing exponentially as frequency increases.

Let's look at the upper frequency range; the magnitudes of the FFT in this range are much smaller than in the lower frequency range. This can be observed in Figure 3.

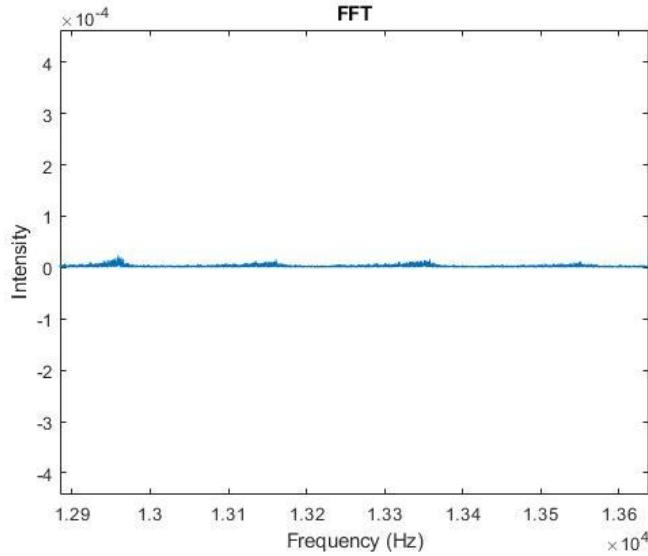


Figure 3: Kutz 100 Middle Frequency Range

The middle frequency range also exhibits some harmonics although significantly weaker. This trend also dies out around 20 kHz, which is commonly agreed upon as the limit of human hearing.

Spectrograms were also generated of the G major scale recordings, since the pitch varies with respect to time in the scale recordings. Both two dimensional and three-dimensional spectrograms were generated, each with their own drawbacks. The two-dimensional spectrogram can be seen in Figure 4. Due to the high sample rate, the spectrograms will also need to be split into three sections for analysis, as each section has different characteristics that need to be analyzed.

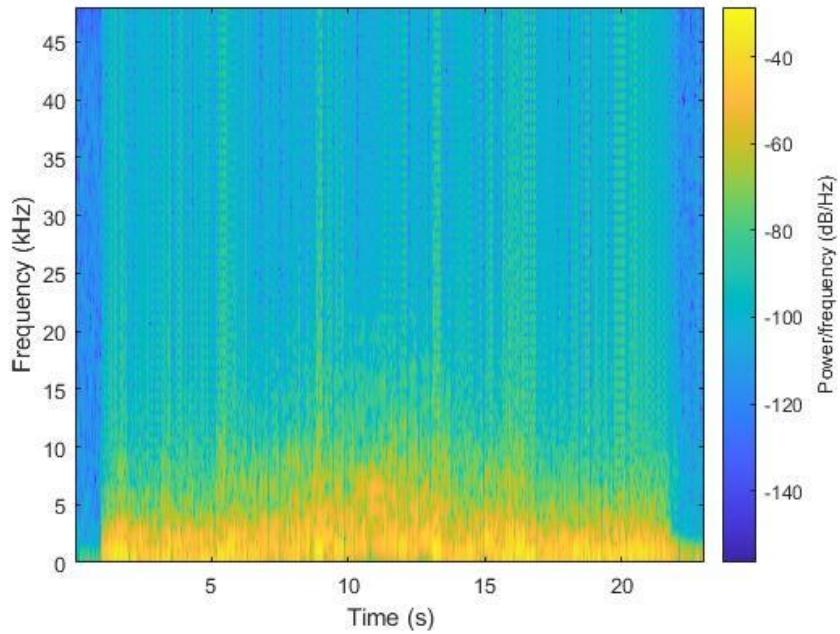


Figure 4: Kutz 100 G Scale Spectrogram

Most of the spectrogram also looks constant, since the spectrogram will also observe up to 48 kHz. The lighter color represents higher presence of harmonic. We can see streaks of harmonics going up the graph, showing that there is indeed upper harmonic behavior well beyond the human hearing range, enough to be detected by the algorithm.

Moving forward, the goal will be to extract the information contained within these higher frequency peaks. Once data processing is complete for all of the instruments, we will begin to compare instruments in the same price ranges to look for consistency and compare across price ranges to determine objective measures of differences in tone.

References

T. -A. Anderson, "Musical instrument classification utilizing a neural network," 2017 12th International Conference on Computer Science and Education (ICCSE), 2017, pp. 163-166, doi: 10.1109/ICCSE.2017.8085482.

Y. Atahan, A. Elbir, A. Enes Keskin, O. Kiraz, B. Kirval and N. Aydin, "Music Genre Classification Using Acoustic Features and Autoencoders," 2021 Innovations in Intelligent Systems and Applications Conference (ASYU), 2021, pp. 1-5, doi: 10.1109/ASYU52992.2021.9598979.

N. M R and S. Mohan B S, "Music Genre Classification using Spectrograms," 2020 International Conference on Power, Instrumentation, Control and Computing (PICC), 2020, pp. 1-5, doi: 10.1109/PICC51425.2020.9362364.

Y. Lin, W. -C. Chang and A. W. Y. Su, "Quantitative evaluation of violin solo performance," 2013 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference, 2013, pp. 1-6, doi: 10.1109/APSIPA.2013.6694296.