Fourier Analysis and Musical Signal Processing

Matthew Lad

Jan.-Mar. 2022

1 Introduction

[[[The goal of this paper is to explore the application of Fourier analysis in signal processing and specifically the analysis of music. The mathematical background of music and Fourier analysis will be covered as well as the Fast Fourier Transform (FFT) algorithm and applications of the FFT on musical signals. Along with the mathematics, explorations will be done through computer algorithms.]]]

2 Musical Preliminaries

2.1 Sound

In physics, **sound** is a series of oscillating low and high air pressures. The oscillations are created when a material vibrates. A **cycle**, or **vibration**, is one high-pressure and one low-pressure area together. The **frequency** or **pitch** is the number of cycles per second, measured in **Hz** [1]. The higher the frequency of the sound, the higher the pitch of the sound heard. The **volume**, "**loudness**", or **amplitude** of a sound is defined by the difference in air pressure between a low area and high area of a cycle. A quiet sound can be described as a transition between two densities of similar value. Likewise, a loud sound is a transition between a very high density and a very low density [2] [3]. These terms oscillating, cycles, frequencies, etc. are all terms used when dealing with certain functions in mathematics. A diagram of this relation is shown in figure 1. The elementary **sine** function graphs the air density over time. As the density increases, the sine wave increases.

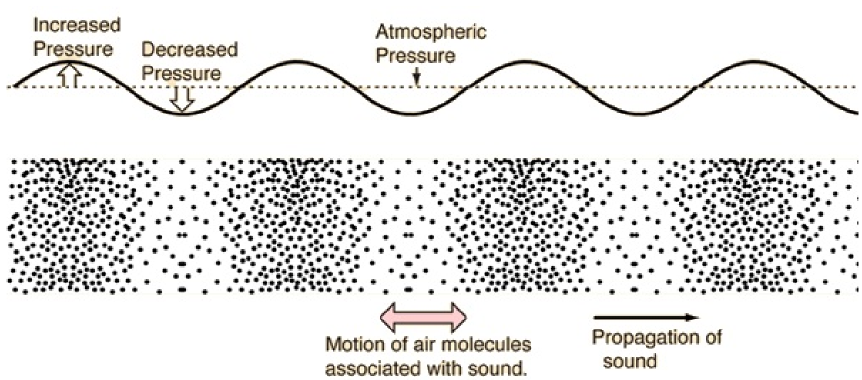
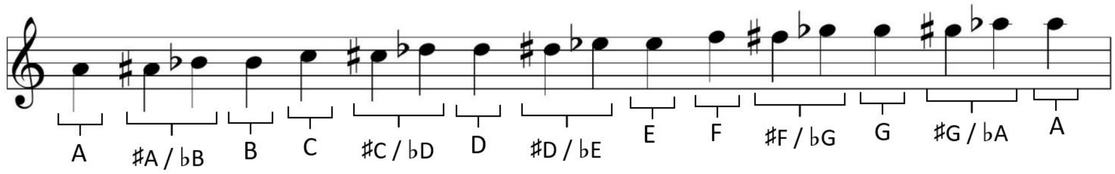


Figure 1: The high and low air pressures of sound can be represented by a wave function [4].

2.2 Music Theory

Different notes are determined by specific frequencies. The lower sounding a note is, the lower the frequency of the sound wave. Similarly with high-pitched sounds, their frequencies are much higher. The most common system in western music for determining the frequency values of musical notes is a 12-tone system of ratios called **equal temperament**. Given a reference note with frequency x, there are 11 notes between x and $2x$. Each of these notes are separated by a ratio equal to the 12th root of 2, ($\sqrt[12]{2} \approx 1.05946$). See figure 2 for a 12-tone scale using equal temperament [5] [2].



12-Tone Equal Temperament			
Note	Interval	Ratio	Frequency(Hz)
A	Unison	1 : 1	440.00
♯A / ♭B	Minor Second	1 : 1.05946	466.16
B	Major Second	1 : 1.12246	493.88
C	Minor Third	1 : 1.18921	523.25
♯C / ♭D	Major Third	1 : 1.25992	554.37
D	Fourth	1 : 1.33483	587.33
♯D / ♭E	Diminished Fifth	1 : 1.41421	622.25
E	Fifth	1 : 1.49831	659.25
F	Minor Sixth	1 : 1.58740	698.46
♯F / ♭G	Major Sixth	1 : 1.68179	739.99
G	Minor Seventh	1 : 1.78180	783.99
♯G / ♭A	Major Seventh	1 : 1.88775	830.61
A	Octave	1 : 2	880.00

Figure 2: The following shows the 12 divisions within the equal temperament system. Each ratio is a $\sqrt[12]{2} \approx 1.05946$ apart [2] [6] [7]. Although considered the same in the equal temperament system, the paired accented notes (♯A / ♭B, etc.) are actually not the same. See [5] for more information on this beyond the scope of this paper.

