Modeling 8

Ryan Swope

April 20, 2021

1 Introduction to the Fourier Transform

The Fourier Transform is an especially useful function which take in a time-series data set and returns the same data in frequency space. In other words, it decomposes functions into their constituent frequencies. The Fourier transformation, in terms of an angular frequency ω, can be formulated as:

$$H(\omega) = \int_{-\infty}^{\infty} h(t) e^{i\omega t} dt \tag{1}$$

Similarly, there is an inverse Fourier transform (whose affect on data can hopefully be implied), which is formulated as:

$$h(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} H(\omega) e^{-i\omega t} d\omega \tag{2}$$

These can then be repackaged in a discrete form so that they can be applied to real data. There is also the FAst Fourier Transform (FFT) which computes the Fourier transform in chunks. Most significantly, it reduces the number of necessary operations from N^2 to $N \log N$, making the FFT significantly faster the the normal Fourier transform. To get a feel for the Fourier transform, we first applied it to some well known functions, as seen in 1. These function all represent signals often seen in physics and engineering. We can then take the Fourier transform, Figure 2, and the inverse Fourier transform, Figure 3, of these functions. As is expected, they resemble each other. We can also take the power spectral density of the functions, which will give us the power of the signal as a function of frequency (Figure 4).

2 Aliasing

Aliasing is an effect observed in signal processing that occurs when the sample rate of the signal is more than half a wavelength, known as the Nyquist frequency. The result is an inability to distinguish frequencies from each other, and this can be observed in the Fourier transform. I created a signal from 0 to 20π by adding the cosine(x) and cosine(x/3). Then, I sampled that signal again

