Musical Instrument Identification Via Machine Learning Using Relative Harmonic Peak Amplitudes

Tyler Boulware
Emporia State University
Capstone Project Final Report

Submitted: May 11, 2018

1 Preliminary Research Proposal Statement

The goal of this research is to develop an algorithm which can accurately and consistently identify the type of musical instrument being played in a digital audio recording. The algorithm will focus primarily of harmonic and amplitude analysis of sound waves, and it will use traditional techniques from wave analysis including the discrete Fourier transform as well as mathematical models generated by analysis of existing audio recordings of known instruments.

The project will also include the development of a computer application which can perform the algorithm on a supplied audio file or live microphone recording and present its results to the user.

2 Postliminary Revised Research Description

The goal of this project was to classify musical instruments using a simple feed-forward neural network where the feature vector for the neural network contains the waveform's fundamental frequency and the relative frequency-domain amplitudes of the fundamental frequency and its harmonics. All analysis was performed using a multi-purpose application written in Python, making extensive use of the following libraries: *NumPy* for fast large-scale data manipulation, *SciPy* for signal processing, *Matplotlib* for data visualization, and *scikit-learn* for machine learning.

3 Overview of the algorithm

1. Given a waveform from a digital audio file (such as a WAV or mp3 file), a fast Fourier transform (FFT) produces a frequency-domain representation of the wave.

2. The harmonic product spectrum (HPS) algorithm is used to determine the fundamental frequency of the waveform from its frequency-domain representation.

3. Once the fundamental frequency is determined, the precise frequency-domain peak locations for the fundamental frequency and its harmonics are determined.

4. With the known peak locations, the relative amplitudes of the fundamental frequency and each of its harmonics are calculated.

5. These relative peak amplitudes along with the fundamental frequency are used as the feature vector to train the neural network for classification of the musical instrument.

6. The trained neural network is used to classify the test dataset to check the accuracy of its fit.

4 Justification of model choices

4.1 Pitch detection

There are several known algorithms which will effectively identify the fundamental frequency (pitch) of a waveform. Some common examples are the *harmonic product spectrum* (used in this project), *cepstrum analysis*, and *autocorrelation.*

HPS was chosen for this project largely due to its computational simplicity. For a neural network to train effectively, it needs a large number of training data samples (generally on the order of 1000, even for simple learning tasks), so the computational speed of *HPS* made it a clear choice over the slightly more accurate, but much more computationally expensive, method of *autocorrelation.*

Cepstrum analysis excels when the waveform has an abundance of strong harmonics and is good at detecting a suppressed fundamental frequency. There are not many cases of instruments with a suppressed fundamental frequency and often there are only a few strong harmonic peaks available, so *cepstrum analysis* was not the right choice for this project.

4.2 Neural network

The initial motivation for this project was to test for suspected simple patterns in the relative harmonic amplitudes of musical instruments. For example, perhaps certain instrument classes (brass, woodwind, string) would tend to have a particularly strong 1st harmonic or particularly weak even or odd harmonics.

This stems from the idea that generally the more harmonic content (particularly more and stronger higher harmonics) an audio waveform has, the brighter or harsher it is perceived [2]. This is easily demonstrable by comparing *pure sine*, *triangle*, *sawtooth*, and *square* waves. These waveforms have known Fourier coefficients and predictable harmonic amplitude ratios (See Appendix). These waveforms have distinct perceived sounds strictly due to their unique harmonic amplitude ratios. The hope was that a similar behavior could be found with real instruments.

Because the hypothesis was that the relationship would be relatively simple, a small and simple feed-forward neural network was used with a small input vector size. Because the neural network is relatively small, it is unable to model complex relationships between the input vector and its instrument classification. The input vector consists of the normalized amplitudes of the fundamental frequency and its first 8 harmonics. The input vector also included the normalized fundamental frequency in order to account for changes in the harmonic signature (the distinct amplitude ratios of harmonics) as the instrument changes pitch.

5 Details of algorithm

5.1 Fast Fourier transform

The real FFT, *scipy.fftpack.rfft()*, is used on the entire waveform (without zero-padding or windowing) with default options and without concern for implementation details and produces the discrete Fourier transform (DFT) for the waveform.

5.2 Harmonic product spectrum

The result of the FFT is successively downsampled and multiplied with itself:

1. A running HPS is initially assigned the data of the original DFT.

2. The original DFT is downsampled by a factor of $q = 2$ and the running HPS is point-wise multiplied with the downsampled spectrum.

3. **Step 2** is repeated for downsampling factors $q = 3, 4, \ldots q_{max}$.