Today the note "A", denoted "Unison" or A4, is commonly set to 440Hz and is used as a reference point for calculating the other frequencies. This standard has changed throughout history and has its origins in medieval Europe. See [8] for further reading.

When a note is played by a computer, the frequency is perfectly calculated and a very artificial sound is heard. With instruments and the human voice however, perfect frequencies are not generated and there are qualities which cause notes to sound unique and beautiful. For example, when a clarinet and a trumpet both play A4, even though they play the same frequency, they sound very different. This is due to differences in air flow and materials, wood vs. metal, lip buzzing vs. reed vibrations, etc. Analyzing the frequencies of computer generated notes and instrument generated notes reveals something amazing behind the physics of musical sound.

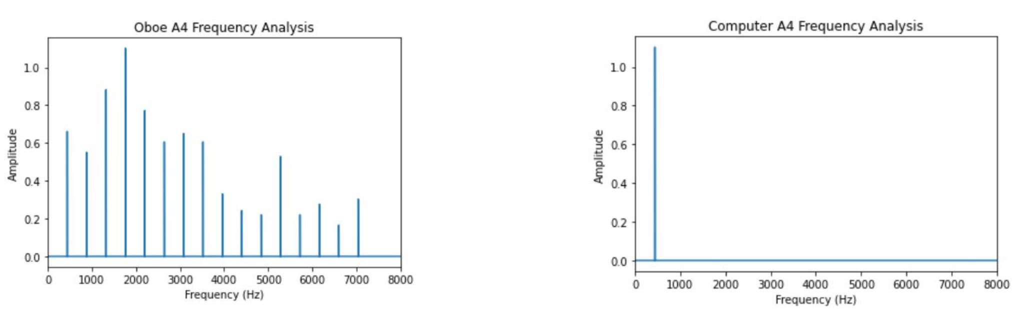


Figure 3: The graph on the left shows the frequency analysis of an oboe playing an A4. The right graph shows a computer playing the same note.

Figure 3 shows the frequency analysis of a computer generated A4 and an A4 played by a oboe. As you can see on the computer graph, the large spike is centered around 440 Hz, which matches the table in figure 2 for the note A4. Notice how the oboe graph contains multiple other spikes decreasing in size from the largest spike at 440 Hz. This phenomena is known as the **overtone** or **harmonic** series and consists of regular intervals from the **fundamental**, or largest, frequency. These successive spikes are created by properties of the instrument used such as material, size, air column, etc., meaning every note played is slightly unique. Because a computer will always generate the exact same frequency, its notes are never unique. This is one reason for the unique beauty in singing and instrumental music.

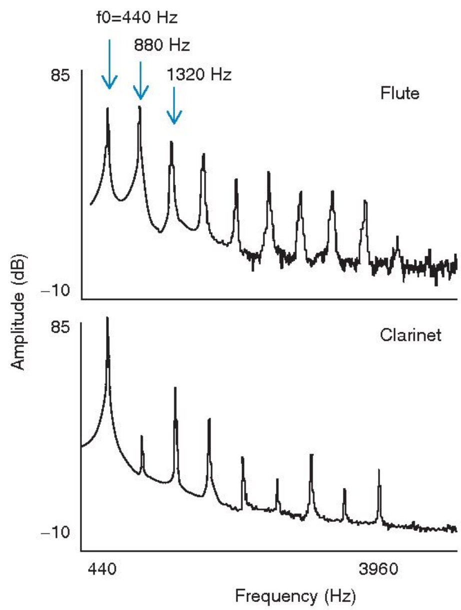


Figure 4: Graphs of a clarinet and a flute both playing the note A4 [].

Figure 4 shows the frequency analysis of a clarinet and flute both playing the note A4. Notice how although both are playing the same note, the differences in spikes leads to the different sound between the two instruments. The frequencies within the harmonic series are determined by whole number ratios. No matter what the fundamental frequency is, the harmonic series will always follow the ratios; 1:1, 1:2, 1:3, 1:4, ..., 1:n for n harmonics in the series.

