Applying the Fourier Transform and Autocorrelation to Sound

Adam Gao

December, 2017

1 Introduction

Sound is a wave. A wave has characteristics in both frequency and variation over time, which can be analyzed computationally.

Tohyama emphasizes the categorization of formulation between temporal and spectral characteristics of sound throughout his book *Waveform Analysis of Sound* and a la chapter titles 'Temporal and Spectral Characteristics of Sequence' and 'Temporal and Spectral Enhancement by Sound Path'.[3] His book formulates different techniques applied to waveform analysis of sound. A combination of the auto-correlation and the fast fourier transform helps analyze these characteristics in both the time and frequency domain. In certain situations, the auto-correlation sequence can get frequency information that the fourier transform cannot, despite its traditional use, as will be explored.

2 Mathematical Methods

2.1 Sound as a sequence:

Raw sound is a function over a continuous time domain. For methods of analysis in the temporal domain to be applied, sound data must be taken as a discrete sequence instead.[3] For the purpose of this project, such methods are largely referenced from *Waveform Analysis of Sound* by Mikio Tohyama, where sound is indeed taken to be a discrete sequence.[3]

Sound as discrete sequences will generally be denoted as a function $x(n)$, with respect to the sequence number n.

2.2 Fourier Transform (brief introduction)

Briefly speaking, a waveform can be expressed as a summation of eigenfunctions of different frequencies. The fourier transform of a sequence returns a new sequence in the spectral domain, where each element of the sequence is a coefficient corresponding to a frequency, in increasing order of frequencies.

Tohyama defines a transform more generalized than the fourier transform, called a z-transform.[3] The z-transform converts a time domain into a frequency domain and the fourier transform parameterizes the z-transform on to a unit circle with expression $e^{i\Omega}$, where Ω denotes a normalized angular frequency.[3] In other words, $e^{i\Omega}$ is the eigenfunction of the fourier transform.

Denote the z transform of a sequence $a(n)$ as follows[3]

$$A(z^{-1}) = \sum_n a(n)z^{-n} \tag{1}$$

The fourier-transform takes different forms and notations. For general purposes, denote the fourier-transform of a sequence $a(n)$ as $A(e^{-i\Omega})$.

2.3 Fourier Transform (phase and maginitude)

It is important to note that the full imaginary form of the fourier transform is used in mathematical formulation, with the real part corresponding to an even function and the imaginary part corresponding to an odd function. Differentiating between the real and imaginary fourier transform will be relevant to later discussion in wave analysis.

2.4 Cross-correlation, Autocorrelation:

Cross-correlation quantifies similarity between two sequences at different amounts of displacement between said sequences.

The cross-correlation sequence $r_c(n)$ between two sequences $a(n)$ and $b(n)$ are defined by the following equation[3]

$$r_c(n) = a \bigotimes b(n) = \sum_m a(m)b(m-n) \tag{2}$$

The auto-correlation $r_a(n)$ of a sequence $a(n)$ is the cross-correlation between $a(n)$ and itself. Thus, the auto-correlation is defined as follows[3]

$$r_a(n) = a \bigotimes a(n) = \sum_m a(m)a(m-n) \tag{3}$$

2.5 Convolution:

Tohyama introduces convolution before he does cross-correlation. This is because cross-correlation can be expressed in terms of convolution.[3]

Roughly speaking, convolution is the measure of how different two sequences are at different distances. As the interest of the discussed waveform analysis only relates to convolution as convolution rates to cross-correlation, I will simply express the cross-correlation sequence with respect to the convolution sequence.

Denote the convolution sequence between $a(n)$ and $b(n)$ as $a * b(n)$.

Then[3]:

$$a \bigotimes b(n) = a * b(-n) \tag{4}$$

3 Waveform Analysis

3.1 Magnitude and Phase of Spectrum

The auto-correlation sequence $r(n)$ of $x(n)$ can be expressed in terms of the power spectral density of $x(n)$ as follows[3]:

$$R(e^{-i\Omega}) = X(e^{-i\Omega})X(e^{i\Omega}) = |X(e^{-i\Omega})|^2 \tag{5}$$

Because the power spectral density is even, the auto-correlation sequence is even. Thus, given a (purely) even sequence (Tohyama calls this null-phase spectrum), auto-correlation is not particularly useful[3].

Nonetheless, Tohyama states 'The auto-correlation, however, is useful for period analysis of a sequence"[3]. In essence, he shows that auto-correlation reveals frequency information that the fast fourier transform may not.

First, Tohyama denotes a periodic sequence of period N as follows[3]:

$$x(n) = x(n + pN) \tag{6}$$

where p is an integer.

Tohyama then states that the sequence can be written as the convolution between a single cycle of the sequence x_0 and a unit pulse train $u(n)$ as follows:

$$u(n) = x_0 * u(n) \tag{7}$$

with the unit-pulse train defined as follows[3]:

$$u(n) = \sum_p \delta(n - pN) \tag{8}$$

Tohyama then expresses the z-transform of the periodic sequence as follows[3]:

$$X(z^{-1}) = X_0(z^{-1})U(z^{-1}) = X_0(z^{-1})\delta(z^{-N} - 1) \tag{9}$$

Squaring both sides, Tohyama argues that[3]:

$$|X(e^{-i\Omega})|^2 \Delta\Omega = P_{0s}(e^{-i\Omega}) \tag{10}$$

Finally, using the autoccorrelation can be expressed[3]:

$$r(n) = 1/2\pi \int_0^{2\pi} |X_0(e^{-i\Omega})|^2 \delta(e^{-i\Omega} - 1)e^{i\Omega} d\Omega = \sum_{k=0}^{N-1} P_0(e^{-i2\pi k/N}) cos(2\pi kn/N) \tag{11}$$

Tohyama uses equation 11 to argue that 'the autocorrelation is also periodic with period N'[3]. He states 'the auto-corrrelation sequence composed of zero-phase spectral components takes its maximum always at $n = 0$, and thus helps to estimate the fundamental period more easily.'[3] What this means is that the auto-correlation sequence being an even function is maximum at 0 path difference, which helps estimate the fundamental period more easily.

