ProsodyPro — A Tool for Large-scale Systematic Prosody Analysis

Yi Xu

Department of Speech, Hearing and Phonetic Sciences, University College London, Chandler House, 2 Wakefield Street, London WC1N 1PF, UK

yi.xu@ucl.ac.uk

Abstract

This paper introduces ProsodyPro — A software tool for facilitating large-scale analysis of speech prosody, especially for experimental data. The program allows users to perform systematic analysis of large amounts of data and generates a rich set of output, including both continuous data like time-normalized F_0 contours and F_0 velocity profiles suitable for graphical analysis, and discrete measurements suitable for statistical analysis. It maximizes efficiency by automating tasks that do not require human judgment, and saving analysis output in formats that are ready for further graphical and statistical analysis.

Index Terms: ProsodyPro, time-normalization, annotation, F_0 velocity

1. Introduction

The need for stringent experimental control in prosody research is increasingly recognized [1, 4, 7, 8]. The analysis of experimental prosody data, however, is not always a straightforward matter. There is often a dilemma between systematic comparison of discrete measurements [8] and detailed analysis of continuous prosody [3, 5]. In most cases, details are sacrificed in favor of straightforward comparisons. In the intonation literature, for example, typically continuous F0 contours of a few utterances are shown as illustrations, and then subsequent analyses are done on only a limited set of measurements, with little or no further examination of continuous contours. This leaves many details in continuous prosody unknown. A likely reason for such compromise is the lack of tools that can simultaneously facilitate both types of analysis.

2. ProsodyPro — An integrated tool

This paper introduces ProsodyPro, a Praat script that allows users to combine systematic comparison with detailed analysis of continuous prosody. The key design is to use time-normalization to facilitate direct comparison of continuous F_0 contours, while at the same time generate multiple measurements from non-time-normalized data suitable for statistical analysis.

2.1. Time-normalization

Time-normalization is the key method of ProsodyPro to facilitate close scrutiny of continuous F_0 contours over multiple tokens. When an experiment has recorded many sentences, especially when each unique sentence is repeated several times by each speaker and by multiple speakers, it becomes difficult to analyze and virtually impossible to report all the data. For example, Figure 1a shows F_0 contours of the Mandarin tone sequence HLFHH produced by four male speakers. These F_0 tracks are generated by the "Export visible pitch" function in Praat [2]. Visually we can see some similarities across the speakers despite the differences. But how can we capture the similarities? One way is to take a measurement in the middle of all syllables and average them across the repetitions as well as the speakers, as shown in Figure 1b, from which we can see that the greatest differences occur on syllable 2, which carries four alternative tones. However, while statistics may show a significant difference across the four tones with this kind of measurement, many finer differences are lost. In Figures 1c and 1d, as two or three measurements are taken from each syllable, more details start to emerge. But it is not until Figure 1e, where eight measurements are taken from each syllable, does the continuous nature of the F_0 contours clearly emerge.

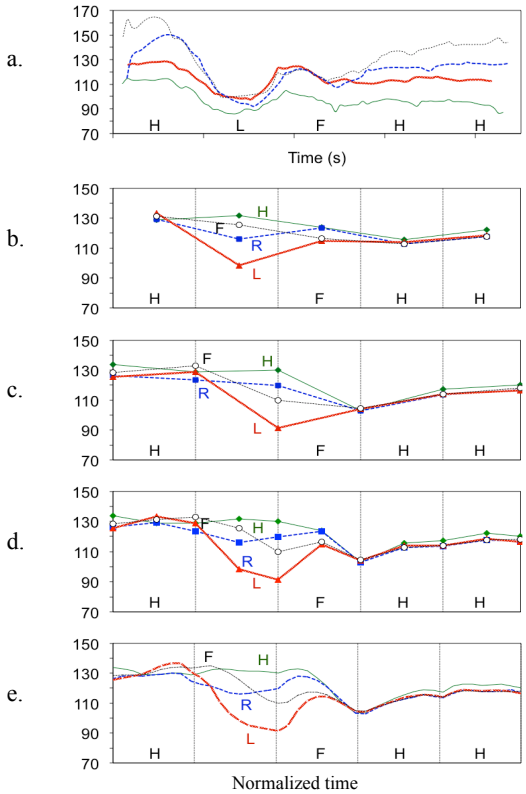

Figure 1: a. Raw F_0 tracks of HLFHH by 4 male speakers, generated by Praat. b-e. *Mean time-normalized F_0 of a five-syllable Mandarin sentence sampled at 1, 2, 3 and 8 samples per syllable.*