2.3 Time and Frequency Domains

There are two common ways of graphing sound. It can be graphed as amplitude over time, which shows the volume of the sound, or as amplitude over frequency, which shows the frequency of the sound. Mathematically, these are referred to as the **time domain representation** and **frequency domain representation**. The time domain shows how the sound physically behaves over time (the x-axis is time and the y-axis is amplitude). The frequency domain shows the frequencies of the sound (the x-axis is frequency and the y-axis is the amplitude, or strength of the frequencies). Time domain data can be generated through a microphone as it is a literal representation of the sound waves. The frequency domain data is much more complicated and requires a bit of math to extract from the time domain data. The rest of this paper will focus on understanding this process of moving from the time domain to the frequency domain.

2.4 Mathematical Preliminaries

The following sections will serve both as an introduction to the mathematics of music as well as an exploration of various concepts through computer programs. Additionally, the process of moving between the time and frequency domains will be described and explored.

2.4.1 Periodic Functions

A **periodic function** is any function for which $f(t) = f(t+T)$ for all t. The smallest constant T which satisfies it is called the **period** of the function (it 'repeats' every T) [9]. The sine wave that represented the sound in figure 1 is an example of a periodic function where $f(t) = A\sin(\omega t + \varphi)$. Here A is the amplitude, or the maximum deviation of the function from zero. $\omega = 2\pi f$ where f is the frequency, also denoted $\frac{1}{T}$, multiplied by 2π to convert it to an angular frequency (a measure of radians per second instead of cycles per second). Lastly, φ specifies the phase of the wave (a measure, in radians, of where the oscillation is in within its cycle at $t = 0$) [10] [11].

Showing an example of a sine function that generates a musical note is figure 5

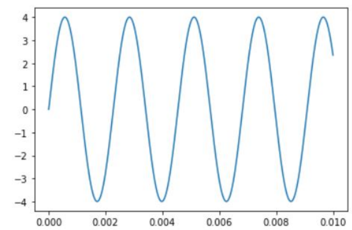


Figure 5: The note A4 created using the sine function: $f(t) = A\sin(\omega t + \varphi)$, where $A = 4$, $\omega = 2\pi 440$, and $\varphi = 0$.

By changing the amplitude $A = 20$ and making the x-axis go from 0 to 10000, the sine function can be calculated and stored into a wav file. This file can then be played on a computer and the note A4 will actually be heard. By adding successive sine functions to the original, a more natural sounding A4 can be generated. These added sine functions represent the additional harmonics.

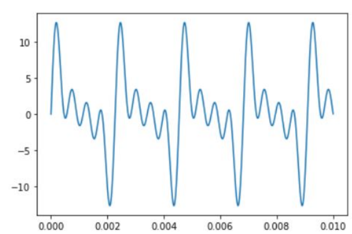


Figure 6: The note A4 created using the sine functions: $f(t) = 4\sin(2\pi 440t + \varphi) + 3.95\sin(2\pi 880t + \varphi) + 3.9\sin(2\pi 1320t + \varphi) + 3.85\sin(2\pi 1760 + \varphi)$.

Notice in 6 that the amplitudes (4, 3.95, 3.9, 3.85) decrease and the frequencies (440, 880, 1320, 1760) increase by the before mentioned ratios (1:1, 1:2, 1:3, 1:4).

Moving now away from the music side, we will dive into the Fourier analysis, which will allow us to move from the time domain into the frequency domain and vice versa. This exploration begins with Orthogonality.

2.4.2 Orthogonality of Vectors and Functions

A **vector space** is any set that satisfies the defined axioms of addition and scalar multiplication. Details not listed here. An **inner product** is a function on a vector space which satisfies the following axioms: (u and v and w are vectors in the vector space and c is a scalar) [12].

1. $\langle u, v\rangle = \langle v, u\rangle$
2. $\langle u, v + w\rangle = \langle u, v\rangle + \langle u, w\rangle$
3. $c\langle u, v\rangle = \langle cu, v\rangle$

4. $\langle v, v\rangle \geq 0$, and $\langle v, v\rangle = 0$ if and only if $v = 0$.

An **inner product space** is a vector space with a defined inner product which satisfies the above axioms for every vector within it. Properties & Definitions related to Inner Product Spaces: [12].

