AutoTune Toy (version 0.1), by Carl Arft
01/10/10
tfralrac@yahoo.com

Table of Contents

Introduction:

The AutoTune Toy was written to demonstrate pitch correction using Matlab. To be exact, I was watching Nova one night on PBS and there was a segment on Antares AutoTune software. I thought "I can do that in Matlab!" so this is my attempt at just that. This tool allows you to record short segments of your voice and interactively correct the pitch. Given that the tool uses memory to store the audio data, it is only practical to record short (20 second or less) audio fragments, but it is still a lot of fun to play around with!

The pitch detection algorithm that I came up with uses something like a modified autocorrelation routine in the time domain. The routine operates on small blocks of the recorded signal at a time. For example, the block length may be 1024 samples. In this case, the first 1024 samples would be extracted, and a copy of it would be made. The copy is then slid across the original block of 1024 points and the "likeness" of the block with the copy is determined. Since a particular pitch is a periodic signal, when the copy is slid the equivalent of one period to the right, the copy should match the original. This is the frequency, or pitch, of the first 1024-sample block. Then, this process is repeated for all of the remaining blocks of samples that comprise the entire signal.

In order to generate more frequency points, the blocks are overlapped. For example, the first block may be samples 1-1024. The next block could be samples 513-1536. The next block could be samples 1025-2048, and so forth. In this case the blocks consist of 1024 samples, but you step over 512 samples each time (half of 1024).

One complication comes about because the human voice has very strong harmonics. For example, you may be singing an "A" at 220 Hz. But, there are also frequency components at multiples of this frequency, i.e. 440, 660, 880 Hz. If the pitch detection routine detects that a certain block of samples simultaneously has frequency components at all of these frequencies, how do you determine what pitch is actually being sung? In this case, the routine finds all of the frequencies and then extracts those that are a multiple of one another, within some tolerance. Then the lowest of these is selected as the actual pitch, as long as its amplitude exceeds a minimum threshold.

Another complication occurs when stretching/compressing the blocks to the desired pitch and recombining them. To do this, each block is stretched/compressed by the appropriate ratio (determined by the old/new pitch) and interpolated. Then, each block is fit next to the old block and another modified autocorrelation is done to determine how the blocks best fit together. It is similar to fitting together two puzzle pieces.

This program was only tested using Matlab 7.5.0 and my Dell laptop. I don't know if it will work with other versions or computers (but I hope so!!). In order to use the tool, a working sound card and microphone is required. I have found that the recording level should be maxed out to generate a good recording.

Revision history:

1. Version 0.1: Initial revision

Getting started:

1. The following files are included in *AutoTuneToy.zip*:
    a. *AutoTuneToy.m*: The executable Matlab file. Run this file to launch the AutoTune Toy.
    b. *AutoTuneToy.fig*: The accompanying Matlab figure file for the GUI.
    c. *README.pdf*: These instructions.
    d. *mary.prj*: A sample file that can be loaded and manipulated using the tool.
2. To record your voice, simply press the **RECORD** button in the **Record Control** panel and start singing! Press **STOP** to stop the recording. It is only practical to record about 20 seconds or less.
3. After you press **STOP**, the tool will automatically attempt to detect the pitch of your voice. A plot of your pitch over time will be displayed. Figure 1 shows the first line of "Mary Had a Little Lamb." You can load this example by selecting **File/Load project…** and opening the *mary.prj* file.