3.2 Triangular Windowing

Because frequency information is time dependent, frame-wise auto-correlations are useful. Tohyama essentially states that as time lag increases, variance increases[3]. He proceeds to state that triangular windowing can decrease these effects.[3]

3.3 Single Sound Reflection

Different methods can be computationally applied to analyze reflection of sound.

Square averages and Auto-correlation:
Assuming a time sequence is the superposition of a direct sound and its reflection, its square average should depend on the cross-correlation between the direct sound and reflection sequences.[3] Tohyama denotes a direct sound $s_d(n)$ and its reflection $s_r(n) = s_d(n-m)$[3]. Thus a perfect reflection should simply be the perfect source with a discretized time translation, m. Assuming a perfect reflection, Tohyama relates the square average to the auto-correlation of the direct sound as follows, where $E[*]$ denotes ensemble average and $r_d(m)$ denotes the auto-correlation of the direct sound:

$$E[(s_d(n) + s_d(n-m))^2] = E[s_d^2(n)] + E[s_d^2(n-m) + 2r_d(m) \tag{12}$$

4 Computational Application

4.1 Code Summary

The python code has been modularized into multiple files.

Files starting with "Waveforms" include functions and sequences to be imported.

"Waveforms.py", generates different waveforms as sequences to be analyzed.

"Waveforms2.py" consists of actual sound samples data, including the sound pressure information as sequences over the temporal domain, as well as the the sound file sampling rates.

"Waveforms3.py" consists of a function which generates waveforms using frequency information. The function sineharmonics takes a list of different harmonic numbers and sums sinewaves corresponding to each harmonic, over the temporal domain. The new sequence can then be analyzed as a waveform over the temporal domain.

4.2 Python packages

All code has been typed in python.

4.2.1 Cross-correlation

The function ”numpy.correlate” from the numpy package of python may be used to return the cross-correlation of an input sequence.[1]

numpy.correlate has multiple modes: ”valid”, ”same” and ”full” which determines the array length. ”valid” returns an array length of 1.[1] ”same” returns an array of length equal to the length of the argument array.[1] ”full” returns the full cross-correlation between all possible distances (negative and positive).[1]

4.2.2 Fourier Transform

The function ”numpy.fft.rfft” uses the fast fourier transform algorithm to take the discrete fourier transform of a real array.[1] This is most relevant to waveform analysis since the sound waves are of real values.

The notes on the scipy website state that ”the negative frequency terms are just the complex conjugates of the corresponding positive-frequency terms, and the negative-frequency terms are therefore redundant.”[1] This means that taking the absolute square of the fourier transform would provide invalid power spectra. Rather, one should either just ignore the negative terms, or remove them in the graph.

4.2.3 Triangular Window

The function ”scipy.signal.triang” from the scipy package returns an triangular window in the form of an array[1]. A triangular window is a linear function that is reflected across the middle of the x axis, forming a ”triangle”. By multiplying an array by the triangular window, it seems that one can decrease the importance of temporal information at the boundaries of the frame. The following is a graphic of a triangular window plot from the scipy website:

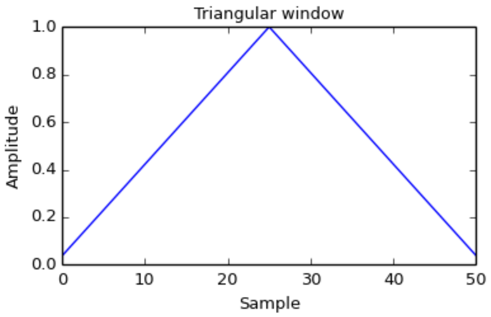

Figure 1: Triangular window taken from scipy website[1]

5 Results and Discussion

5.1 Auto-correlation Properties: Unit Pulse Trains

The file ”auto-correlation properties.py” tests the principle of equation 7, where a sine wave of multiple cycles is equivalent to the convolution between a unit pulse train of an equal amount of cycles and the sine wave for one cycle. The implication is that the auto-correlation sequence is effective at revealing fundamental frequency information.

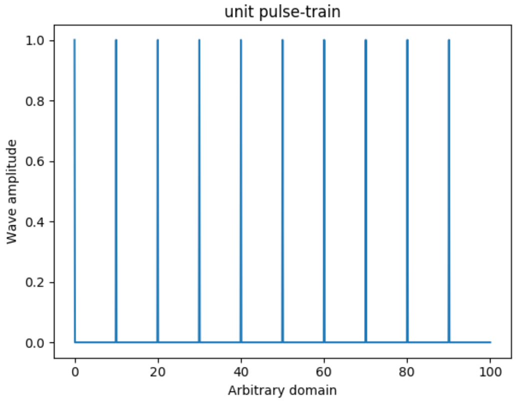

Figure 2: Unit pulse train

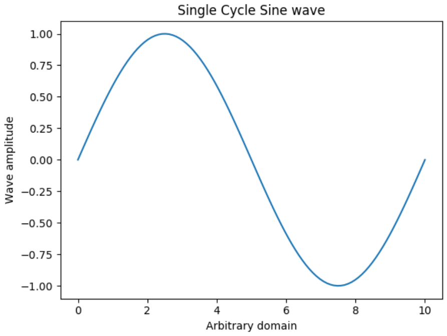

Figure 3: Sine wave of a single cycle

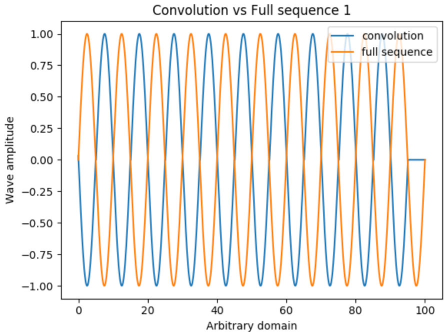

Figure 4: Convolution vs Full sequence, sine wave

Figure 2 shows a unit pulse train of period 10. Every 10 indexes the function returns 1, and in all other indexes the function returns 0.