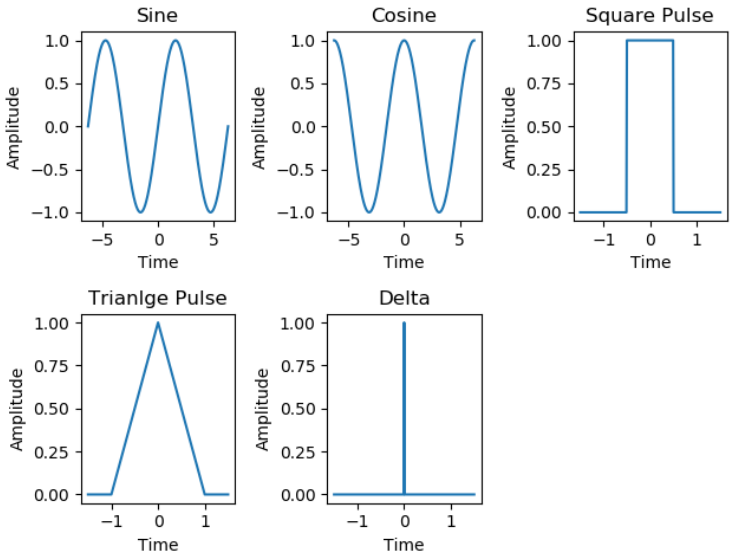

Figure 1: The functions used

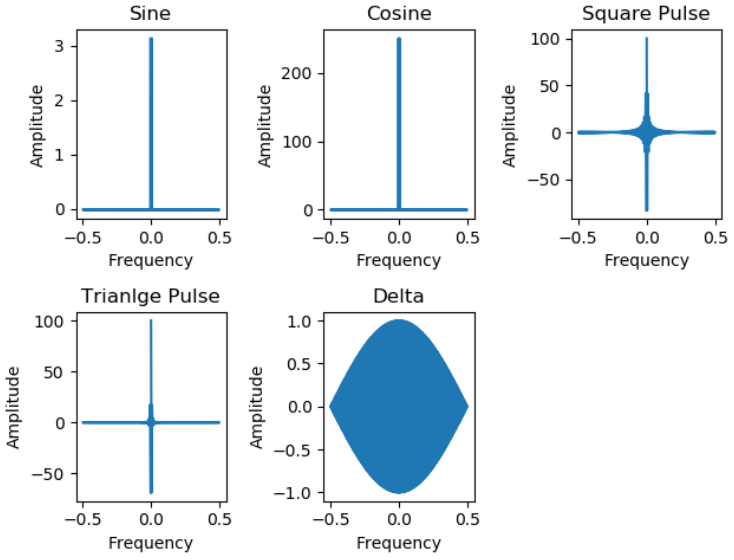

Figure 2: The Fourier transform of the functions

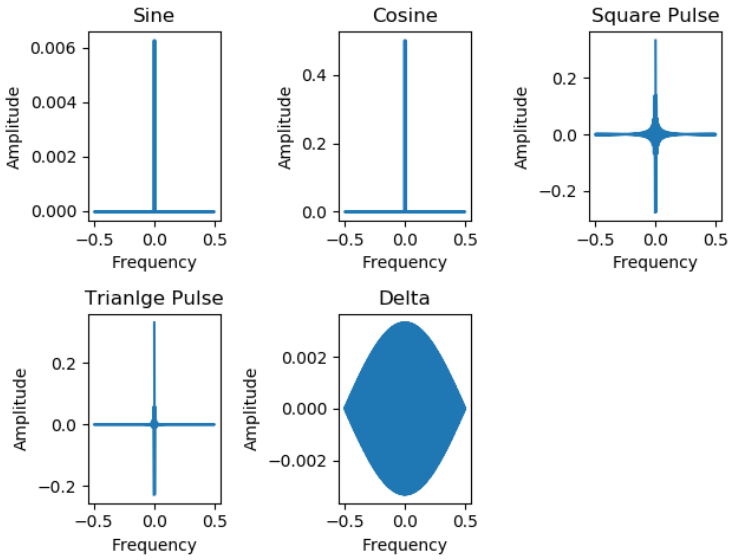

Figure 3: The inverse Fourier transform

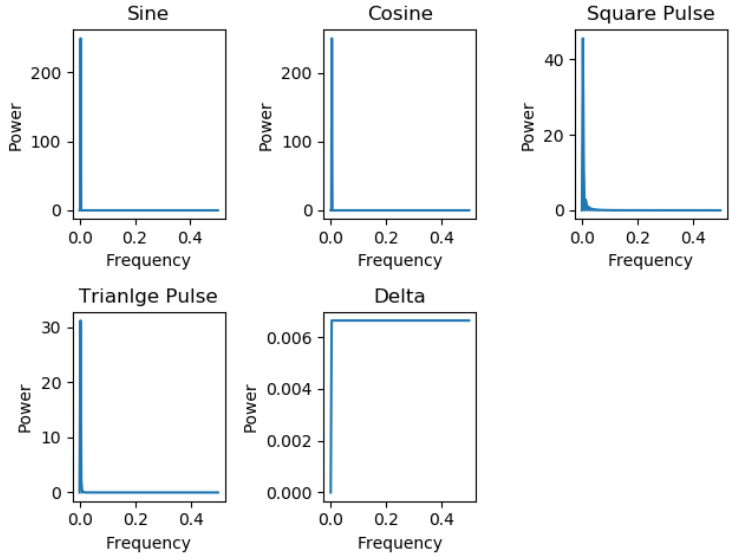

Figure 4: The power spectral density

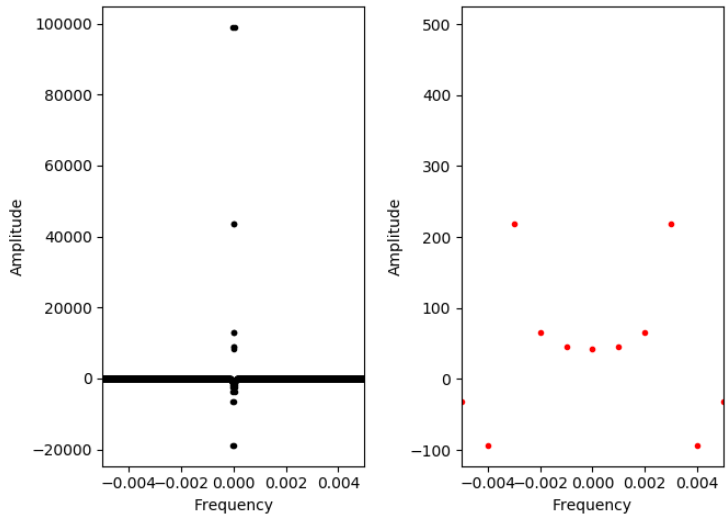

Figure 5: The Fourier transform of an unaliased (black) and aliased (red) signal

at a rate 200x slower than the initial data. The resulting Fourier transforms are in Figure 5. As a result of the aliasing, the data does not accurately represent its constituent frequencies. The result is a broader and less intense distribution of constituent frequencies in frequency space, which is exactly what we see in those plots. Similarly, we can calculate the power spectral density of the functions. Again due to aliasing, the power of the signal is distributed amongst a much larger range of frequencies than it ought to be, as seen in Figure 6.