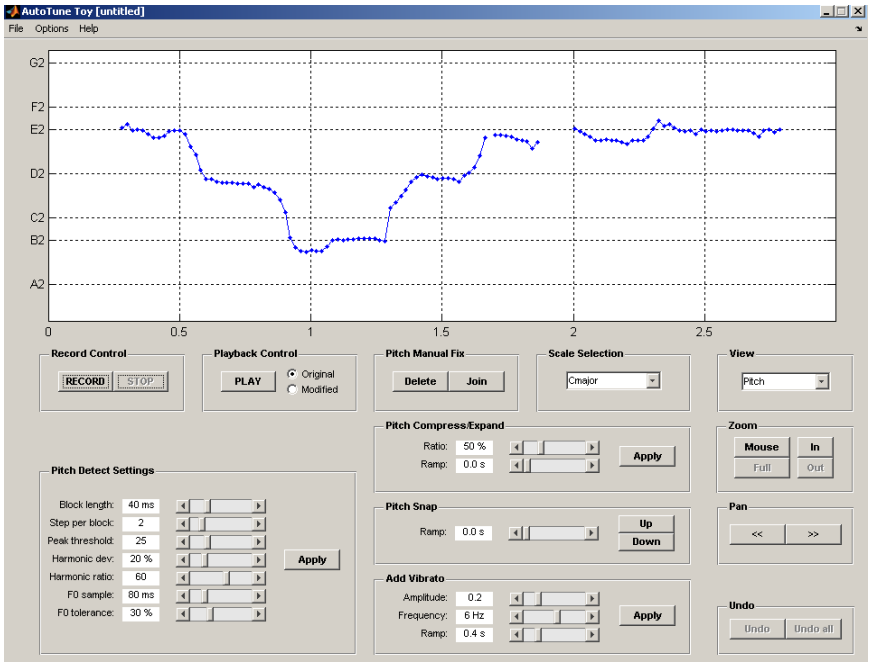

Figure 1. The detected pitch from the first line of "Mary Had a Little Lamb".

4. In this case, the pitch was calculated fairly well. There are four problems that may occur when determining the pitch of recorded voice:
    a. No pitch information, or very little/spotty pitch information: Try turning up the recording level on your microphone/sound card. The recording level must be pretty much maxed out to get good pitch detection.

b. Some missing pitch information: In the **Pitch Detect Settings** panel, decrease the peak yhreshold, then press **Apply** to repeat the pitch detection.

c. Some pitch information is an octave off: In the **Pitch Detect Settings** panel, decrease the harmonic deviation and/or increase the harmonic ratio, then press **Apply** to repeat the pitch detection.

5. Manually fixing the pitch points:

   a. Sometimes you will see a few erroneous pitch points that obviously do not belong. To manually delete these, select the point with the mouse and click **Delete** in the **Pitch Manual Fix** panel. To select multiple pitch points, hold down the *Ctrl* key while selecting the points with the mouse.

   b. Conversely, you will often find small "holes" in the detected pitch where there are missing pitch points. To manually fix this, select the surrounding points and click **Join** in the **Pitch Manual Fix** panel. In the "Mary Had a Little Lamb" example, select all of the pitch points with the mouse, and click **Join**. The result is shown in Figure 2.

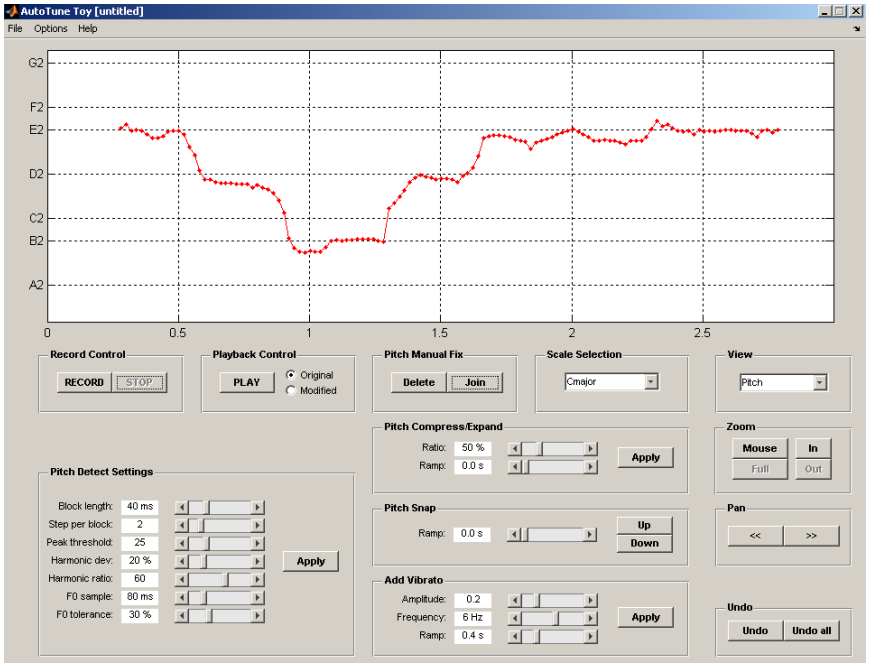

Figure 2. The pitch after all of the points are selected and the Join button is pressed to fill in the "holes".