Figure 3 shows a sine wave of a single cycle, also of period 10.

Figure 4 shows the convolution between the unit pulse train and the single cycle sine wave. Figure 4 also shows the same sine wave of period 10 plotted over multiple cycles.

As consistent with equation 7, in Figure 4, the convolution sequence and the directly plotted sine wave are identical, with an offset in the argument domain, which can be attributed to the location of the first spike in the unit pulse train.

Of course, knowing fundamental frequency of a sine wave is trivial. However, if this result can be generalized to other waveforms, then fundamental frequency information should also be clear for other waveforms. Thus, the convolution between a waveform consisting of multiple harmonics and the same unit pulse train was observed.

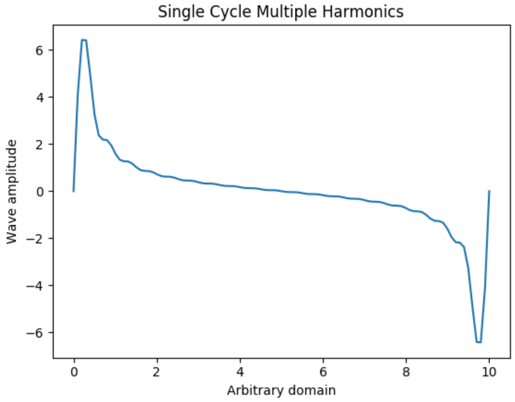

Figure 5: Single Cycle Multiple Harmonics

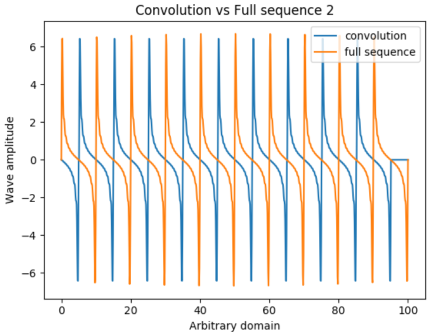

Figure 6: Convolution vs Full sequence, multiple harmonics

Figure 5 shows a signal generated using harmonic numbers from Waveforms3.py. The signal is composed of harmonics 1 through 20. The coefficient of each sine wave decreases as the harmonic number increases. This is to smooth the function. The lowest harmonic is the fundamental frequency and determines the period of the overall waveform.

Figure 6 shows the convolution between the waveform in figure 5 and the unit pulse train in figure 8. Figure 6 also shows the waveform in 5 over multiple cycles. The two waveforms appear very similar. Thus, the auto-correlation sequence as argued for in 11 should be able to reveal fundamental frequency information.

5.2 Single Sound Reflection

Briefly speaking, an attempt at confirming equation 12 was made for a sine wave superposed over a delayed instance of itself. This was done in sound_reflection.py.

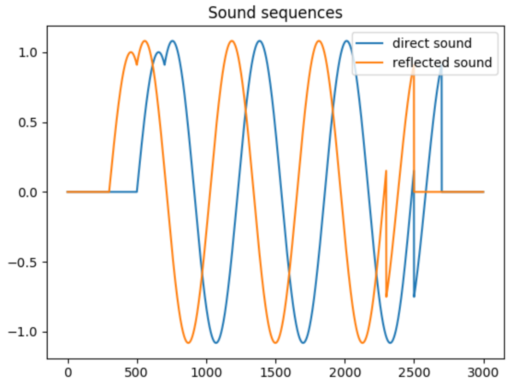

Figure 7: Waveform to be analyzed with respect to equation 12

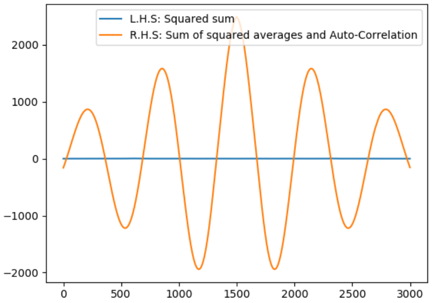

Figure 8: Comparison between left side and right side of equation 12
of figure 7

Figure 7 shows the sequence being analyzed and figure 8 shows the effective left hand sides and right hand sides of equation 12 as sequences. The ensemble average of the sequence was interpreted to be the exact sequence, since there is only one instance of a waveform.

Essentially, the two sides of the equation did not equate for the entire sequence. The auto-correlation term on the right hand side dominated the entire term, being much greater than the square of the other sequences for all indexes.

This attempt was not very fruitful and I found it a much better use of time to explore other aspects of waveform analysis as applied to acoustics.

5.3 Effect of Harmonics on Auto-Correlation

The value of auto-correlation in getting fundamental frequency information was tested using generated waveforms imported from Waveforms3.py, in the file 'sound_fundamental_frequency.py'.