3 Guitar Chord Analysis

One of the most common uses of the Fourier transform is the decomposition of musical chords into their constituent notes. The frequencies of the notes used can be used and then the corresponding musical note can be found to reconstruct the chord. Applying this to the guitar chord provided, we see frequencies of about 82, 164, and 247. This corresponds to the notes E_2, E_3, and B_3. There also appear to be overtones of E_4 and $G_4^{\#}$. All this suggests an E major chord, and indeed when I played those notes on my guitar, the sound was exactly the same (without the distortion).

4 Bach

The effects of undersampling music are effectively aliasing, as undersampling (i.e. sampling at too slow of a rate) is what causes the inability to distinguish different frequencies. When applied to something like music, the effect is akin

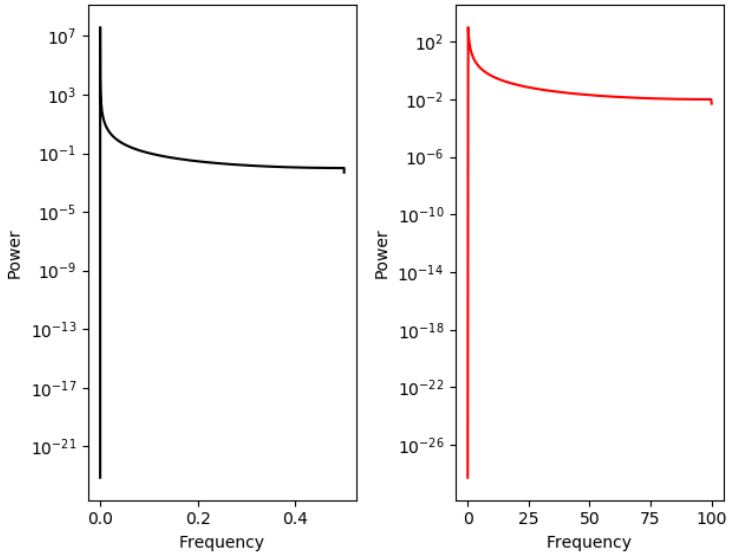

Figure 6: The power spectral density of an unaliased (black) and aliased (red) signal