1. $\langle 0, v\rangle = \langle v, 0\rangle = 0$
2. $\langle u + v, w\rangle = \langle u, w\rangle + \langle v, w\rangle$
3. $c\langle u, v\rangle = \langle u, cv\rangle$
4. The **norm** (or **length**) of u is $||u|| = \sqrt{\langle u, u\rangle}$
5. The **distance** between u and v is $d(u, v) = ||u - v||$
6. The **angle** between two nonzero vectors u and v is:
   $\cos(\theta) = \frac{\langle u,v\rangle}{\|u\|\|v\|}$, $0 \leq \theta \leq \pi$
7. Vectors u and v are **orthogonal** if $\langle u, v\rangle = 0$.

When a set of vectors $S = \{u_1, u_2, u_3, ...\}$ satisfies $(u_n, u_m) = 0$ when $n \neq m$, we say that the elements of S are **mutually orthogonal** and that they form an **orthogonal basis** for the space spanned by S. Any vector in that space can be represented as a linear combination of those basis vectors [12].

Sets of functions: The set of functions $\{f_1(t), f_2(t), f_3(t), ..., f_k(t), ...\}$ is orthogonal on an interval $a < t < b$ if for any two functions $f_n(t)$ and $f_m(t)$ in the set, [12].

$$\langle f_n, f_m\rangle = \int_a^b f_n(t) f_m(t)\, dt = \begin{cases} 0 & \text{for } n \neq m \\ r_n & \text{for } n = m \end{cases} \tag{1}$$

2.4.3 Complex Exponentials

Let $z = x + jy$ where $j = \sqrt{-1}$. Some facts from complex algebra used in this paper: [13].

1. $z^* = x - jy$ (complex conjugate)

The complex exponentials $e^{jn\omega_0 t}$ together form the set [13].

$$S = \{..., e^{-j3\omega_0 t}, e^{-j2\omega_0 t}, e^{-j\omega_0 t}, 1, e^{j\omega_0 t}, e^{j2\omega_0 t}, ..., e^{jn\omega_0 t}, ...\}$$

<u>Theorem 2.1:</u> **Orthogonality of the Complex Exponentials**

The complex exponentials $e^{jn\omega_0 t}$ of the above set satisfy the orthogonality condition

$$\frac{1}{T_0}\int_{-T_0/2}^{T_0/2} e^{jn\omega_0 t} e^{jm\omega_0 t^*}\, dt = \begin{cases} 0 & \text{if } n \neq m \\ 1 & \text{if } n = m \end{cases}$$

where $T_0 = 2\pi/\omega_0$ and $e^{jm\omega_0 t^*}$ is the complex conjugate $e^{-jm\omega_0 t}$.

<u>Proof:</u>

$$\begin{aligned} \frac{1}{T_0}\int_{-T_0/2}^{T_0/2} e^{jn\omega_0 t} e^{jm\omega_0 t^*}\, dt \; &= \; \frac{1}{T_0}\int_{-T_0/2}^{T_0/2} e^{jn\omega_0 t} e^{-jm\omega_0 t}\, dt \; = \; \frac{1}{T_0}\int_{-T_0/2}^{T_0/2} e^{jn\omega_0 t - jm\omega_0 t}\, dt \\ &= \frac{1}{T_0}\int_{-T_0/2}^{T_0/2} e^{jw_0 t(n-m)}\, dt \end{aligned} \tag{2}$$

When $n \neq m$, then $n - m$ is a nonzero integer which we shall call p, and so (2) continues as

$$\begin{aligned} &= \frac{1}{T_0}\int_{-T_0/2}^{T_0/2} e^{jp\omega_0 t}\,dt \;=\; \frac{1}{T_0}\Big[\frac{1}{j\omega_0 p}e^{j\omega_0 pt}\Big]_{-T_0/2}^{T_0/2} \;=\; \frac{1}{T_0}\Big(\frac{e^{j\omega_0 p(T_0/2)}}{j\omega_0 p} - \frac{e^{j\omega_0 p(-T_0/2)}}{j\omega_0 p}\Big) \\ &= \frac{1}{T_0}\Big(\frac{e^{(j\omega_0 pT_0)/2} - e^{(-j\omega_0 pT_0)/2}}{j\omega_0 p}\Big) \;=\; \frac{1}{T_0}\Big(\frac{e^{jp\pi} - e^{-jp\pi}}{j\omega_0 p}\Big) \;=\; \frac{e^{jp\pi} - e^{-jp\pi}}{T_0 j\omega_0 p} \\ &= \frac{e^{jp\pi} - e^{-jp\pi}}{2\pi jp} \;=\; \frac{(\cos p\pi + j\sin p\pi) - (\cos p\pi - j\sin p\pi)}{2\pi jp} \;=\; \frac{\cos p\pi + j\sin p\pi - \cos p\pi + j\sin p\pi}{2\pi jp} \\ &= \frac{2j\sin p\pi}{2\pi jp} \;=\; \frac{\sin p\pi}{p\pi} \;=\; 0 \end{aligned} \tag{3}$$