6. The next step is to figure out what key you are in. In "Mary Had a Little Lamb" the third note is the fundamental of the scale, so this is in Bmajor. Select "Bmajor" in the **Scale Selection** panel.

7. Now it's time to start fixing the pitch! Zoom in to the first three notes ("Ma-ry-had") by clicking the **Mouse** button in the **Zoom** panel and selecting the first three notes (the section from 0.2 to 1.4 seconds). Reduce the pitch "waver" or variation of the first note by selecting just the pitch points in the "Ma" note (around E2) and clicking the **Apply**

button in the **Pitch Compress/Expand** panel. The more you click, the more the pitch variation of the note decreases. The result is shown in Figure 3.

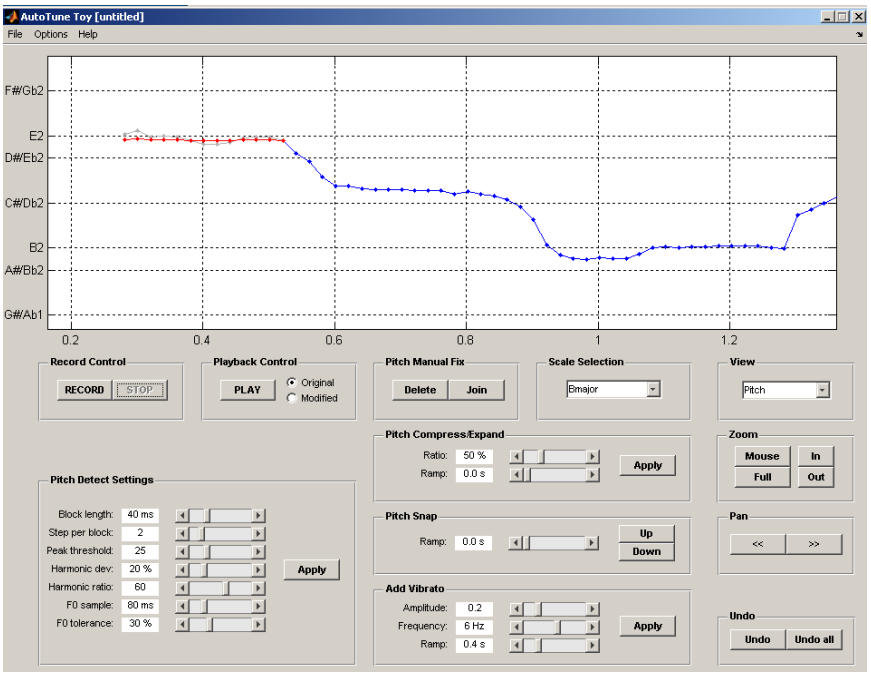

Figure 3. The first three notes ("Ma-ry-had"), shown after the variation of the first note ("Ma") has been reduced using the Pitch Compress feature.

8. Now the "waver" of the first note is gone, but the note is way too high (it should be a D#!). There are two ways to move the points down to the proper note:
    a. Hold down *Shift* and drag the highlighted pitch points down to D#. (Note: to drag individual points up/down, you don't need to select the points first. Just hold down *Shift* and drag points one by one).
    b. Click the **Down** button in the **Pitch Snap** panel. This will automatically snap all of the selected points down to the next note in the scale.
9. Repeat the previous two steps for the next two notes ("ry-had"). To make the voice seem natural, don't snap all of the points perfectly to the nearest note. Instead, leave the intermediate points as is and only fix the points that are near the correct note. See the results in Figure 4.