Thus time-normalization allows averaging across repetitions as well as speakers, a process that also smoothes out random variations unintended by the speaker, as well as individual differences, leaving only consistent variations due to tone and contextual tonal variations. From Figure 1 we can also see that time-normalization is only a further extension of the coarser sampling as in Figures 1b-1d, which are in fact also time-normalized. But the finer sampling allows us to see much more details, leaving little to guesswork. This can be seen in the two examples in Figure 2, where similarities in focus realization can be seen between Mandarin and English. Note that the English F_0 contours in Figure 2b are displayed in real rather than normalized time. This is achieved by taking time values together with F_0 values, and averaging both across repetitions and speakers. The averaged real time can then be used as the time axis for plots like Figure 2b.

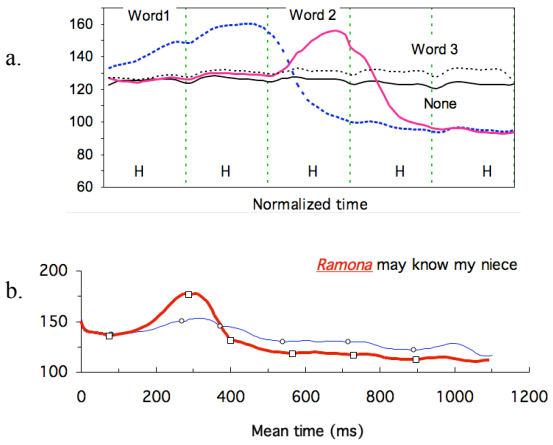

Figure 2: *Mean F_0 contours of Mandarin (a) and English (b) sentences in different focus conditions. The plots are adapted from studies [6, 11] using precursor versions of ProsodyPro.*

Time-normalization, however, requires users to define the temporal domain of normalization. In ProsodyPro this is done by inserting interval boundaries in the TextGrid of an utterance. Technically ProsodyPro allows the use of any units as the temporal domain of normalization, e.g., syllable, word, or even phrase. But it is important that there are good reasons to believe that the F_0 contours of the unit are consistently produced. Our recommendation is to use the syllable (or rhyme) whenever possible. This is based on evidence that speakers produce syllable-sized contours consistently [9, 10].

2.2. Other continuous prosody output

In addition to time-normalized F_0 contours, ProsodyPro also generates a number of non-time-normalized continuous prosody outputs. The following is a list of all the continuous prosody files, with a variety of time scales:

X.rawf0 — Real-time F_0 (Hz) converted directly from vocal periods ($F_0 = 1/T$, where T is vocal period in second) marked in X.pulse

X.f0 — Real-time F_0 (Hz) smoothed with a trimming algorithm (Xu, 1999)

X.smoothf0 — Real-time F_0 (Hz), smoothed from X.f0 with a triangular filter

X.normtimef0 — Time-normalized F_0 (Hz), with default sampling rate of 10 points/interval

X.actutimenormf0 — Time-normalized F_0 (Hz) with real-time x-axis values

X.samplef0 — F_0 (Hz) sampled at at fixed time intervals specified by "f0 sample rate"

X.semitonef0 — Semitone version of X.samplef0

X.f0velocity — Continuous F_0 velocity (instantaneous rates of F_0 change) in semitone/s sampled at time intervals specified by "f0 sample rate"

X.normtime_f0velocity — Time-normalized continuous F_0 velocity

All these files are automatically generated after an utterance is annotated. These outputs allow users to examine continuous prosody of each utterance. However, as explained above, only the time-normalized ones can be averaged into mean contours across repetitions and speakers.