On the other hand, when $n = m$ then (2) continues as

$$\begin{aligned} &= \frac{1}{T_0}\int_{-T_0/2}^{T_0/2} e^{jw_0 t(n-m)}\,dt \;=\; \frac{1}{T_0}\int_{-T_0/2}^{T_0/2} e^{j\omega_0 t(0)}\,dt \;=\; \frac{1}{T_0}\int_{-T_0/2}^{T_0/2} e^{0}\,dt \;=\; \frac{1}{T_0}\int_{-T_0/2}^{T_0/2} dt \\ &= \frac{1}{T_0}[t]\Big|_{-T_0/2}^{T_0/2} \;=\; \frac{1}{T_0}\Big(\frac{T_0}{2} + \frac{T_0}{2}\Big) \;=\; \frac{T_0}{T_0} \;=\; 1 \end{aligned} \tag{4}$$

And the proof is complete. Q.E.D. [13].

2.4.4 Fourier Series

Using this property of orthogonality, let $f_p(t)$ be a periodic function with period T_0, and assume it can be expressed as an infinite sum of complex exponentials, that is, that

$$f_p(t) \;=\; \sum_{n=-\infty}^{\infty} F(n)e^{jn\omega_0 t} \tag{5}$$

The complex exponentials on the right-hand side all repeat at least once whenever t is increased by T_0, so both sides are periodic with period T_0. To obtain the constants $F(n)$, we first multiply both sides by $e^{-jm\omega_0 t}$ and integrate

$$\begin{aligned} f_p(t) \;&=\; \sum_{n=-\infty}^{\infty} F(n)e^{jn\omega_0 t} \\ \frac{1}{T_0}\int_{-T_0/2}^{T_0/2} f_p(t)e^{-jm\omega_0 t}\,dt \;&=\; \frac{1}{T_0}\int_{-T_0/2}^{T_0/2}\sum_{n=-\infty}^{\infty} F(n)e^{jn\omega_0 t}e^{-jm\omega_0 t}\,dt \\ \frac{1}{T_0}\int_{-T_0/2}^{T_0/2} f_p(t)e^{-jm\omega_0 t}\,dt \;&=\; \sum_{n=-\infty}^{\infty} F(n)\Big[\frac{1}{T_0}\int_{-T_0/2}^{T_0/2} e^{jn\omega_0 t}e^{-jm\omega_0 t}\,dt\Big] \end{aligned} \tag{6}$$

By Theorem 2.1;

when $n \neq m$

$$\frac{1}{T_0}\int_{-T_0/2}^{T_0/2} f_p(t)e^{-jm\omega_0 t}\,dt \;=\; \sum_{n=-\infty}^{\infty} F(n)\,[0] \;=\; 0 \tag{7}$$

when $n = m$

$$\frac{1}{T_0}\int_{-T_0/2}^{T_0/2} f_p(t)e^{-jm\omega_0 t}\,dt \;=\; \sum_{n=-\infty}^{\infty} F(n)\,[1] \;=\; F(m) \tag{8}$$

Then replacing m with n we get

$$F(n) \;=\; \frac{1}{T_0}\int_{-T_0/2}^{T_0/2} f_p(t)e^{-jn\omega_0 t}\,dt \tag{9}$$

[13]

We can summarize all this into

<u>Theorem 2.2:</u> Complex Fourier Series for Periodic Functions

Let $f_p(t)$ be periodic with period T_0, defined analytically. Then it can also be represented by the infinite series of complex exponentials

$$f_p(t) \;=\; \sum_{n=-\infty}^{\infty} F(n)e^{jn\omega_0 t} \tag{10}$$

where the coefficients $F(n)$ can be found from the analytical definition of $f_p(t)$ as follows:

$$F(n) \;=\; \frac{1}{T_0}\int_{-T_0/2}^{T_0/2} f_p(t)e^{-jn\omega_0 t}\,dt \qquad (\forall n) \tag{11}$$

The beginning of section 2.2 to now constitutes the basis for what will be covered in the rest of the paper. To summarize parts of theorem 2.2

- Equation (10) is called the **synthesis equation**.
- Equation (11) is called the **analysis equation**.
- The constants $F(n)$ are called the **complex Fourier coefficients** [13].