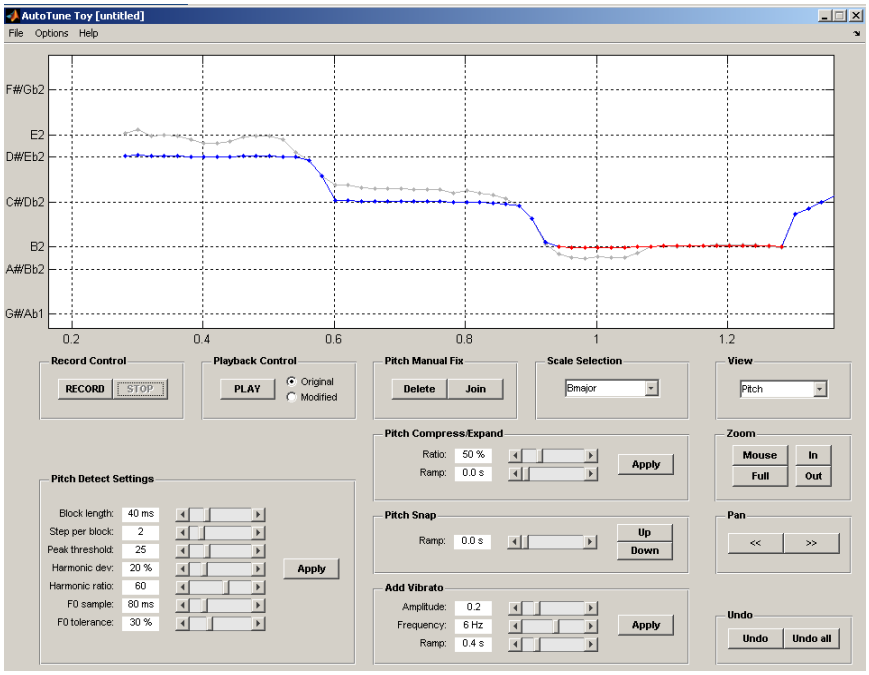

Figure 4. The first three notes ("Ma-ry-had") after each note has had the variation reduced and has been snapped to the correct note.

10. At any time you can listen to your handiwork by pressing the **PLAY** button in the **Playback Control** panel. To listen to the unmodified recording, select **Original** before pressing **PLAY**.
11. Move to the next portion of the recording by pressing the >> button in the Pan panel a few times. You can also zoom in/out using the buttons in the **Zoom** panel. Repeat the process of selecting all of the pitch points that comprise a note, compressing the pitch, then snapping to the proper note. After the entire passage is complete, the result will be as shown in Figure 5. Press **PLAY** (with the **Modified** button selected) to hear perfect singing!

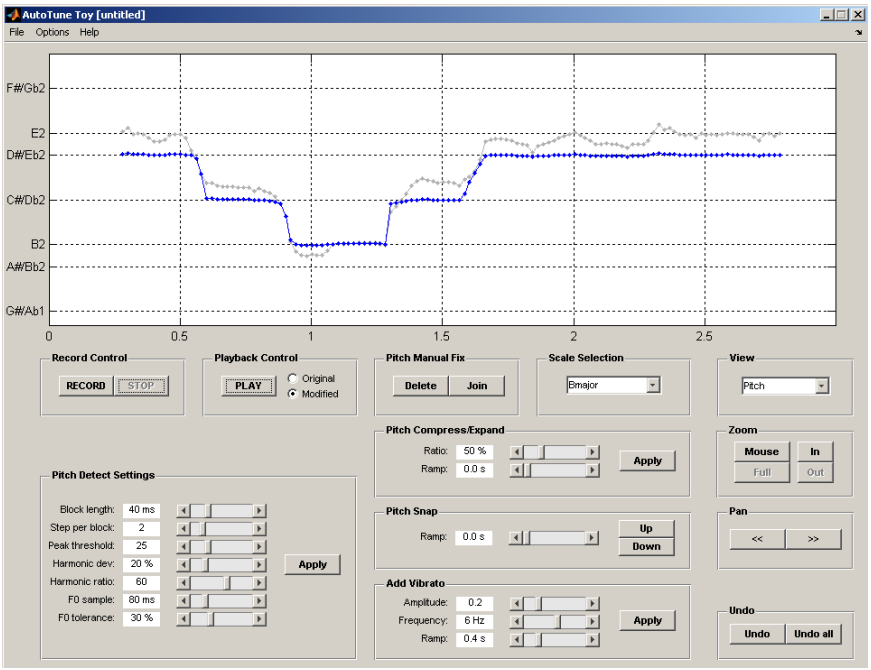

Figure 5. The entire phrase "Mary had a Litle Lamb" after each note has had the variation reduced and has been snapped to the nearest note.