4. Choosing an appropriate q_{max} is crucial. If the value is too low and there are a large number of upper harmonics, the results may not completely converge by the final downsample and multiply. If the value is too high, the fundamental frequency peak may get dominated by the large activity in the low frequency band caused by the large downsampling factor.

5. For this project, the highest overall success rates occured when $q_{max} = 5$. For the primary instruments studied, trumpet and violin, this value of q_{max} achieved successful pitch identification for 99.4% and 95.9% of the samples for each instrument respectively.

6. Because noise in the low frequency band (approximately $f < 50Hz$) can produce false peaks, all peaks in the *HPS* below $50Hz$ are ignored.

7. The peak with the highest amplitude in the resulting *HPS* is chosen as the fundamental frequency with one caveat. *HPS* will often detect the 1st harmonic as the fundamental frequency. By the method suggested in [5], to correct for this mistake, this implementation checks for a peak approximately half the frequency of the highest peak, and if its

amplitude is significant relative to the highest peak (20% suggested by [5]), then the lower peak is chosen as the fundamental frequency instead. Using this additional step increased the average success rate from 93% to 98%.

5.3 Peak finding in the DFT

1. The fundamental frequency peak is detected by searching in a frequency band centered at the fundamental frequency detected by the *HPS*: f_f. A bandwidth radius equal to half f_f was used. That is, the highest value in the DFT on the interval $[f_f - \frac{f_f}{2}, f_f + \frac{f_f}{2}]$ was chosen as the fundamental frequency amplitude.

2. Similarly, frequency bands centered on integer multiples of f_f are searched to find the harmonic peak amplitudes. In general, the k-th harmonic amplitude along the DFT is

$$max([k \cdot f_f - \frac{f_f}{2}, k \cdot f_f + \frac{f_f}{2}]).$$

3. Peak amplitudes for the fundamental frequency and its first 8 harmonics are detected. The particular harmonic count of 8 was chosen mostly arbitrarily. Since the algorithm relies primarily on the harmonic amplitudes, 8 was chosen as an excessively large number which should encompass the majority of the information for most spectrums of instruments.

5.4 Training the neural network

1. A feed-forward neural network with a 10-node input layer (fundamental frequency and the 9 normalized peak amplitudes), one 6-node hidden layer, and a 2-node output layer (for classification as either trumpet or violin).

2. All violin and trumpet audio samples (n=1,687 audio samples) were split randomly into either the training set or the test set with various split ratios, 90%/10%, 80%/20%, and 50%/50%, for the training and test sets respectively.

3. For each sample in the training set, the peak amplitudes of the fundamental frequency and its harmonics along with the fundamental frequency are normalized and added to the set of feature vectors on which the neural network was trained.

4. Various epoch counts from 10 to 1000 were used and the training data was shuffled each epoch and various regularization parameters from 0.0001 to 0.99 were used.

5. A confusion matrix is generated for the test set after each epoch and inspected to see how accurately the neural network is identifying instruments.

6 Results

With the entire data set (n=1,687 audio samples), largely independent of parameter tweaking, the neural network was able to achieve approximately an 85% classification success rate on the test data set. However, it still consistently failed to identify audio samples from outside the training or test set (approximately 50% success rate, which is the same as a random choice for a 2-way classification).

7 Discussion

The neural network was able to relatively accurately (85%) classify items in the test set but was unable to successfully classify items outside of the training or test set. I suspect one of the most likely explanations for this is that the data set was too closely correlated with itself, but not representative of a more general sample set. That is, I only drew samples from two large data sets. The samples within each data set were all relatively similar to one another and so they likely had a strong correlation with one another while perhaps not being representative of all audio samples.

It seems clear to me that the neural network was indeed training correctly, since it was able to very accurately overfit the training data with a consistent accuracy above 97%. Because we suspect the network is correctly training, the discrepancy between the test set accuracy and validation set accuracy seems best explained by the correlation being too high within the training and test set while not being representative of a more general pool of samples.

Another explanation for the poor validation set classification success is that it's possible that there is indeed no widely applicable real correlation between an instrument and its relative harmonic peak amplitudes. [6] shows that even humans do worse when attempting to identify an instrument without the attack or decay, and we are feeding our neural network even less information than that, stripping the transient time-dependent behavior entirely from the waveform. Therefore, it's possible that there is just no real strong correlation between an instrument type and its relative harmonic peak amplitudes.

8 Flaws

The clearest flaw was not using a more diverse pool of audio samples for training. Restricting to only 2 major sources may have introduced a very strong bias in our neural network results.