2.4.5 Fourier Transform

Until now, we have looked exclusively at periodic functions of which we can create a Fourier series representation. The question arises if single pulse functions which do not repeat over a period can be represented by Fourier series as well. The side of Fourier analysis dealing with single pulses primarily deals with the **Fourier transform**.

Given a single pulse, we can think of it as being embedded in a single period of a periodic function. We can then have the period T_0 tend towards infinity. Even though we have a periodic function, because the period tends toward infinity we essentially eliminate all other pulses but our original single pulse [13]. There is a great deal of complexity beyond this, but for the purposes of this paper, we will move straight into theorem 3.1.

<u>Theorem 3.1:</u> Fourier Transform for a Single Pulse

(provided that the integrals exist:)

Synthesis:

$$f(t) \;=\; \frac{1}{2\pi}\int_{-\infty}^{\infty} F(\omega)e^{j\omega t}\,d\omega \tag{12}$$

Analysis:

$$F(\omega) \;=\; \int_{-\infty}^{\infty} f(t)e^{-j\omega t}\,d\omega \tag{13}$$

Equation (13) shows that a time domain function $f(t)$ can be **analyzed** to give an associated frequency domain function $F(\omega)$ that contains all the information in $f(t)$. $F(\omega)$ is known as the **Fourier transform** of $f(t)$. Likewise, in equation (12) we can see that we can **synthesize** a pulse given it's Fourier transform function. $f(t)$ then is called the **inverse Fourier transform** of $F(\omega)$ [13].

An amazing feature of the Fourier transform/ inverse Fourier transform is they can be used to operate on all pulse cases. One-time pulses of finite duration, one-time pulses of infinite duration, and periodic functions that inherently have infinite duration can all be analyzed/ synthesized.

2.5 Sampling

In order to store music and sound on computers, the entire signal must be broken down into a digital format. This consists of sampling the signal at a regular interval, essentially converting the continuous signal into a discrete signal. Saying $s(t)$ is a signal input function, the sampled version of it is a list of $s(t)$ values taken every T seconds. $s(nT)$ is the sampled function, where$T$ is the **sampling period** and n is an integer. The average number of samples taken in one second is the **sampling frequency** $f_s = 1/T$.

2.6 Discrete Fourier Transform

Combining the concepts of the continuous Fourier Transform in section 4.1 with sampling, there is a Fourier Transform that works on sampled functions. Before jumping in, there is some background math that needs to be covered.

Theorem?:

Analysis:

$$F_k = \sum_{n=-\frac{N}{2}+1}^{N/2} f_n e^{-j2\pi nk/N} \tag{14}$$

Synthesis:

$$f_k = \sum_{n=-\frac{N}{2}+1}^{N/2} F_n e^{j2\pi nk/N} \tag{15}$$

(more to be added)

3 Methodology, Algorithm, Etc.

3.1 Time and Frequency Domain Analysis

3.1.1 Time Domain of a Periodic Signal

(more to be added)

3.1.2 Time Domain of a Periodic Signal

(more to be added)

3.1.3 Time Domain of a Real Time Signal

In order to visualize music, the audio signal must be broken down so that various aspects of the sound can be analyzed. The most sensible first steps would be to look at the musical signal in the time and frequency domains. The time domain can be easily obtained without much math with the help of a python library. Using the library PyAudio, the microphone can be accessed and sound input can be read and manipulated. The library takes in the microphone data as samples and graphs the samples in real time. The x and y axes are time and amplitude. Figure 7 shows work done so far in real-time sound analysis. The code starts off by opening a PyAudio "stream." This object takes in microphone data and organizes it into chunks. The chunks are loaded into a buffer and the graphing library, matplotlib.pyplot, then takes the buffer and graphs it. There is also some data manipulation to get the microphone raw data into a graph-able format but details are unimportant for the scope of this paper. The Python, PyAudio, and PyPlot versions used here and throughout the rest of the paper are 3.7.7, 0.2.11, and 3.5.1 respectively [14] [15] [16]. Figure 8 shows another such graph.

