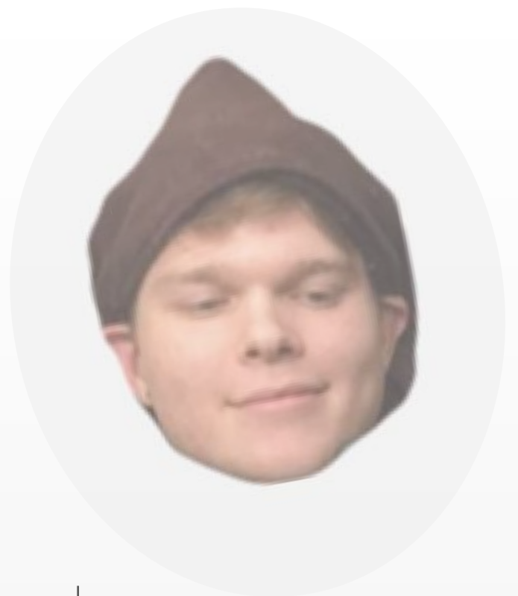
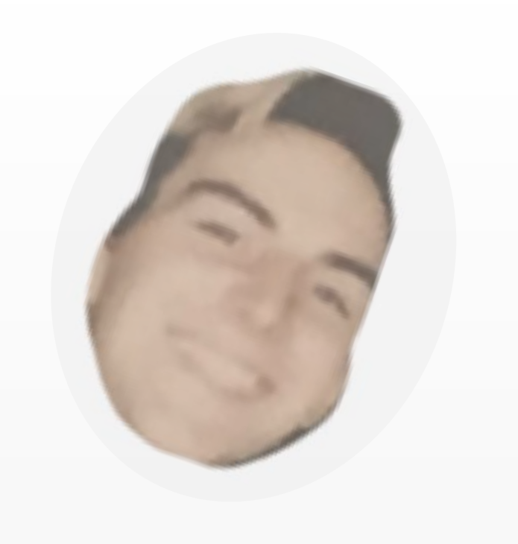
Autotune Implementation in MATLAB

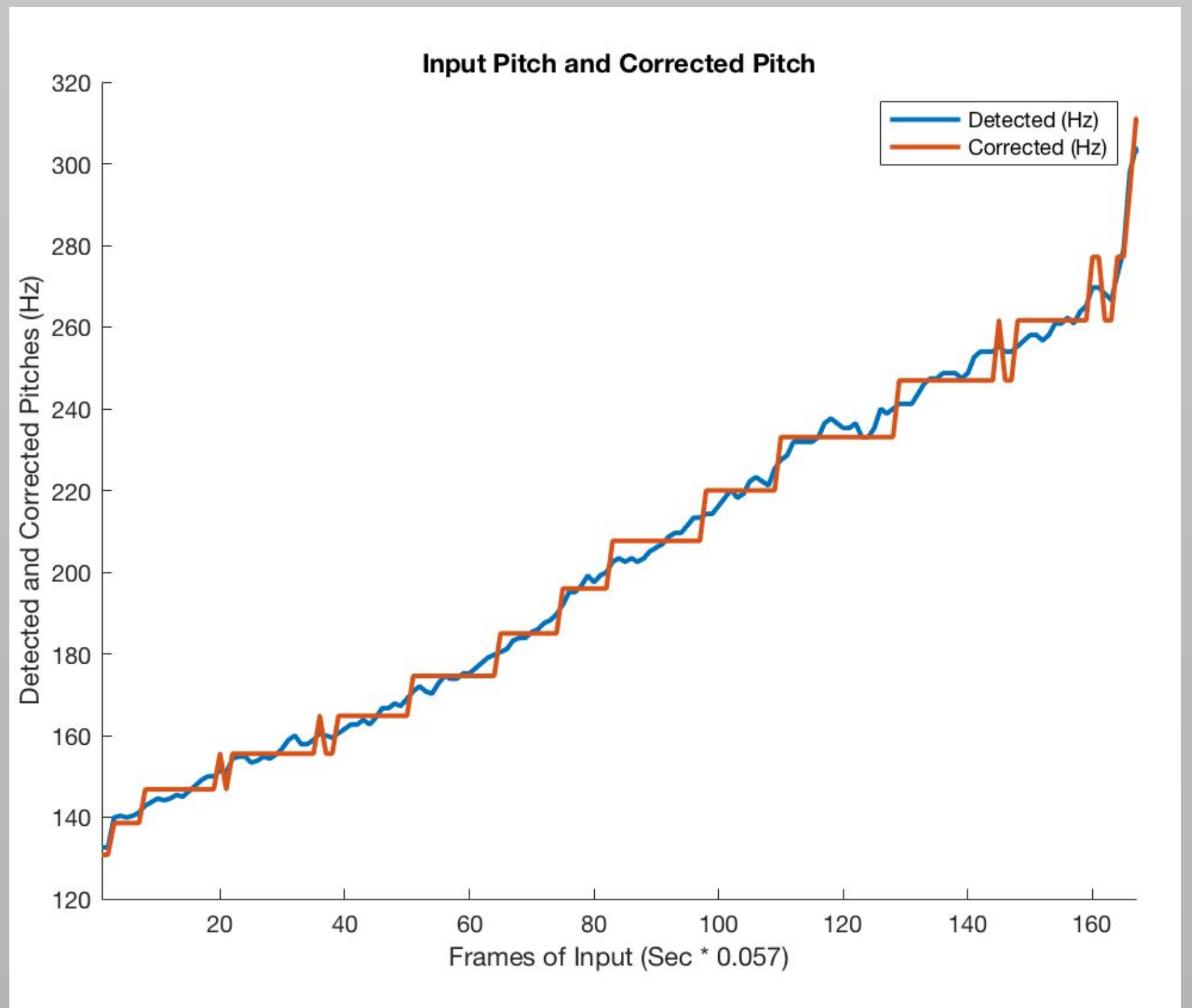
Evan Lo, Jacob Melchi, Max Pagnucco - *AME272 University of Rochester*