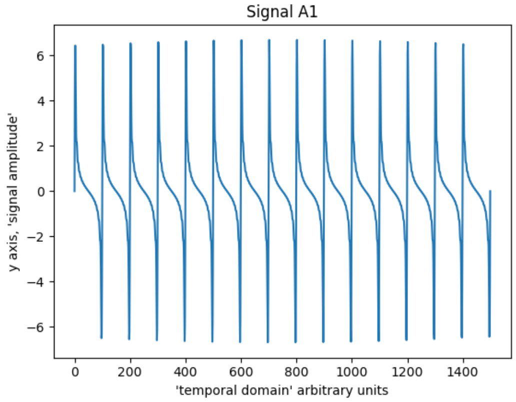

Figure 9: Signal A1

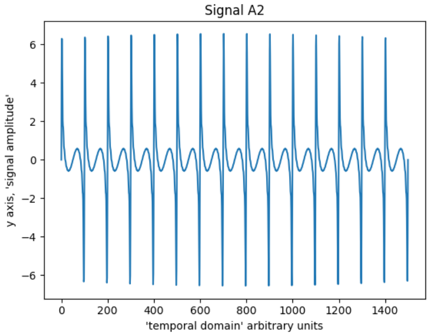

Figure 10: Signal A2

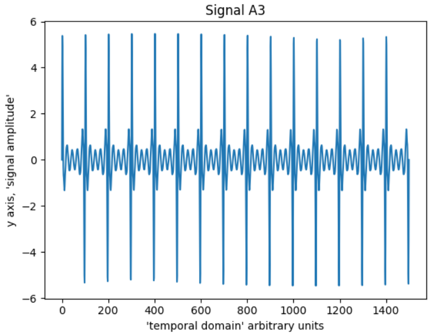

Figure 11: Signal A3

Figures 9, 10, and 11 show different signals generated using harmonic numbers. Their respective plot bins (temporal domains) have units kept in terms of bin number, rather than argument value for the sine functions. This is for ease of comparison with the auto-correlation sequences.

Figure 9 shows Signal A1, a signal composed of harmonics 1 through 20. The coefficient of each sine wave decreases as the harmonic number increases. Signal A1 is identical in form to the previously generated wave in figure 5. Thus, as was true in the previous unit pulse train study, the lowest harmonic is the fundamental frequency and determines the period of the overall waveform.

Figure 10 shows Signal A2, which is Signal A1 with the first harmonic removed. Effectively, Signal A2 is Signal A1 without its fundamental frequency. One can observe that A2 retains a similar periodic behavior as A1, with obvious spikes of same period as in A1. Only waveform has changed.

Figure 11 shows Signal A3, which is Signal A1 with the first five harmonics removed. Even so, fundamental frequency information is still very obvious and consistant with A2.

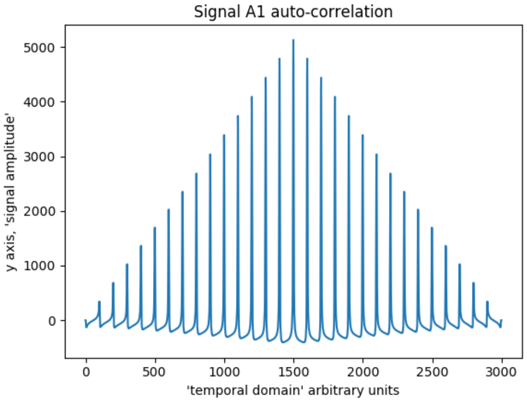

Figure 12: Signal A1 auto-correlation

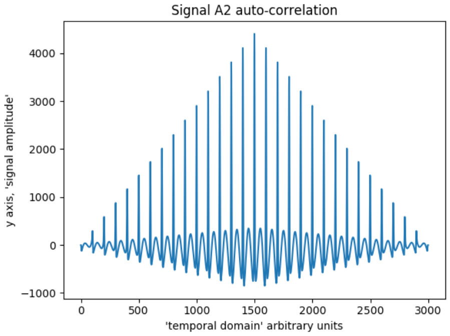

Figure 13: Signal A2 auto-correlation

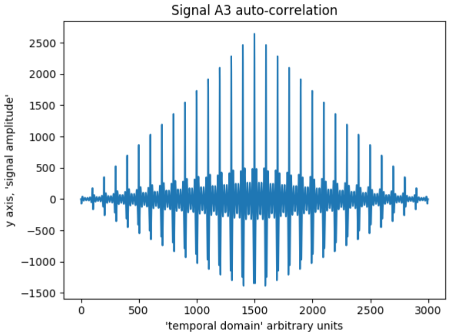

Figure 14: Signal A3 auto-correlation

Figures 12, 13, and 14 show the full auto-correlation sequences for each of their respective signals. The maximum value of the auto correlation sequence is at the center of the domain for all signals. This means that distance is located at the center of each signal's domain.

It is clear that fundamental frequency information is similarly preserved for auto-correlation sequences of all signals.

However, since fundamental frequency information is quite obvious from the raw signals, in this case the auto-correlation sequences are redundant in usage, though clearly effective.