9 Possible Extensions

This project intentionally extracted and used only a small amount of data from the waveform in order to explore whether there exists a strong correlation between relative harmonic peak amplitudes. Naturally, there is a lot more data packed into a waveform aside from this. [1] shows high success with using information from the low-frequency band (below the fundamental frequency; 0-100Hz) as well as using the FFT only on particular regions of the waveform (such as the attack envelope).

There is also information in the time-dependent transient behavior of a waveform – perhaps a short-time Fourier transform could allow us to look for correlations between an instrument class and its waveform's time evolution.

A small, but perhaps important factor, is the inharmonicity of instruments. Because of the physical build of particular instrument classes, they will often exhibit characteristic levels of inharmonicity – even the method by which an instrument is played can affect the distribution of its harmonics (for example, whether a violin is bowed or plucked) [7].

10 Reflections

Throughout the project, I developed some knowledge and skill in a few fields that I knew very little of.

Prior to the project, I had never done any decently in-depth signal analysis. I have a new appreciation now for the power of the frequency-domain representation of a waveform as well as for how much more difficult it can be to analyze real digital waves (as opposed to idealized waves that are often studied in mathematics or physics courses). Developing algorithms for things like pitch detection is trivial when you're dealing with idealized waveforms, but with real waveforms, there are a lot of obstacles to overcome and corner cases to cover. Prior to researching effective pitch detection algorithms, I spent many weeks trying to develop my own algorithm for harmonic peak detection in the DFT, employing custom thresholding and statistical methods, but never achieved a decent level of success. After researching well-developed algorithms and applying them, I appreciate how effective, generalized, and elegant they are.

Prior to this project, neural networks were "magical" to me. After seeing what neural networks are capable of, I wanted to try to learn how they worked and try to implement one of my own to solve a problem. Unfortunately, I wasn't able to successfully solve my problem, but I was able to learn how neural networks work internally and learn by experience a few of the many pitfalls that show up when working with a neural network – and learned that they are essentially just a clever implementation of a really large self-adjusting function with anywhere from 10s to billions of constantly shifting parameters.

For most data analysis in the past, I've used Mathematica. It's a strong tool, but suffers from verbose syntax at times. Learning some of the details of NumPy and SciPy and using them consistently made me a lot more comfortable with doing large-scale data analysis and visualization quickly and effectively, a skill that's invaluable as I move forward and continue to explore new problems.

11 References

1. https://arxiv.org/pdf/1705.04971.pdf
2. https://courses.physics.illinois.edu/phys406/sp2017/Lecture_Notes/P406POM_

Lecture_Notes/P406POM_Lect6.pdf

3. https://ccrma.stanford.edu/ pdelac/research/MyPublishedPapers/icmc_2001-pitch_best.pdf
4. https://cnx.org/contents/i5AAkZCP@2/Pitch-Detection-Algorithms
5. http://musicweb.ucsd.edu/ trsmyth/analysis/Harmonic_Product_Spectrum.html
6. http://journals.sagepub.com/doi/pdf/10.2307/3345201
7. http://newt.phys.unsw.edu.au/jw/harmonics.html

12 Training and Test Set

The neural network was trained and tested with trumpet and violin samples from:

1. http://www.philharmonia.co.uk/explore/sound_samples/
2. https://freesound.org/people/MTG/packs/20224/
3. https://freesound.org/people/MTG/packs/20232/