Introduction:

Autotune is an audio effect that automatically 'tunes' the input signal to a desired scale. This type of processing is often used on vocal tracks to correct anywhere that the singer may have been slightly out of tune. Many believe this type of effect is 'cheating,' and produces an artificial and inauthentic sound. However, when used in the proper context, autotune can be both a powerful creative tool as well as an invaluable effect to bring vocal recordings closer to perfection.

The program first needs to detect the fundamental frequency of the input. Using that value, it then computes the closest frequency that maps to a note in an equal temperament scale, tuned to 440 Hz. Finally, the pitch of the signal is shifted to the corrected frequency, without altering the playback time. Our program works both with a pre-recorded input file, and with live input/playback.



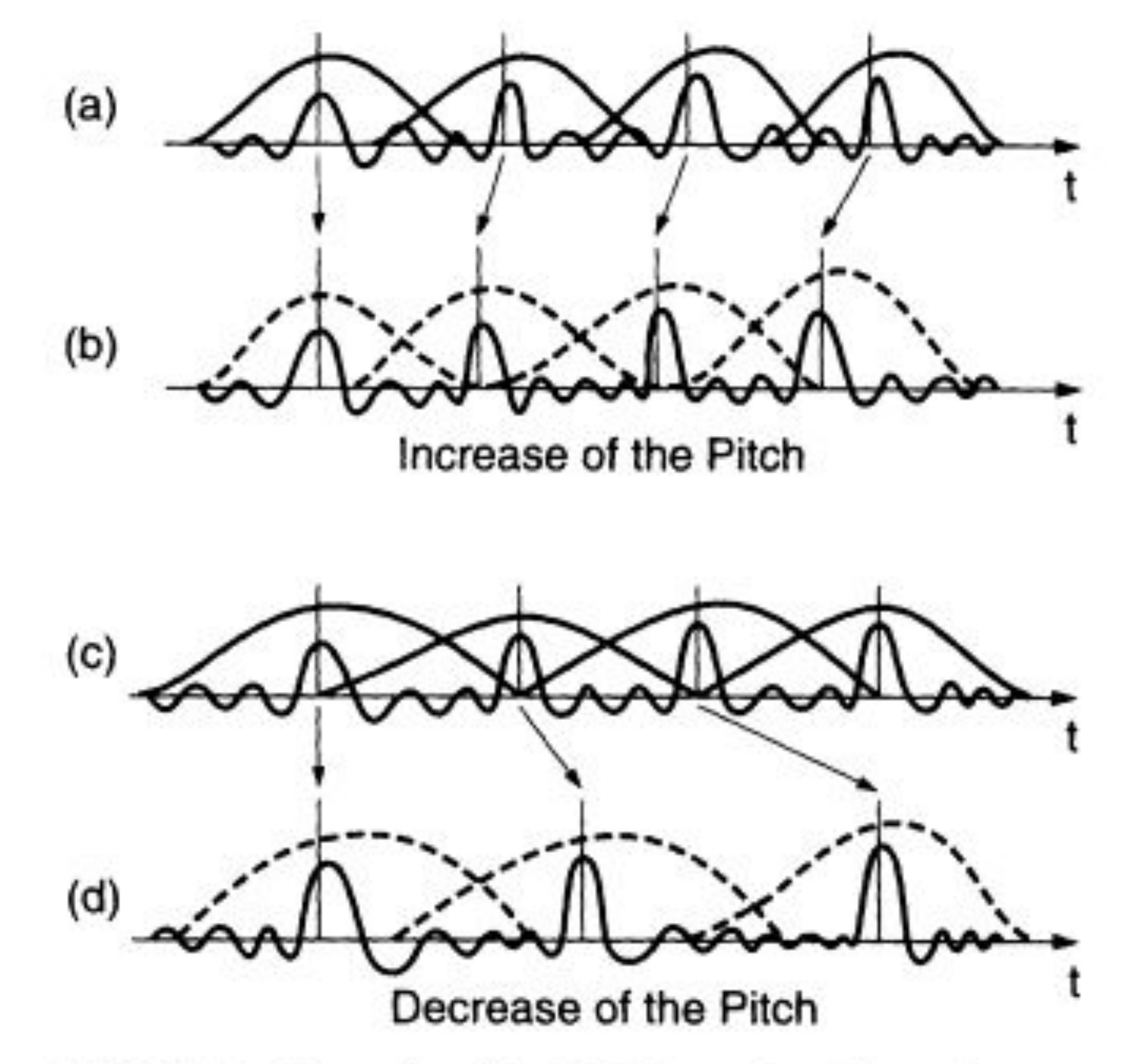


FIGURE 1 Example of the PSOLA method (from Sagisaka, 1990).

Pitch Detection:

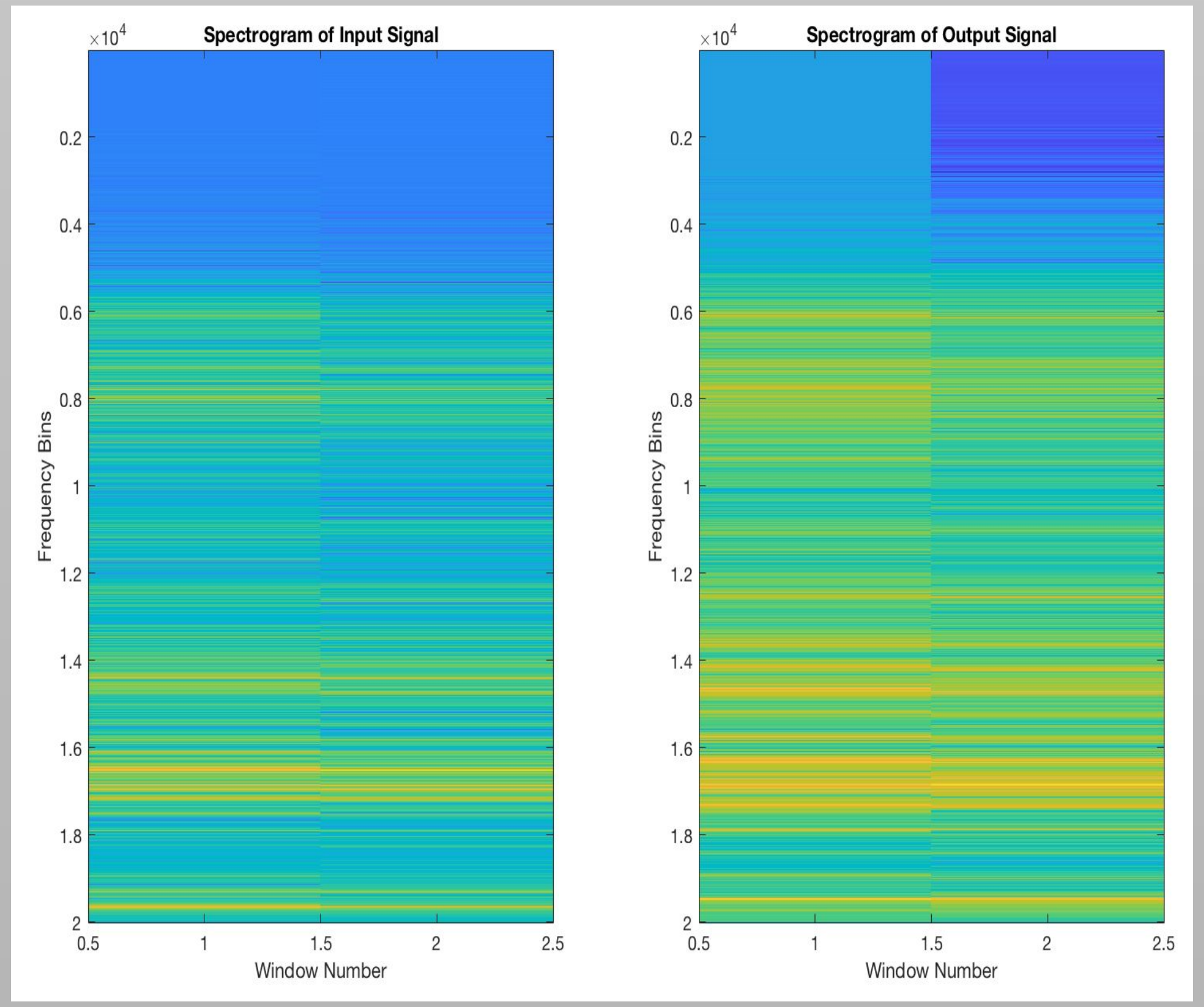
The first stage of autotune is determining the pitch of the given input signal, so that we can figure out how much it needs to be corrected. We tried several analysis methods to accurately return the fundamental frequency of the input such as spectral peak analysis and cross correlation, but both were inconsistent.