2.3. Discrete measurements

Time-normalization, despite its many advantages as mentioned above, is meant only for graphical comparisons. For statistical analysis, it is impractical to compare every time-normalized point. Nonetheless, time-normalization allows us to see the locations and manners of maximum differences when mean continuous contours from different experimental conditions are plotted in overlaid graphs, like those in Figure 1e and Figure 2. This can help identify optimal measurements that best reflect the key differences between experimental conditions, and, just as importantly, avoid pitfalls. Figure 2a, for example, shows that the F_0 at the onset of the first post-focus syllable "-na" is higher than in the same syllable in the neutral focus contour. This means that maxf0 of that syllable is not the best measurement for showing the lowered F_0 characteristic of all the other post-focus syllables.

The following measurements are automatically generated by ProsodyPro for each non-blank interval in the TextGrid and saved in the file X.means, where X stands for the name of the sound file being analyzed:

1. maxf0 — maximum F_0 in Hz
2. minf0 — minimum F_0 in Hz
3. excursion_size — difference between maximum F_0 and minimum F_0 in semitones
4. meanf0 — average F_0 in Hz
5. max_velocity — maximum F_0 velocity in semitones/s
6. finalf0 — F_0 near the interval offset in Hz
7. final_velocity — F_0 velocity near the interval offset in semitones/s
8. duration — interval duration in ms
9. mean intensity — mean intensity in dB

2.4. Ensemble files

To facilitate both graphical and numerical analysis, ProsodyPro has a function to pool the outputs of all individual sounds in a folder together into a large set of ensemble files. The following is a list of ensemble files generated by the current version of ProsodyPro (some are optional):

1) normf0.txt
2) normtime_semitonef0.txt
3) normtime_f0velocity.txt
4) normtimeIntensity.txt

5) normactutime.txt
6) samplef0.txt
7) f0velocity.txt
8) maxf0.txt
9) minf0.txt
10) excursionsize.txt
11) meanf0.txt
12) duration.txt
13) maxvelocity.txt
14) finalvelocity.txt
15) finalf0.txt
16) meanintensity.txt
17) mean_normf0.txt
18) mean_normtime_semitonef0.txt
19) mean_normtime_f0velocity.txt
20) mean_normtimeIntensity.txt
21) mean_normactutime.txt
22) mean_maxf0.txt
23) mean_minf0.txt
24) mean_excursionsize.txt
25) mean_meanf0.txt
26) mean_duration.txt
27) mean_maxvelocity.txt
28) mean_finalvelocity.txt
29) mean_finalf0.txt
30) mean_meanintensity.txt
31) mean_normf0_cross_speaker.txt

These are all text files that can be opened by spreadsheet, graphing or statistical programs. The first seven files contain continuous prosody of all the annotated sounds in the folder in normalized, real or sampled time. Files 8-16 each contains values of a specific measurements from all the sounds. Files 17-30 contain values averaged across the repetitions of unique sentences. And file 31 contains time-normalized F_0 values averaged across multiple speakers. Values in files 17-21 are ready for graphing mean F_0, F_0 velocity or intensity contours of individual speakers. Values in file 31, which contains across-speaker means, can generate plots like those in Figures 1e and 2. The discrete mean measurements in files 22-30 are ready for statistical analysis such as anova and t-test, for which each subject needs to contribute only a single mean value for each of the factors being tested.

3. Workflow and time-saving features

To analyze large amounts of experimental data, much labor is needed simply to process the data files, taking and recording the measurements, and readying the output data for graphical and statistical analysis. ProsodyPro has incorporated many labor-saving features specifically designed for experimental research.

- First, ProsodyPro is written as a Praat script, so that it is executable on all major computer platforms.
- Second, the entire program consists of a single script file to be run in the same folder as the sound files being analysized, thus reducing installation effort to a minimum.
- Third, ProsodyPro automates tasks that require little human judgment, including, in particular, finding and opening sound files, and saving analysis results.
- Finally, ProsodyPro arranges output data in formats that are nearly ready to be processed further by spreadsheet, graphing and statistical programs.

The following is a brief sketch of the workflow of PrododyPro, with graphic illustrations shown in Figure 3.