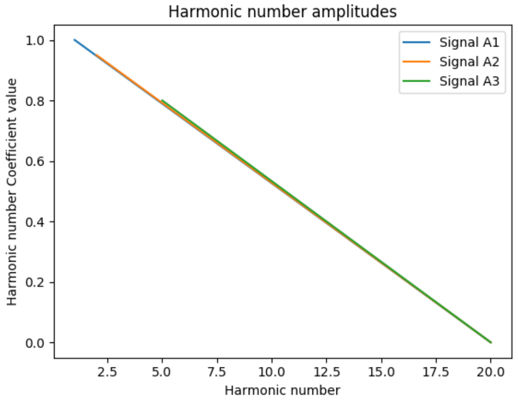

Figure 15: Harmonic Number information

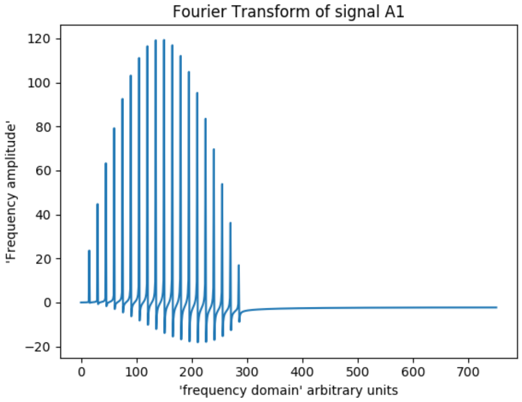

Figure 16: Signal A1 fourier transform

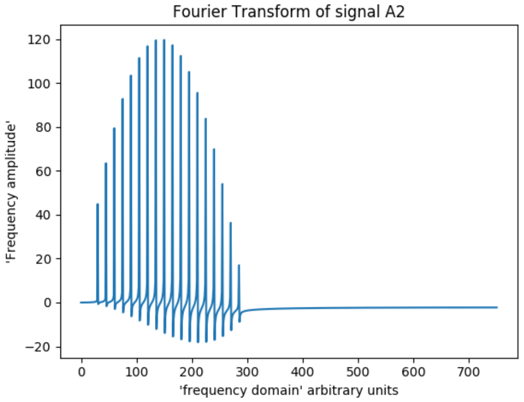

Figure 17: Signal A2 fourier transform

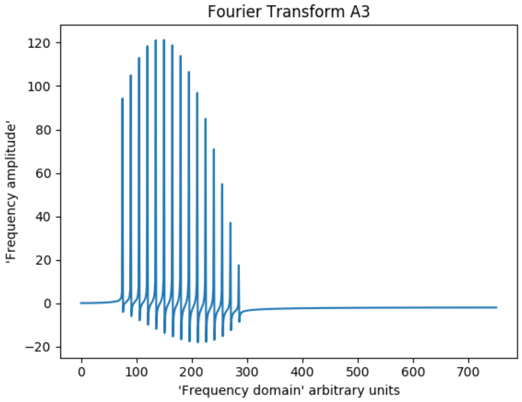

Figure 18: Signal A3 fourier transform

Figure 15 displays the amplitude of each harmonic number graphed with respect to the harmonic number. This information was directly used to generate the analyzed waveforms A1, A2, and A3.

Figures 16, 17, 18 display the fast fourier transforms of each respective signal using np.fft.rfft. There are 19 equally spaced peaks corresponding to each harmonic in 16 as expected (the 20th harmonic has a coefficient of 0). In figures 17 and 18, the respective harmonics are missing, as expected. This means that the fourier transform is not as helpful for finding the fundamental frequency in signals A2 and A3.

Contrary to what is expected, the amplitudes of the peaks are not proportionally aligned to what is shown in figure 15. In other words, the fourier transforms imply different coefficients for the harmonics, for all signals. This means that frequency information in the fast fourier transform algorithm is distorted, that the amplitude of one frequency is distorted by the existence of other frequencies. This is to say that a single sine wave has a fast fourier transform corresponding to its harmonic information, but waveforms A1, A2, and A3 do not.

5.4 Reverbated Speech

Speech samples from Mc Squared System Design Group, Inc were analyzed[2]. As stated on the company website, The sound files were ”constructed by Wade McGregor using Sound Forge software”[2].

Two sound files were analyzed in ’sound_speech.py’. They are identical speech samples in situations of different reverberations (echo levels). The speech sample with no reverb will be referred to as dry speech. The speech sample with 2 seconds of reverb will simply be referred to as the reverberated sample.

Frames of several milliseconds were taken from the sound files so that auto-correlation sequences could be taken. They were processed through a triangular window before the auto-correlation sequences were taken.

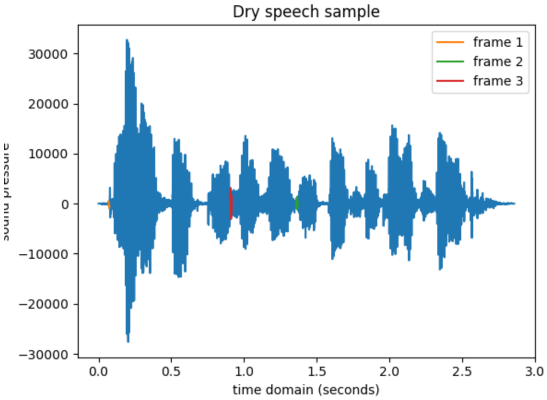

Figure 19: Dry speech sample

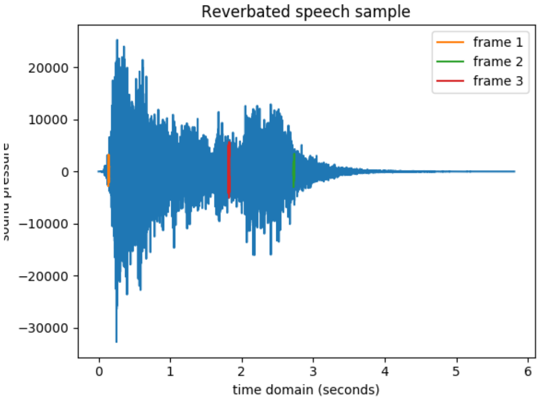

Figure 20: Reverberated speech sample

Figures 19 and 20 show plots of the soundwaves over the temporal domain as extracted from the sound files, for dry speech and reverberated speech respectively. Additionally, the frames that were taken and analyzed as temporal sequences are highlighted. The highlighted parts are very small, as the frames must be short. If the frames are too long, the frequency information will vary too much for any meaningful information to be extracted.[3] Also, the reverberated sample is longer than the dry sample. This is probably to account for the sound of the echoes in the reverberated sample.