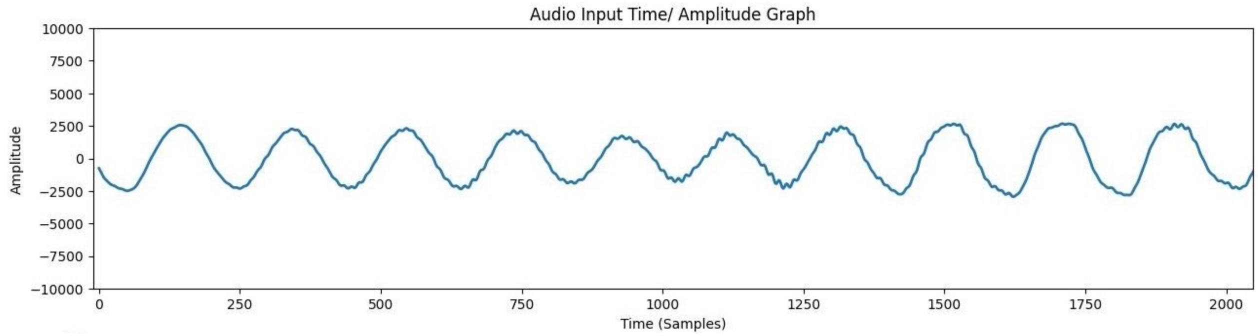


Figure 7: A graph of the amplitude over time of the author singing a note somewhere between G4 and A4 [7]

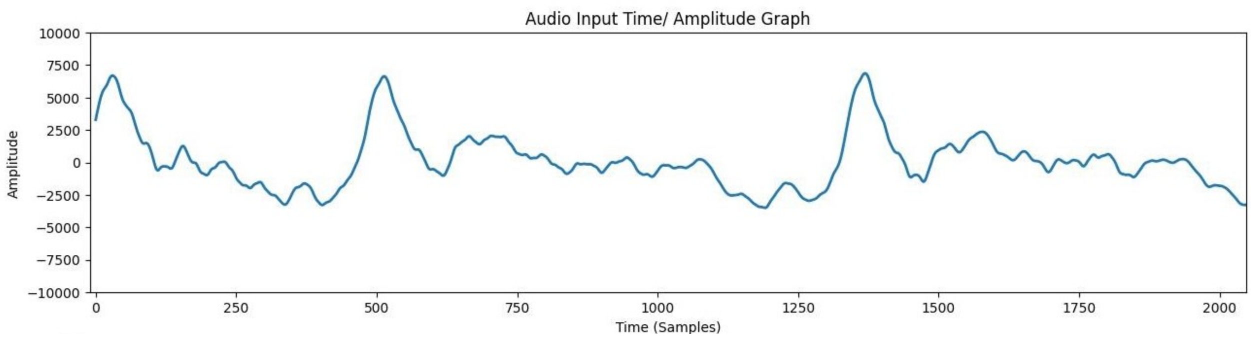


Figure 8: A graph of the amplitude over time of the author and the author's friend trying to harmonize.

3.1.4 Frequency Domain of a Real Time Signal

The next step in this direction is to look at the frequency domain. Taking the same exact signals as shown in figures 7 and 8 a library called SciPy can be used to go from the time domain to the frequency domain. With the fast Fourier trasform (FFT) build into SciPy, the frequencies that make up the signals of figures 7 and 8 can be extracted. This was a bit more complicated than the previous analysis but luckily all of the mathematical intricacies are handled by SciPy. [17]. Figures 9 and 10 show the same note and harmony done by the author but instead of plotting the time on the x-axis, the frequency is plotted instead. This is how the author approximated what note was being sung.