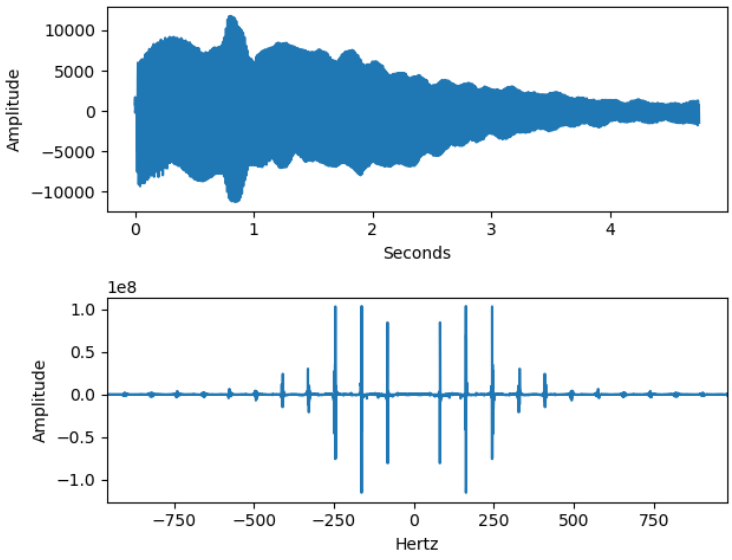

Figure 7: The waveform and Fourier transform of a guitar chord. Its constituent frequencies are clear.

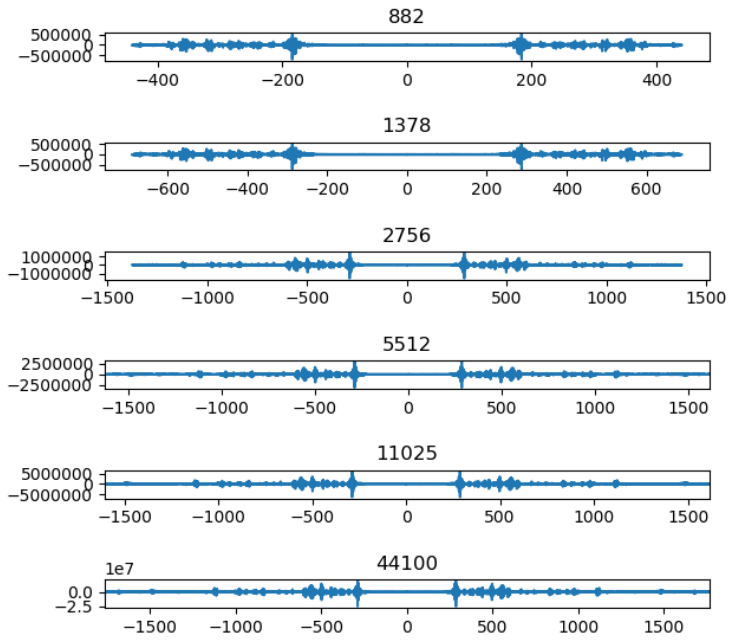

Figure 8: Bach's Partita and undersampling. I coudn't get the axis labels right in python, but they are frequency (HZ) and amplitude.

to the musicians on the titanic playing while continuing to sink underwater. The more undersampled, the farther down they sound. Plotting undersampled portions of the music reveals the same patterns as we saw during aliasing: a broadening and shortening of the constituent frequencies. In this example a segment from Bach's Partita was sampled as frequencies between 882Hz and 44100Hz. At 11025 Hz and above, the music sounds close enough to a live performance and is recognizable, but below this point it is too jumbled too sound like the real deal.

5 Dodger Stadium

We can also analyze the periodicity of people's movements using the Fourier transform. We were given car density data as well as event times and attendance. Plotting the attendance of the car density does reveal spikes during high-attendance events, although I was unable to quantify this.