13 Appendix (beginning next page)

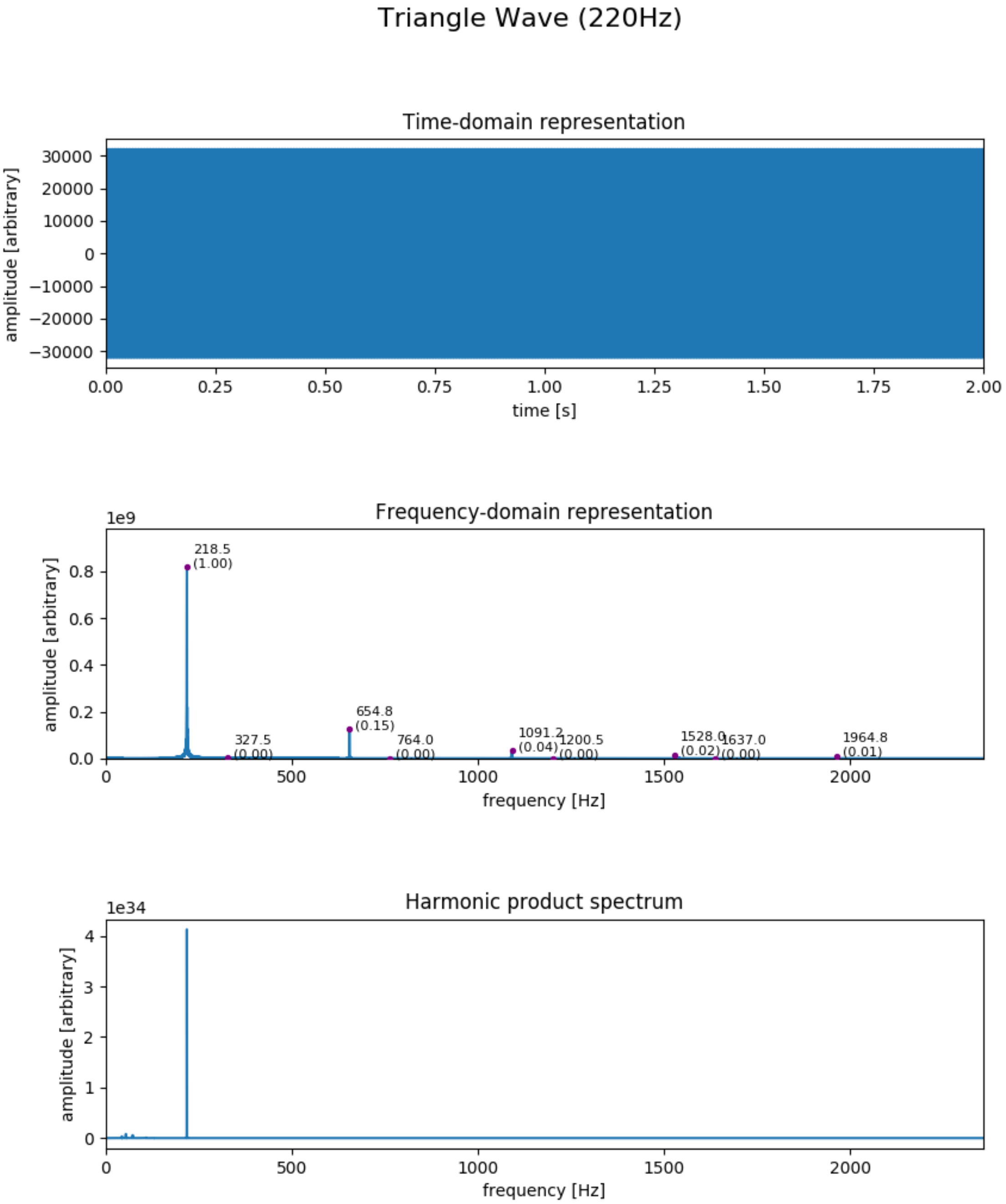

Figure 1: Results for a *220Hz* triangle wave. There is a distinct lack of odd harmonics (at *440Hz*, *880Hz*, etc.). Note that peak detection of these essentially non-existent odd harmonics fails for our harmonic peak detection algorithm. This isn't a major issue because instruments very rarely have such weak harmonic peaks in the lower harmonics.