1. Put ProsodyPro.praat in the folder containing the sound files to be analyzed.
2. Launch ProsodyPro either from "Open Praat Script..." command from Praat menu, or by double-clicking the ProsodyPro icon if it is recognizable as a Praat script by the operating system.
3. When the script window opens in Praat, select "Run" from the Run menu (or use the shortcut command-r or control-r).
4. In the startup window (Figure 3a), check or uncheck the boxes when necessary, and set appropriate values in the text fields or simply use the default ones. Select the task "Interactive labeling" by ticking the first radio button.
5. Click Ok and three windows will appear. The PointProcess window (Figure 3b) displays waveform together with vocal cycle marks (vertical lines) generated by Praat. Here one can manually add missing marks (e.g., in the last vocalic sound in the figure) and delete redundant ones. This needs to be done only for the named intervals, as explained next.
6. The TextGrid window (Figure 3c) displays waveform and spectrogram of the current sound. (Note that the pitch track in the spectrogram panel is not used by ProsodyPro.)
7. At the bottom of this window are the annotation tiers, where users can insert interval boundaries (Tier 1) and comments (Tier 2). Any blank intervals (or those labeled "sil") in Tier 1 are ignored, which allows easy exclusion of temporal regions from analysis. The interval labels can be as simple as a, b, c... or 1, 2, 3..., since ProsodyPro ignores label content.
8. The Pause window (Figure 3d) controls the workflow. To bring up the next sound to be analyzed, change the number (or leave it as is) in the current_file box and press "Continue". The number indicates the order in the String object "list" in the Object window (and in the file "FileList.txt" automatically saved to the current folder). The next sound will be 1 + current_file (So, entering 0 opens sound 1).
9. To end the current analysis session, press "Finish" in the Pause window, and the order number of the last sound analyzed is shown in the Praat Info window. That number can be used as a starting point in the next analysis session.
10. After processing individual files, ensemble files can be generated by running ProsodyPro again with the "Get ensemble files" radio button checked.
11. The analysis parameters in the startup window can be modified after annotating all the sound files. To do so, start a new session with the radio button "Process all sounds without pause" checked. ProsodyPro will exhaustively run through all files nonstop.
12. To generate the means files (files 17-30 listed in Section 2.4), set the value of "Nrepetitions" in the startup window to the number of repetitions in the dataset when running ProsodyPro with the "Get ensemble files" button checked. Note that the number of labeled intervals must be identical across repetitions.