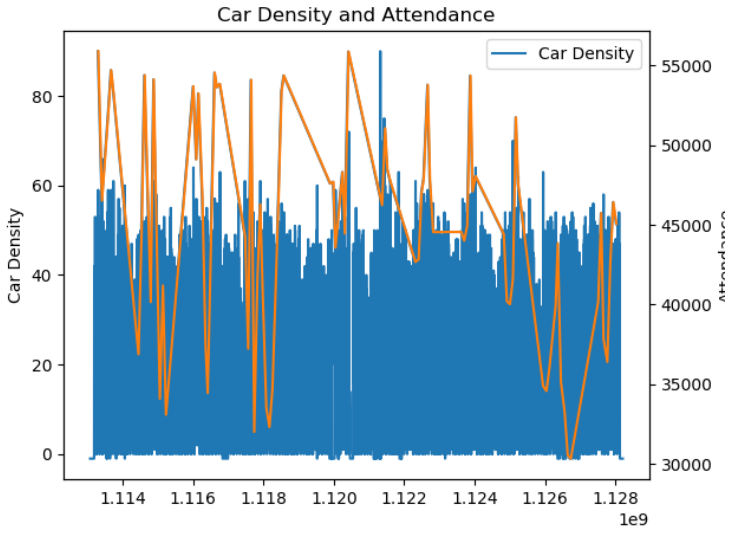

Figure 9: Attendance and Car Density

However, the Fourier transform did reveal something (albeit an expected result) about car densities, and it's that they are periodic in 24 hour intervals. This suggests that with relative frequency the same number of cars take the Dodger Stadium exit at the same time every day — in other words, rush hour traffic!

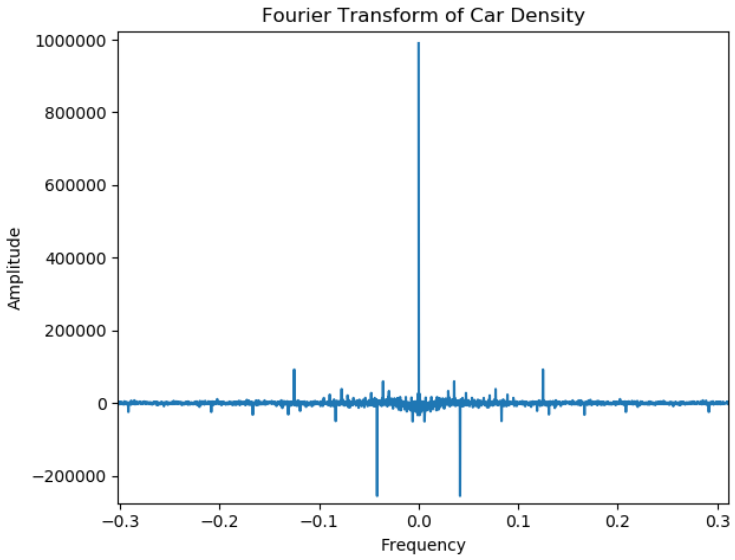

Figure 10: Fourier transform of car density data

6 Boiling Water

Another important application of the Fourier transform is correlation, and in our case specifically autocorrelation, or how closely a function matches itself across its time range. This is done using the Correlation theorem, which says that the product of the Fourier transform of one function and the complex conjugate of the Fourier transform of another function is equal to the Fourier transform of the correlation between the two, so we simply take the inverse transform of the product to get the correlation. In the case of autocorrelation the same process is applied, only with the same function twice. As an example we can find the autocorrelation of the sound of boiling water.

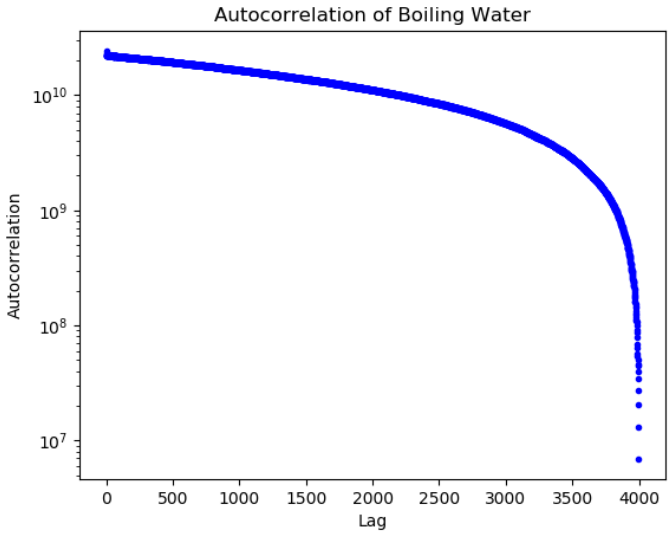

Figure 11: Autocorrelation of boiling water

What this plot reveals is that boiling water is most similar to itself with no lag (obviously). However, the sound of boiling water is relatively constant and self-similar, and that is evident in the fact that the correlation remains high over much of the lag-space; in fact, a logarithmic scale is required to properly capture its magnitude. The slow decline is due to the lag pushing the end of the data past the data it's being compared to, so there is nothing to correlate it with. Only when the lag approaches the length of the sound byte does the correlation drop off to significantly lower levels.