Explanation of mouse controls:

- **Pitch point selection**: Left-click and drag to select pitch points. Pitch points turn red when selected. To select additional pitch points, hold down the *Ctrl* key while selecting. To deselect all points, click anywhere on the graph.
- **Pitch point correction**: After selecting pitch points, hold down Shift and drag the selected points up/down in pitch. If there are no selected pitch points, you can drag individual pitch points by holding down *Shift* and clicking on a single pitch point to drag it up/down.
- **Mouse zoom**: To zoom in on a specific area, click the **Mouse** button in the **Zoom** panel and then select the area to zoom.

Explanation of menu items:

File:

- **Save project**: Save the current original and modified waveforms and all settings to a **.prj* file.
- **Load project**: Reload a previously saved **.prj* file.

Options:

- **Record options**: Change the following settings:
  - **Sampling rate**: Default is 11025 samples/sec.
  - **Bits per sample**: Default is 16 bits/sample.
  - **Record start delay**: For some sound cards, there is a short (<100 ms) bit of noise generated when starting the recording. The start delay allows a small amount of the initial signal to be truncated automatically from the recording to eliminate this noise. The default is 100 ms.

Help:

- **About**: About the tool.
- **Help on using the AutoTune Toy**: This document.

Explanation of panel controls:

Record Control panel:

- **RECORD**: This button starts the recording. A working microphone and sound card must be installed in your computer.
- **STOP**: This button stops the recording. Immediately after pressing **STOP**, the tool will run the pitch detection algorithm.

Playback Control panel:

- **PLAY**: Press this button to play back the recorded sound.
- **Original**: Click this before pressing **PLAY** to hear the original, unmodified recording.
- **Modified**: Click this before pressing **PLAY** to hear the modified (pitch-corrected) recording.

Pitch Detect Settings panel:

- **Apply**: Press this button to run the pitch detection algorithm again, using the newest settings, which are as follows:
- **Block length**: The pitch detection algorithm operates on small blocks of samples, and determines the pitch of each block independently. The block length determines how many samples are in each block (N = *block length***sample rate*). The tool was tested mainly using the default setting of 40 ms; other settings may cause the pitch detection algorithm to fail.
- **Step per block**: This determines how many samples to skip when extracting the next block for processing. The default setting is 2. You can increase this number to obtain finer resolution of the pitch (the number of pitch points will be increased). As a simple example, in Figure 6 the block length is set to 2 sec, and there are two steps per block. Therefore a calculated pitch point will occur every 2/2 = 1 second. To increase the number of pitch points to every 0.5 second, increase the steps per block to 4.

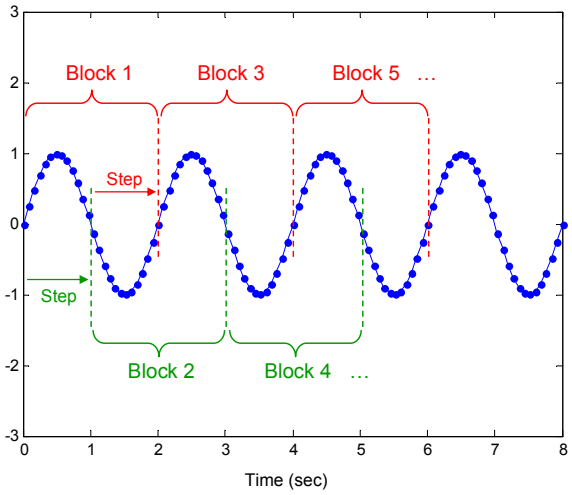

Figure 6. Example of a sampled waveform with block length and step-per-block indicated. In this case, the block length is 2 seconds. With a sampling rate of 12.5 samples/second, this means that each block consists of 25 samples. There are two steps per block, so that each block is 12.5 samples to the right of the previous block.