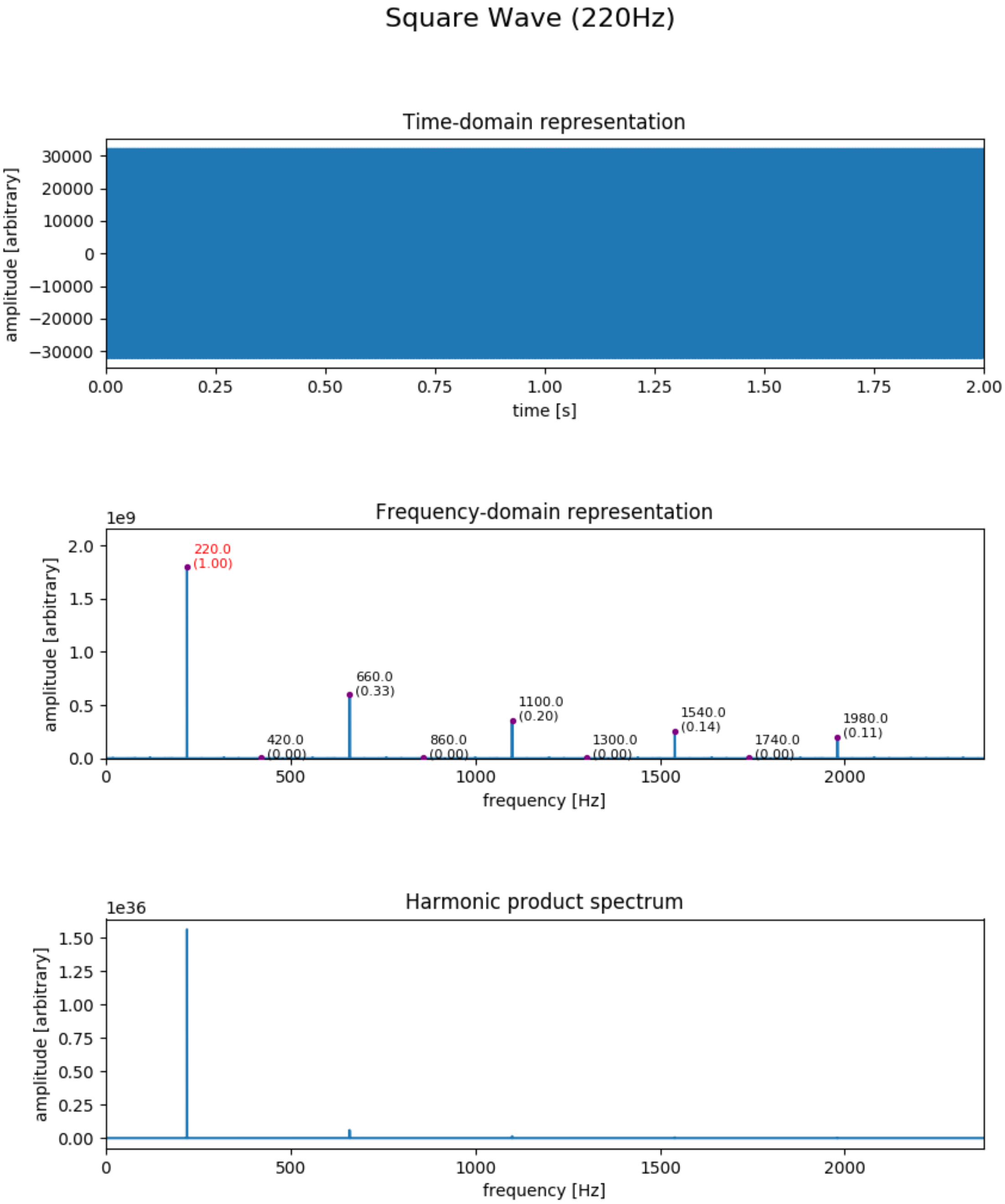

Figure 2: Results for a *220Hz* square wave. Like the triangle wave, the odd harmonics are very weak, but there is noticeably less dropoff between successive even harmonics when compared to the triangle wave.