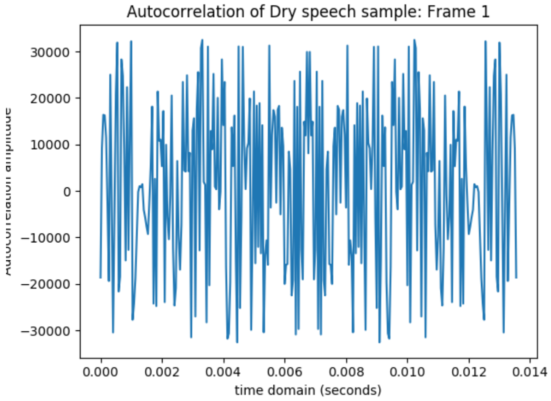

Figure 21: Auto-correlation of frame 1, dry, raw

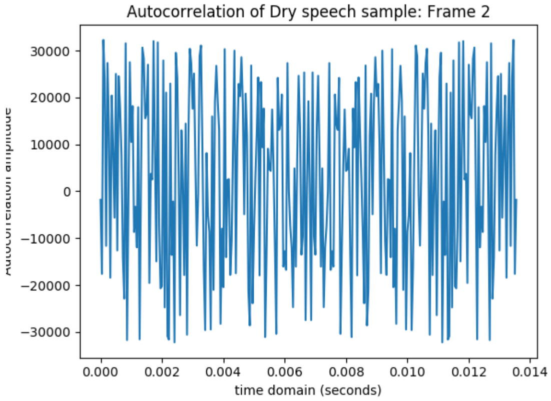

Figure 22: Auto-correlation of frame 2, dry, raw

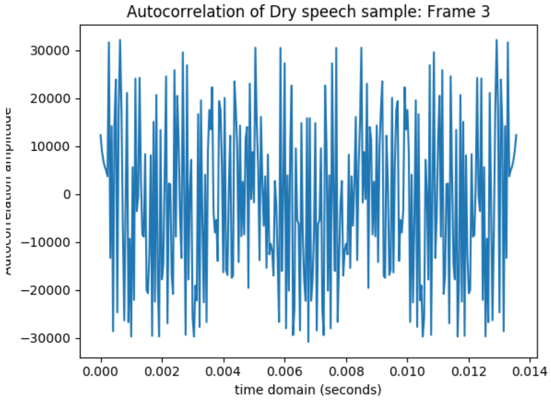

Figure 23: Auto-correlation of frame 3, dry, raw

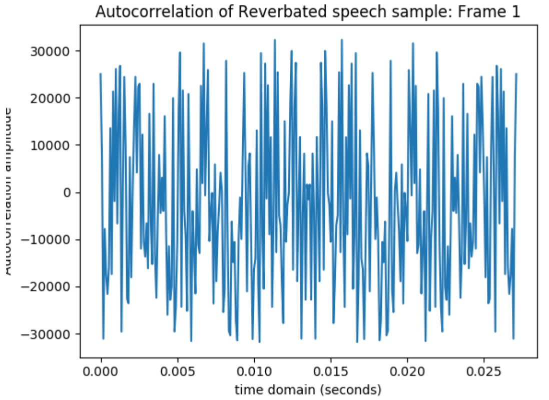

Figure 24: Auto-correlation of frame 1, reverb, raw

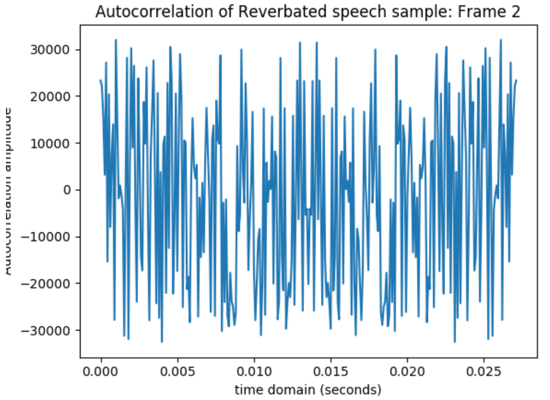

Figure 25: Auto-correlation of frame 2, reverb, raw

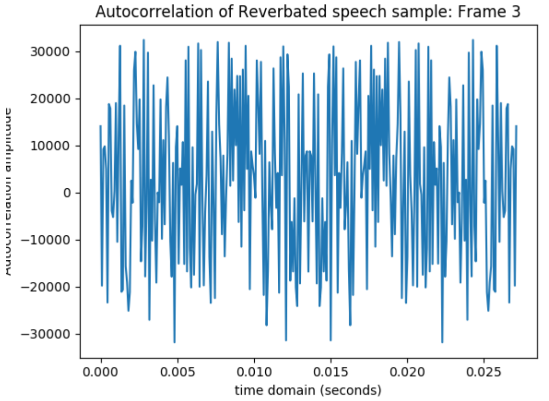

Figure 26: Auto-correlation of frame 3, reverb raw

Figures 21 through 26 show the auto-correlation sequences of the direct frames with out being temporally filtered through a triangular window. The sequences all seem rather erratic.

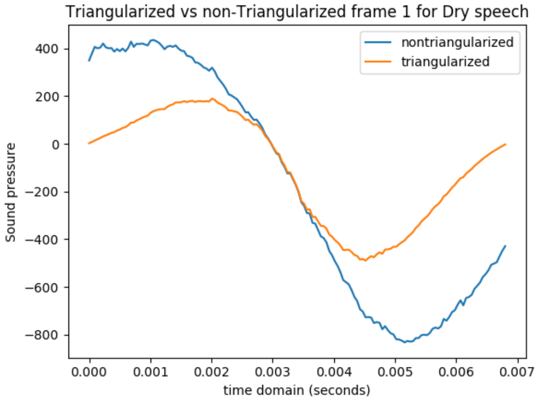

Figure 27: Frame 1 of Dry speech sample

Figure 27 shows the triangular windowed and the non-windowed versions of the raw soundwave in frame 1 of the dry speech sample. On a qualitative level, one cannot discern any meaningful frequency information from the speech sample shape. A human voice is a complex arangement of frequencies, after all. It seems that triangular windowing decreases information near the boundaries. In theory, this allows the auto-correlation sequence to be effectively used.