7 Sunspots

We can apply both the Fourier transform and autocorrelation to look for periodicity in data, keeping in mind that the Fourier transform finds constituent frequencies while autocorrelation finds periodicity across the entirety of time-space. Sunspots are good candidates for such analysis, because they appear and disappear with known frequencies, and that frequency has been constant since observations started. Plotting the Fourier transforms, we find periodicity at 11 years, or 132 months, as expected. These values were found by taking the reciprocal of the frequency value at which extrema occured.

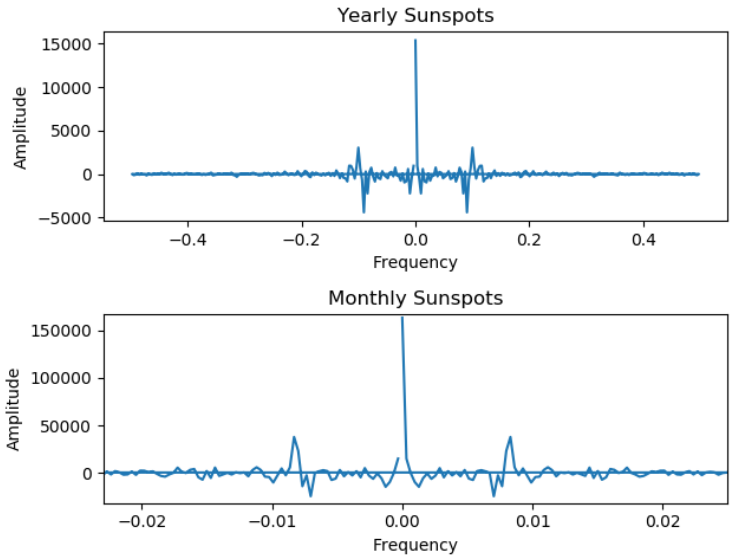

Figure 12: Fourier transform of sunspot data

Taking the autocorrelation of the data sets reveals the same periodicity of 11 years and 132 months in the peaks of the autocorrelated data. These values were found by finding the difference between lag positions of peaks in the autocorrelation. This confirms with greater certainty that sunspots do in fact appear in 11 year cycles (this does not account for the polarity flip, which extends the full cycle to once every 22 years).

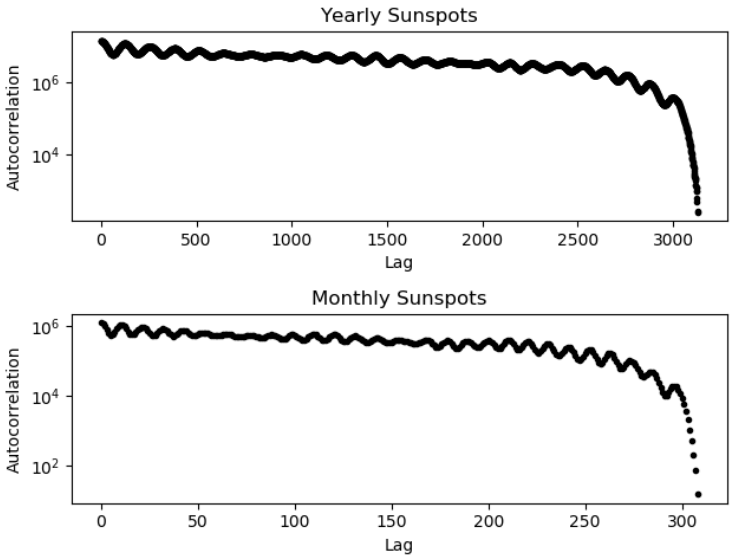

Figure 13: Autocorrelation of sunspot data