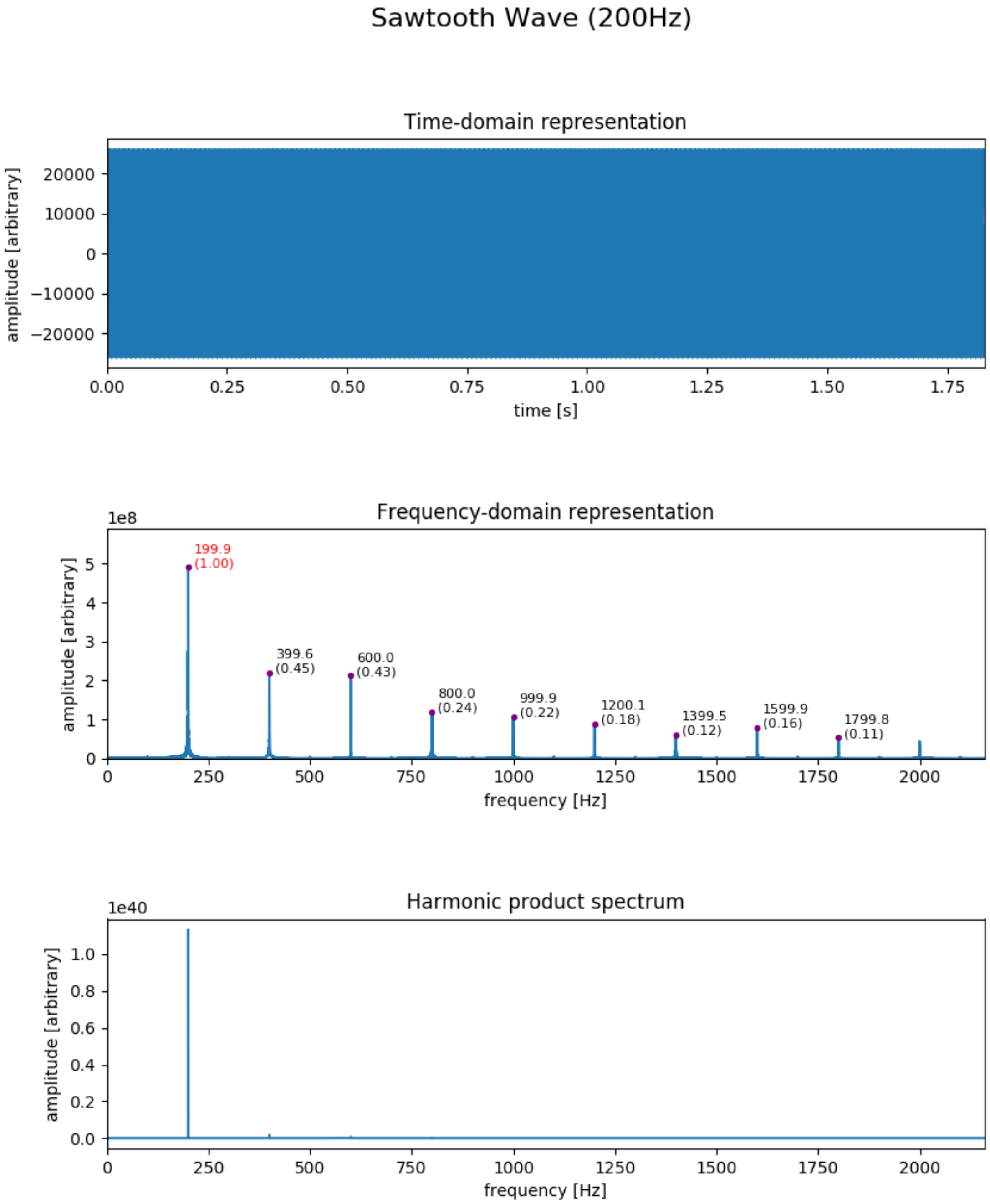

Figure 3: Results for a *200Hz* sawtooth wave. Unlike the triangle and square waves, the sawtooth has measurable amplitudes for all of its initial harmonics with a slow dropoff as the harmonic number increases.