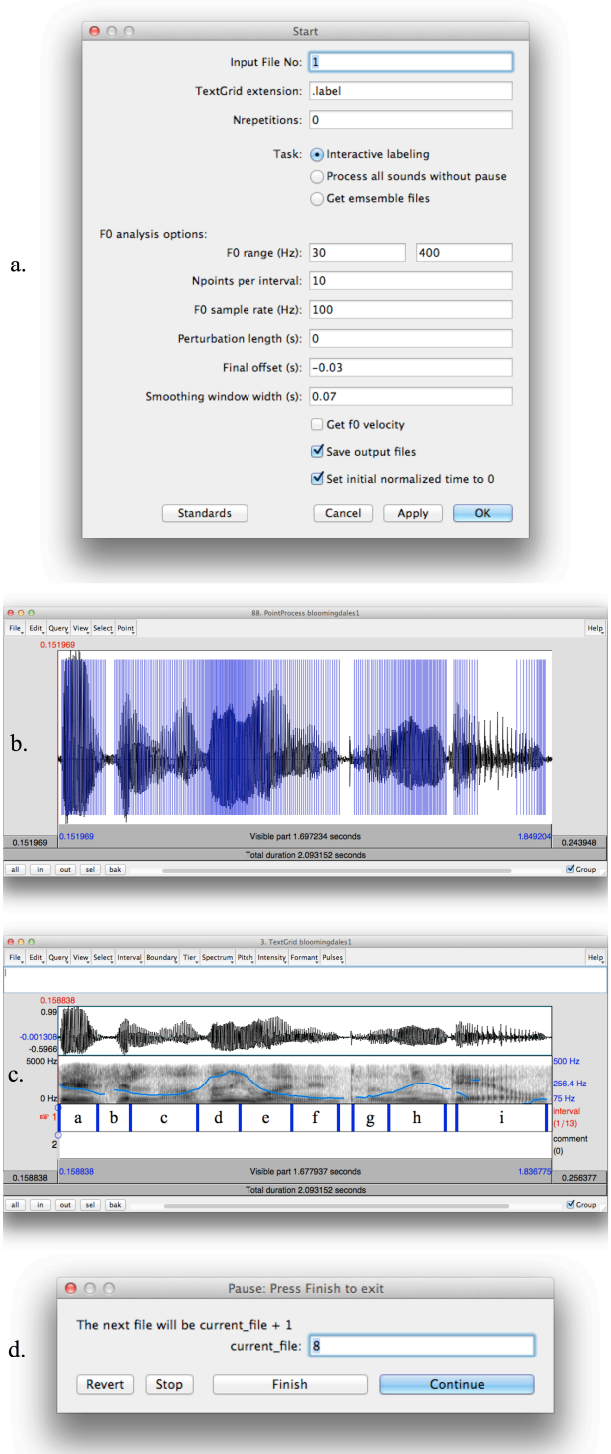

Figure 3: *ProsodyPro workflow. a. Startup window for setting control and analysis parameters, b. PointProcess window for manually rectifying vocal pulse markings, c. TextGrid window for segmentation and annotation, d. Pause window for controlling workflow.*

13. To generate mean F_0 contours averaged across speakers, the following steps can be followed:
    a. Create a text file (speaker_folders.txt) containing the speaker folder names arranged in a single column.
    b. Run ProsodyPro with the 4th task (Average across speakers) checked. The script will read mean_normf0.txt from all the speaker folders, average the f0 values on a logarithmic scale, and then convert them back to Hz.
    c. The grand averages are saved in "mean_normf0_cross_speaker.txt".

4. Limitations

Despite its many time-saving features, the use of ProsodyPro is still often labor-intensive. This is first because typical experimental datasets are large, having multiple speakers and multiple repetitions. Secondly, the rectification of vocal pulse markings can be time-consuming, which is especially the case when the expectation for accuracy of F_0 tracking is raised once human intervention is involved. While this may not be a real disadvantage given that the amount of labor is proportional to the increase in accuracy of prosody analysis, it is desirable to develop methods in future versions that can accelerate the labeling and vocal pulse marking process.

5. Conclusions

As an integrated prosody analysis tool, ProsodyPro resolves the dilemma between detailed analysis of continuous prosody and systematic comparison of discrete measurements. It also minimizes labor by automating tasks that do not require human judgment, and facilitates human intervention of processes that are prone to error, thus delivering high accuracy and reliability in prosody analysis.

6. References

[1] Bruce, G., and Touati, P., "On the analysis of prosody in spontaneous speech with exemplification from Swedish and French", Speech Communication, 11:453-458, 1992.
[2] Boersma, P., "Praat, a system for doing phonetics by computer", Glot International, 5:9/10:341-345, 2001.
[3] Hawkins, S., "Roles and representations of systematic fine phonetic detail in speech understanding", Journal of Phonetics, 31:373–405, 2003.
[4] Nakai, S., Turk, A. E., Suomi, K. et al., "Quantity constraints on the temporal implementation of phrasal prosody in Northern Finnish", Journal of Phonetics. 40(6):796-807, 2012.
[5] Post, B., D'Imperio, M., and Gussenhoven, C., "Fine phonetic detail and intonational meaning", Proceedings of The 16th International Congress of Phonetic Sciences, Saarbrücken, 191-196, 2007.
[6] Xu, Y., "Effects of tone and focus on the formation and alignment of F0 contours", Journal of Phonetics, 27:55-105, 1999.
[7] Xu, Y., "In defense of lab speech", Journal of Phonetics, 38: 329-336, 2010.
[8] Xu, Y., "Speech prosody: A methodological review", Journal of Speech Sciences, 1:85-115, 2011.
[9] Xu, Y., and Liu, F., "Tonal alignment, syllable structure and coarticulation: Toward an integrated model", Italian Journal of Linguistics, 18:125-159, 2006.
[10] Xu, Y., and Liu, F., "Intrinsic coherence of prosody and segments", Understanding Prosody – The Role of Context, Function, and Communication, O. Niebuhr, ed., 1-26: Walter de Gruyter, 2012.
[11] Xu, Y., and Xu, C. X., "Phonetic realization of focus in English declarative intonation", Journal of Phonetics, 33:159-197, 2005.