– **Peak threshold**: This determines the minimum correlation value required to determine the frequency of the signal. A value of 100 represents perfect correlation from one period to the next, while 0 represents no similarity. In Figure 7, two waveforms are shown. The one on the left is noise, and the one on the right is a human voice at a frequency of 161.2 Hz (somewhere around E2). The noisy signal has no repetitive pattern; therefore, the correlation plot below shows no peaks and very low peak amplitude. The human voice signal, however, shows strong peaks at regular intervals. The peak amplitude is about 80% (shown as 0.8 on this graph), and the first (lowest frequency) peak occurs at 6.2 ms. The frequency is then 1/0.0062 = 161.2 Hz. If you find that the pitch detection algorithm is finding erroneous points, increase the peak threshold value; conversely, if the algorithm is not finding enough pitch points, decrease the peak threshold value. The default value is 25.

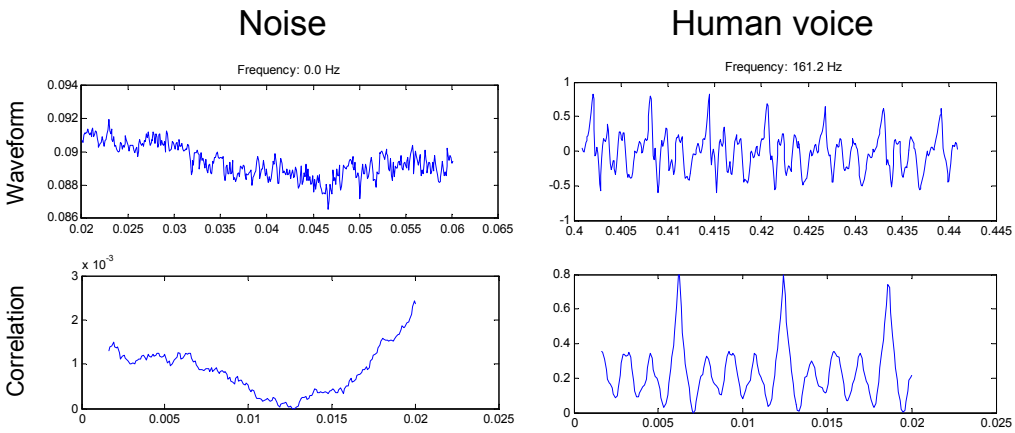

Figure 7. Examples of recorded noise (left) and human voice at a frequency of 161.2 Hz (right). For the noise signal, the correlation has no peaks and a very low amplitude. For the voice, the correlation shows peaks at multiples of the fundamental period/frequency. The lowest peak is at 6.2 ms, which corresponds to 161.2 Hz.

- **Harmonic dev**: As shown in Figure 7, the pitch detection algorithm finds peaks at multiples of the fundamental period. The peaks occur at harmonics (multiples) of the fundamental frequency. The algorithm finds all of these peaks and determines which ones are multiples of each other to within a certain tolerance. For example, the peaks might occur at 1.0 sec, 2.0 sec, 3.0 sec, and 4.5 sec. The first three are obviously harmonics of the fundamental at 1.0 sec, while the fourth is probably not. The harmonic deviation is the allowable tolerance (in percent) that a peak can be off from the predicted harmonic frequency and still be considered a peak. If the algorithm is not finding very many pitch points, try increasing the value; conversely, if the algorithm detects some pitch points an octave off from where they should be, try decreasing this value. The default value is 20%.
- **Harmonic ratio**: Referring to Figure 7, the harmonic ratio is the allowable ratio in peak height to be classified as a harmonic. Referring to the previous figure, the first peak height is about 0.8, the second also at about 0.8, and the third is about 0.75, or about 94% of the height of the first two. If the peak were below a certain ratio (determined by the harmonic ratio setting) it would be rejected as a possible harmonic. If the algorithm is not finding very many pitch points, try decreasing the value; conversely, if the algorithm detects some pitch points an octave off from where they should be, try increasing this value. The default value is 60.
- **F0 sample**: The algorithm stores the previous few pitch points for use in predicting what the pitch of the next block will be. The F0 sample setting determines how many previous samples are used for this prediction. The default value is 80 ms.
- **F0 tolerance**: This is the maximum deviation allowed from the frequency predicted by the previous few samples. The default value is 30%.