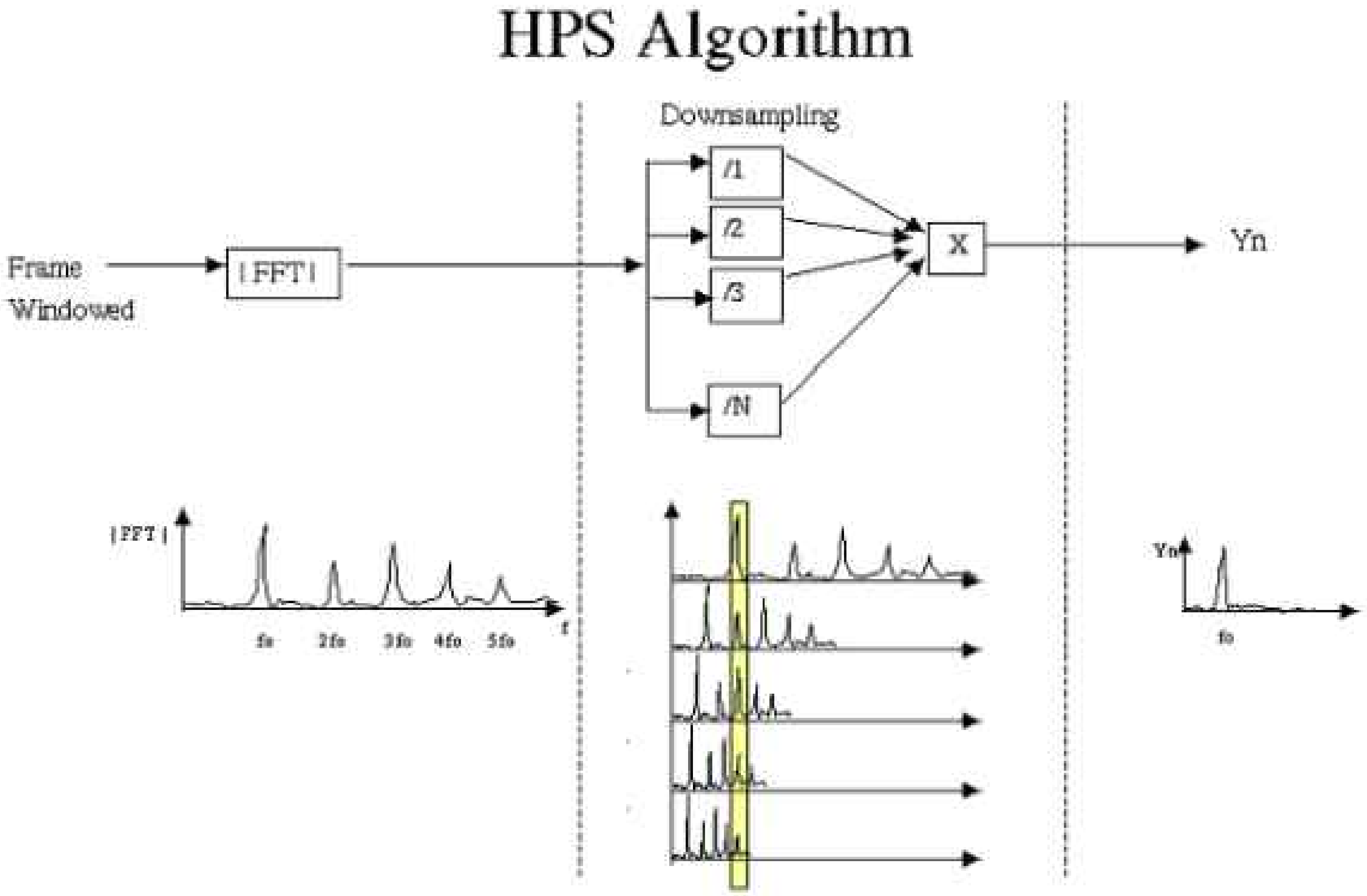

Figure 4: Visualization of how the harmonic product spectrum algorithm works from a free online resource *ECE 301 Projects Fall 2003* by Charlet Reedstrom [4]. Bottom center shows how successive downsamplings and multiplication will strengthen the peak at the fundamental frequency while weakening all other peaks.

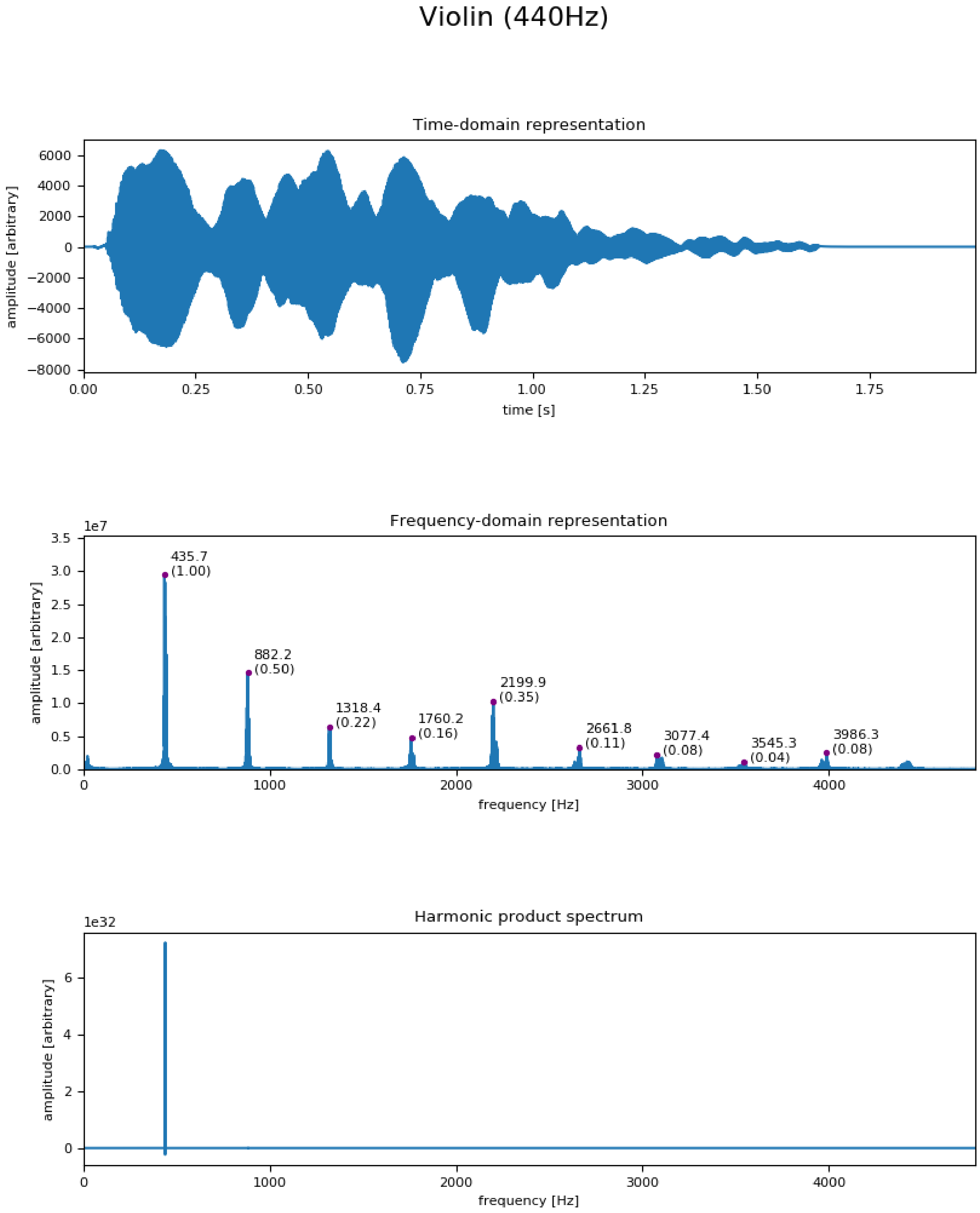

Figure 5: An example spectrum for a violin at 440Hz (A4).