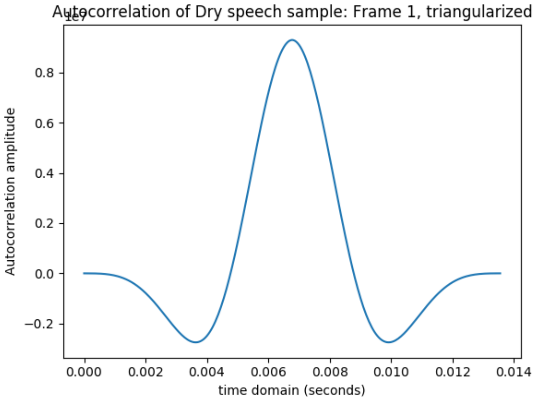

Figure 28: Auto-correlation of frame 1, dry, windowed

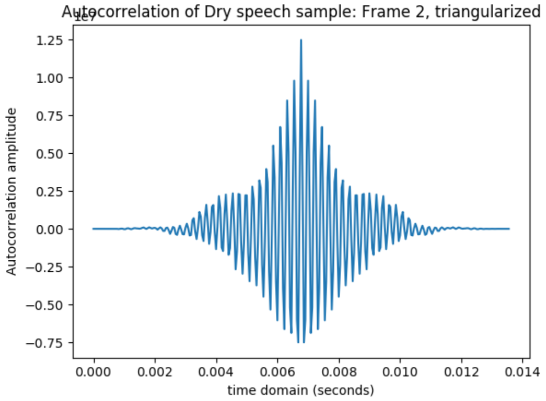

Figure 29: Auto-correlation of frame 2, dry, windowed

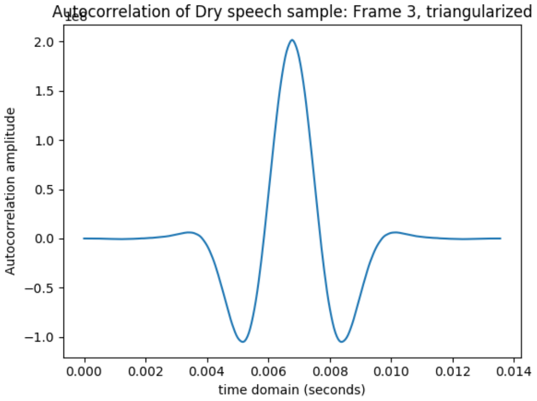

Figure 30: Auto-correlation of frame 3, dry, windowed

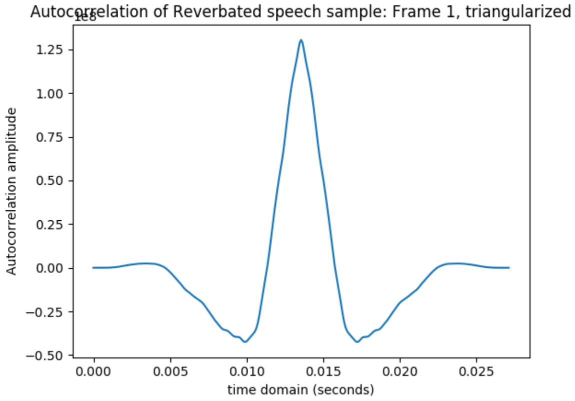

Figure 31: Auto-correlation of frame 1, reverb, windowed

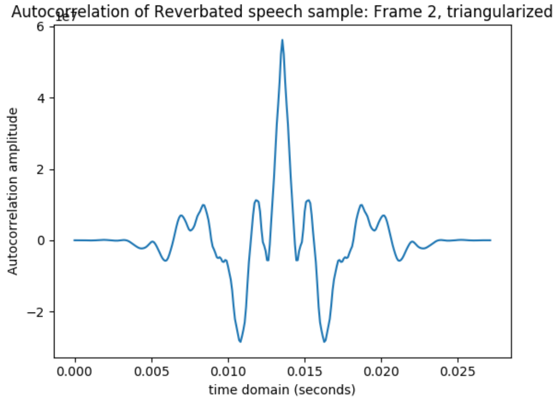

Figure 32: Auto-correlation of frame 2, reverb, windowed

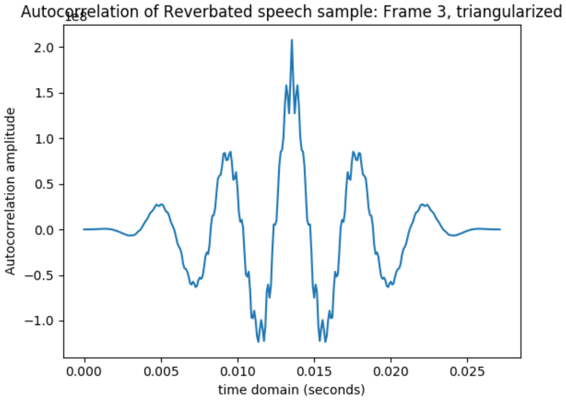

Figure 33: Auto-correlation of frame 3, reverb windowed

Figures 28 to 33 show the auto-correlations of the triangular windowed frames in the speech samples. Frequency information is discernible even on a qualitative level. There are clear sinusoidal oscillations particularly in figures 29 and 33. Wherever the period of the oscillations are too large for a fundamental frequency to be discernible, most likely the frame is too short and needs to be increased in size.

Theoretically, if the dry sample and reverberated samples begin at the same exact time, the triangular windowed autocorrelations for the dry sample and reverberated samples should not differ much in fundamental frequency information, assuming identical frame locations, even under heavy reverb[3]. This explains why the human ear can discern speech under heavy reverb conditions[3].

Comparing the most frames with the most concrete frequency information, frame 2, between the dry and reverberated speech samples, it seems that the frequencies are not proportionally equal with respect to the frame lengths. Frame 2 for dry speech (figure 29) is half as long as frame 2 for reverberated speech (figure 32), implying that the oscillations for the dry speech graph should visually be half as long in length as that for the reverberated speech graph, such that the fundamental frequencies are equal. This is clearly not true, which means that the two speech samples must begin at different times.