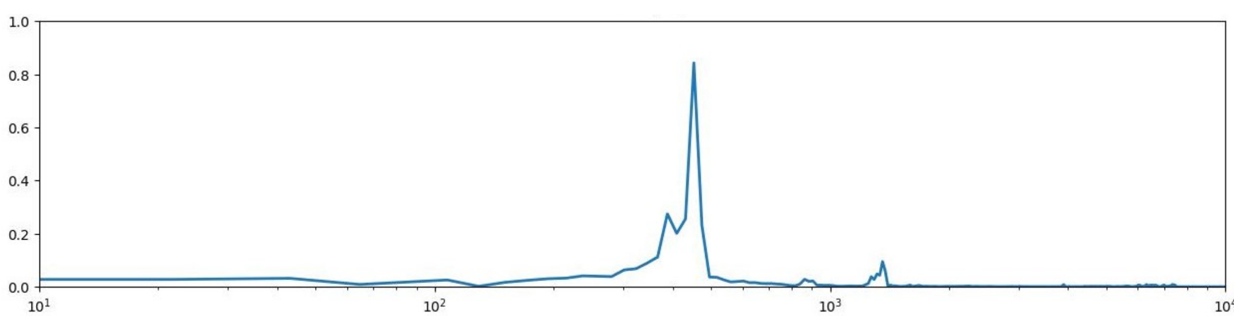


Figure 9: A graph of the frequencies sung by the author in figure 7

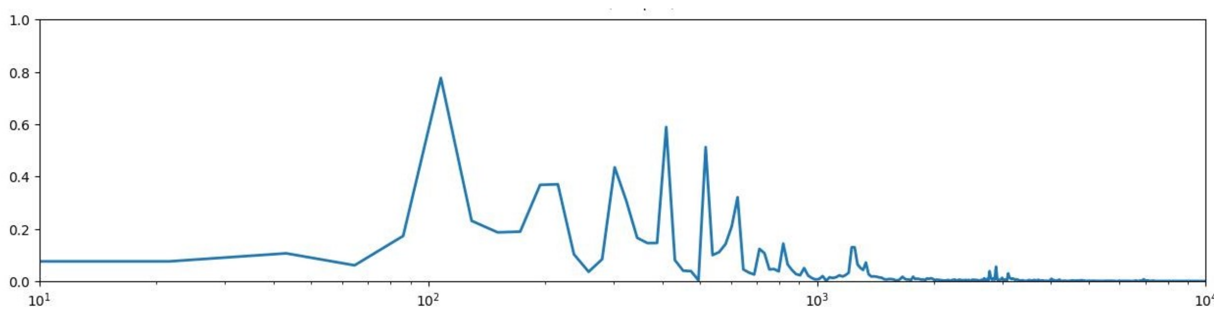


Figure 10: A graph of the frequencies sung by the author and author's friend's attempted harmony in figure 8

(more to be added)

4 Expected Outcome

The expected outcome of this project is to develop a computer program which can take in microphone input, analyze it using Fourier analysis, and then display some form of colorful shapes which represent the musical input. I do not expect the program to be complete by the end of the semester and my main focus of developing this project is in understanding the mathematics behind Fourier analysis and how that math is used in programming, specifically python.

5 Conclusion

(more to be added)

6 Acknowledgements

A Special Thank You To:

Dr. Kwadwo Antwi-Fordjour and Dr. Brian Toone for being my project advisors.

Samford University for my education.

My Family for supporting my education.

Alan Crisologo for being the "author's friend." (See Figure 4)

† JMJ † (Jesus, Mary, Joseph)

References

[1] Merriam-Webster. Hertz., 2022. In Merriam-Webster.com dictionary.

[2] Howard Boatwright. *Introduction to the Theory of Music*. W. W. Norton and Company, Inc., 1956.

[3] Nathan Lenssen and Deanna Needell. An introduction to fourier analysis with applications to music. *Journal of Humanistic Mathematics*, 4(1):72–91, 2014.

[4] Carl R. (Rod) Nave. Sound waves in air, hyperphysics. Digital image from HyperPhysics website, 2001. Hosted by Georgia State University.

[5] Nicholas Temperley. Tuning and temperament. 2007.

[6] Larry Jacobsen. Just vs equal temperament chart, 2012.

[7] B. H. Suits. Physics of music - notes, tuning, 2022. Website hosted by the Physics Department of Michigan Technological University.

[8] Wikipedia contributors. Concert pitch from wikipedia, the free encyclopedia, 2022. Online; accessed 3-March-2022.

[9] H. P. Hsu. *Fourier Analysis*. Simon and Schuster, New York, 1970.

[10] Wikipedia contributors. Sine wave — Wikipedia, the free encyclopedia, 2022. [Online; accessed 3-March-2022].

[11] Wikipedia contributors. Angular frequency — Wikipedia, the free encyclopedia, 2021. [Online; accessed 3-March-2022].

[12] P. C. Shields. *Elementary Linear Algebra*. Worth Publishers, New York, 1968.

[13] Norman Morrison. *Introduction to Fourier Analysis*. John Wiley and Sons, Inc., 1994.

[14] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3): 90–95, 2007.

[15] People.csail.mit.edu. Pyaudio documentation - pyaudio 0.2.11, 2016.

[16] Guido Van Rossum and Fred L. Drake. *Python 3 Reference Manual*. CreateSpace, Scotts Valley, CA, 2009. ISBN 1441412697.

[17] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, St'efan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, Ilhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antonio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. Scipy 1.0: Fundamental algorithms for scientific computing in python. *Nature Methods*, 17:261–272, 2020.