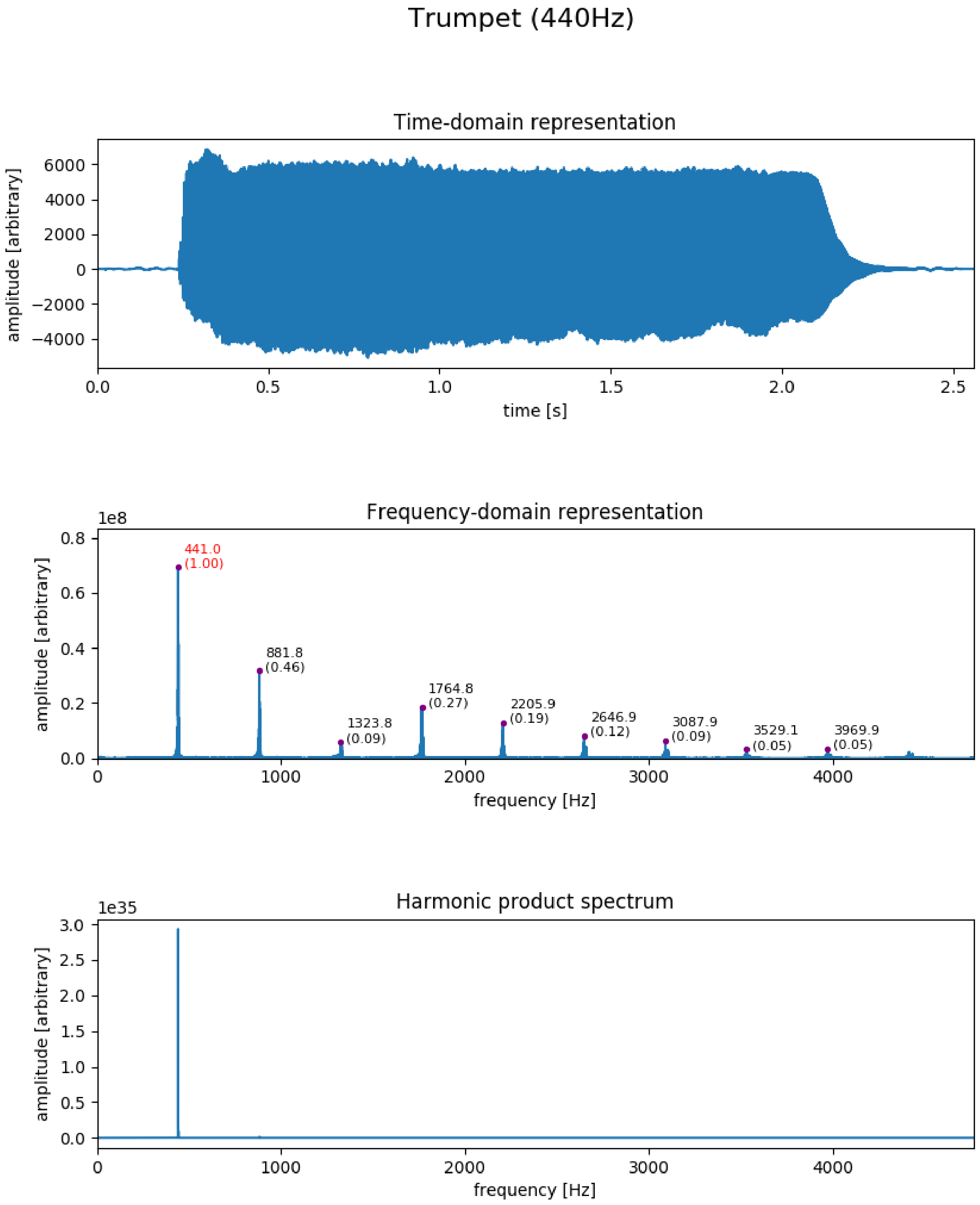

Figure 6: An example spectrum for a trumpet at 440Hz (A4).