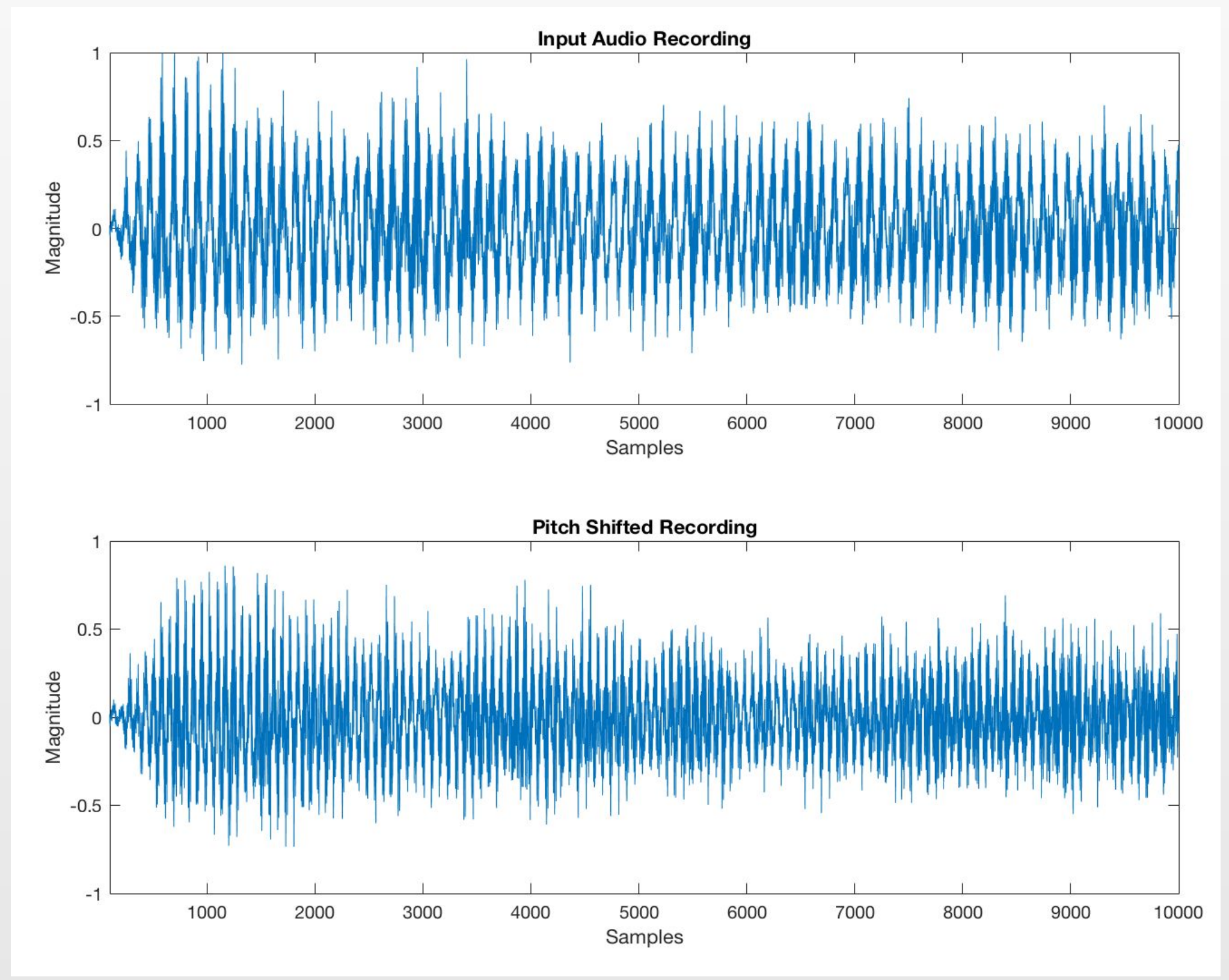
We ended up settling on using MATLAB's 'pitch' function, which was the most efficient and reliable means for returning the fundamental frequency. Our program then compares the detected fundamental frequency against all the given values in a frequency array, until it finds the one that is closest (the absolute value between the two is minimized). This process returns two pieces of information for the next stage: the fundamental frequency of the input and the target frequency to tune to.