Pitch Manual Fix panel:

- **Delete**: Press this button to delete all selected pitch points. This is useful to remove any pitch points that are obviously erroneous, as shown in Figure 8.
- **Join**: Press this button to add in pitch points between selected points. This is useful if there is a "hole" in the detected pitch, as shown in Figure 8.

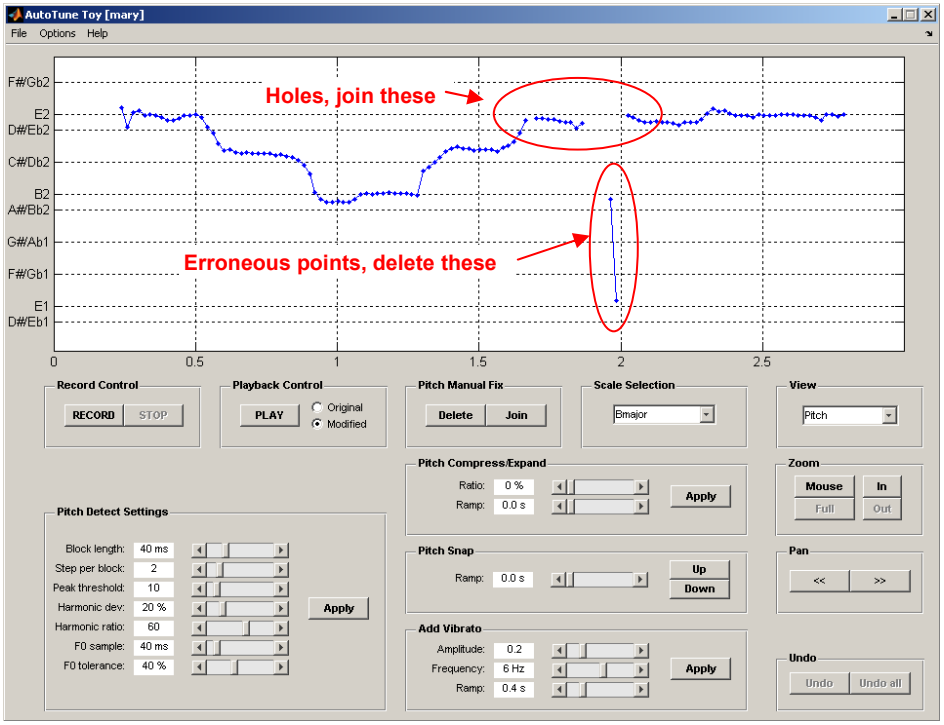

Figure 8. Illustration of erroneous pitch points that should be deleted, and holes that can be joined.

Scale Selection panel:

- Select the scale to display on the graph.

Pitch Compress/Expand panel:

- **Apply**: Press this button to compress/expand the selected pitch points. Compression decreases the "waver" or variation in the pitch, while expansion increases the variation in the pitch. The settings are as follows:
- **Ratio**: This determines how much the pitch variation is decreased with each click of the Apply button. A ratio of 50% means that the overall pitch variation of the selected points will decrease by 50% with each click. For example, Figure 9(a) shows the original pitch of a voice with significant "waver" and Figure 9(b) shows the pitch with a 50% compression factor applied (ramp = 0 sec).

– **Ramp**: This determines the time over which the pitch is ramped from no compression to the desired compression ration. This is used to "smooth" out the transition to the compressed pitch. For example, Figure 9(b) shows the pitch with a ramp of 0 sec, while Figure 9(c) shows the pitch with a ramp of 2.0 sec.