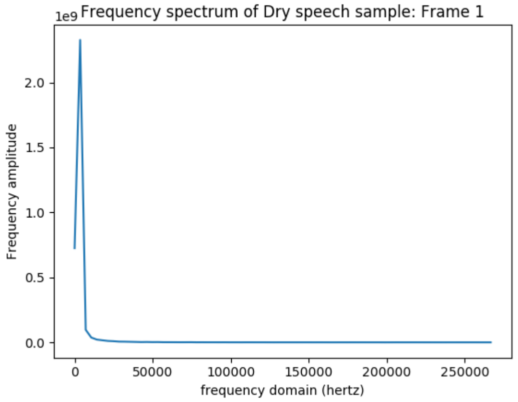

Figure 34: Frequency spectrum of frame 1, dry speech

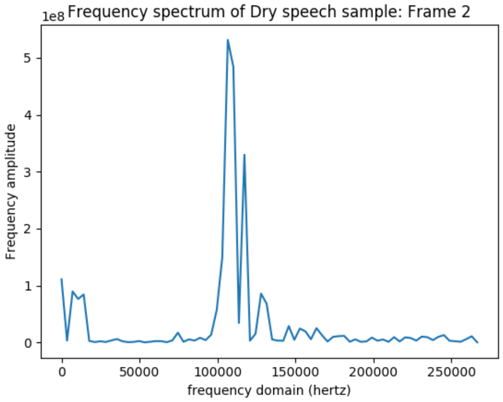

Figure 35: Frequency spectrum of frame 2, dry speech

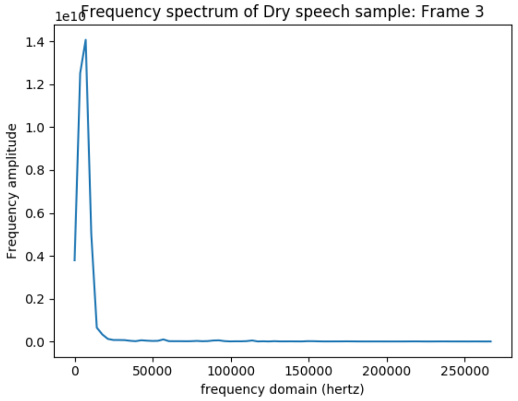

Figure 36: Frequency spectrum of frame 3, dry speech

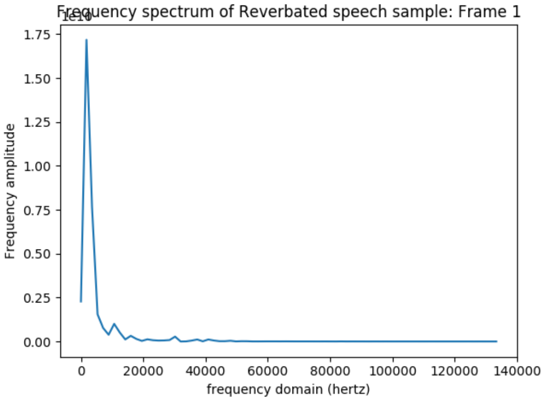

Figure 37: Frequency spectrum of frame 1, reverberated speech

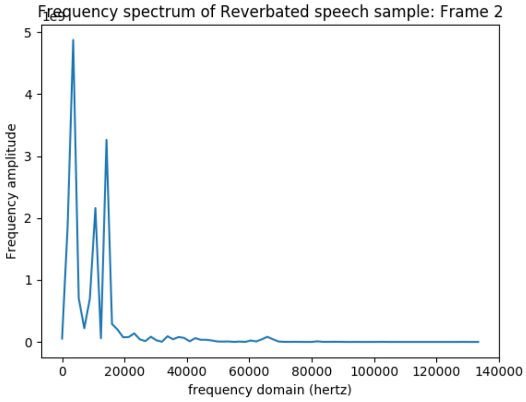

Figure 38: Frequency spectrum of frame 2, reverberated speech

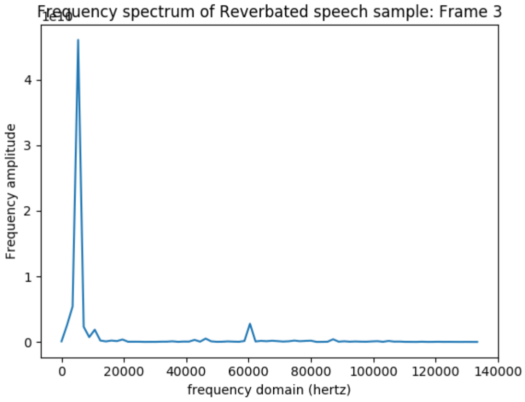

Figure 39: Frequency spectrum of frame 3, reverberated speech

Figures 34 through 39 show the frequency information calculated from taking the absolute squared values of the fourier transforms of the frames. While the negative values of the real fourier transforms are redundant, they should not distort the power spectra. The fourier transform is indeed useful in getting frequency information as well, since figure 35 and 39 have discernible spikes much beyond the lowest frequency, corresponding to the visually discernible frequencies in the auto-correlation sequences. However, in all the fourier transform plots there are other spikes as well corresponding to other frequencies. Thus, auto-correlation is a useful tool to be used with the fourier-transform to discern important frequency information.

5.5 Testing/Benchmarking

Effectively, using the auto-correlation sequence to test generated waveforms of different harmonics served to confirm that the auto-correlation function was being applied correctly. By confirming that fundamental frequency information was correctly obtained for waveforms where the frequencies could be controlled, it can be safely assumed that the same conditions apply to speech analysis.

While the convolution between different waveforms and unit pulse trains were not effectively test-cases for the auto-correlation sequence, they did help benchmark the effectiveness of auto-correlation usage to analyze frequency information for a signal of different harmonics, specifically ”signal A1” in figure 12.

5.6 Error Analysis and Conclusions

Wherever there was trouble using the fourier-transform, the auto-correlation sequence helped. Fundamental frequency information is complex in nature and relevant research is largely subjected to what the human ear can discern[3]. Thus, error cannot be directly quantified. However, at several points, particularly in the analysis of generated waveforms as was done in figure 14, and in the speech analysis, the auto-correlation provided very clear frequency information. Variance affects the data particularly in speech analysis as the time domain becomes long. While error was not an issue in the generated waveforms, in the case of human speech where frequency information varies rapidly, the analysis of frequencies both through fourier transforms and auto-correlation were heavily affected.

References

[1] Jones, E., T. Oliphant, P. Peterson, et al. (2017). SciPy: Open source scientific tools for Python.

[2] Mc Squared System Design Group, I. (2017). Reverberation time demonstration.

[3] Tohyama, M. (2015). *Waveform Analysis of Sound.* Springer.