Pitch Shifting:

Initially, we attempted to use the Phase Vocoder technique to perform the pitch shifting. Phase Vocoder is a technique that first changes a signal's pitch by resampling it, and then compensates for the unwanted time change by linearly interpolating between the first and last frames of the signal's spectrogram. All of the necessary processing does not happen fast enough to keep up with the sampling rate of the output DAC, and the resulting sound is quite egregious. Therefore, we decided to seek out an alternative method for pitch shifting.

The technique we decided to research and ultimately implement is called Pitch Synchronous Overlap Add (PSOLA). Once the fundamental frequency of an audio signal is found, markers are placed on the signal at every period of this frequency (preferably at each peak of this frequency).





These markers become midpoints for a very small-scale windowing process. The signal is split up into windows at these markers, using an appropriate window function. We decided on a Hanning window for this step because it begins and ends at zero, which lowers the chances of creating any unwanted audio artifacts.

Next, the chopped-up windows of the input signal are spread apart or pushed together based on the desired fundamental frequency. This effectively alters the fundamental frequency of the input signal. Frames are duplicated or dropped to make the length of the output match that of the input. The only thing left to do is to use the overlap-add technique to construct a single output signal Because these shifts in time are so miniscule, the output signal's character is only minimally altered, sometimes not at all.

Error & Conclusions:

During live implementation, our program has an issue with dropping frames, due to the autotuning algorithm not being able to keep up with the live processing. This can also be heard in the 'gaps' in our output. However, overall, we were able to create a fully functional autotune effect with a strong user interface and all the functionality we had planned. This project taught us about time domain signal processing techniques we had never been aware of, and opened our eyes to new audio signal manipulation techniques.

References:

[1] B.H. Suits, "Frequencies for equal-tempered scale, A4 = 440 Hz," Physics Department, MTU, Web, 1998.

[2] Daniel Shub, "Using xcorr in pitch detections," Help Forum, MathWorks, Web, 21 August, 2011.

[3] Zhiyao Duan, "Assignment: Homework 5," MATLAB template, University of Rochester, Web, 7 March, 2019.

[4] National Academy of Sciences. "Voice Communication Between Humans and Machines," Washington, DC: The National Academies Press, 1994.i

[5] Peimani, Michael A., "Pitch Correction for the Human Voice," Web Paper, University of California Santa Cruz, 10 June, 2010

[6] J. Yuan, Y. Chen, Z. Zhao, S. Liu, "EECS 451 Team Project: Conan's Bowtie," University of Michigan, Web, 7 March, 2019