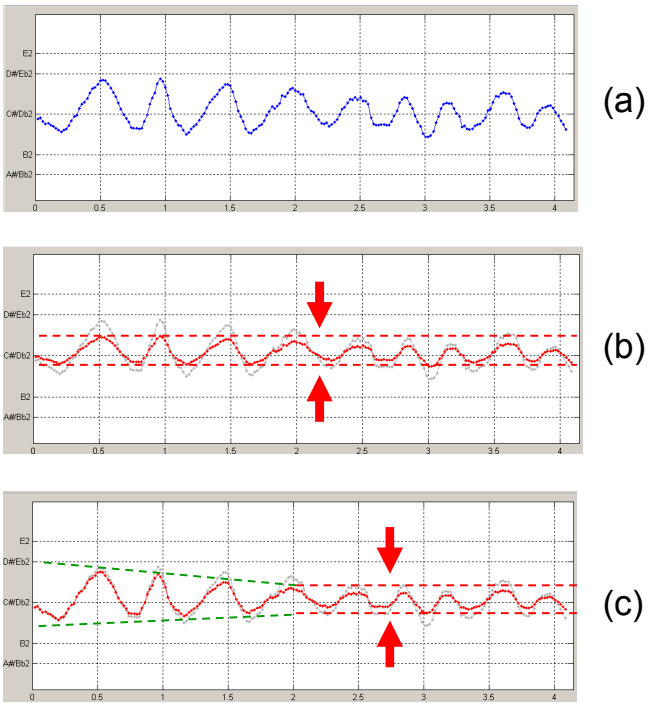

Figure 9. (a) Detected pitch of a voice with significant "waver" or deviation around the desired note. (b) The variation is compressed by 50%. (b) The variation is compressed by 50%, with a ramp of 2.0 sec.

Pitch Snap panel:

– **Up/Down**: Press these buttons to snap the selected pitch points up/down to the next nearest note, with the following setting:

– **Ramp**: This determines how quickly the pitch transitions from the old pitch to the new "snapped" pitch.

Add Vibrato panel:

– **Apply**: Press this button to add vibrato to the selected pitch points, using the following settings:

– **Amplitude**: The magnitude of the pitch variation of the vibrato. A value of 1.0 corresponds to a variation of one semitone (from F to F# for example) from the nominal frequency. As an example, Figure 10(b) shows vibrato with an amplitude of +/- 1.0 semitone.

- **Frequency**: The frequency of the pitch variation of the vibrato. Figure 10(b) shows a frequency of 5.0 Hz (5 variations per second).
- **Ramp**: How quickly the vibrato grows to the peak amplitude, as shown in the figure below. Figure 10(b) shows a ramp of 1.0 seconds.

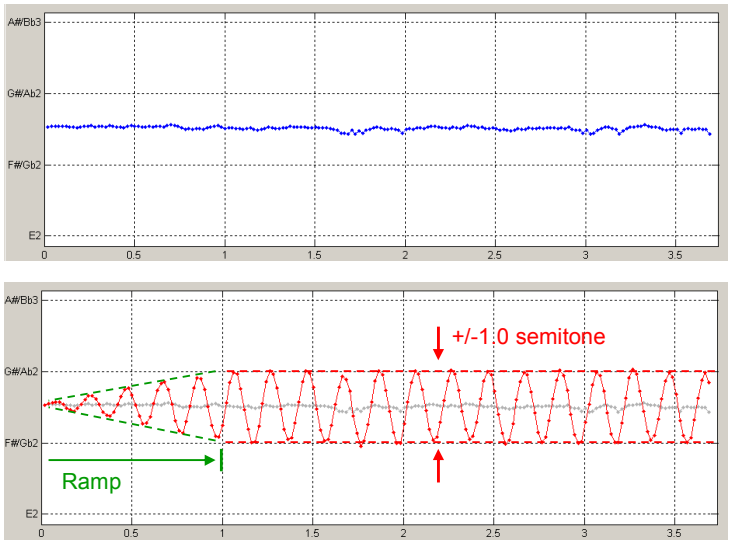

Figure 10. (a) Pitch of voice singing a constant note. (b) Pitch with vibrato added. The amplitude is +/-1.0 semitone, the frequency is 5.0 Hz (5 variations/second), and the ramp time is 1.0 sec.

View panel:

- Select Pitch or Waveform to view the detected pitch or the recorded waveform.

Zoom panel:

- **Mouse**: Click this button then select an area with the mouse to zoom in on the pitch.
- **Full**: Click this button to return to the full view.
- **In/Out**: Click these buttons to zoom in/out on the current view.

Pan panel:

- <</>>: Click these buttons to pan left or right from the current view.

Undo panel:

- **Undo**: Click this to undo the previous change.
- **Undo all**: Click this to undo ALL changes since the last time the pitch detection algorithm was run.