9th International Conference on Speech Prosody 2018
13-16 June 2018, Poznań, Poland

PySFC - A System for Prosody Analysis based on the Superposition of Functional Contours Model

Branislav Gerazov[1,2] *and Gérard Bailly*[2]

[1] FEEIT, University of Ss. Cyril and Methodius in Skopje, Macedonia
[2] Univ. Grenoble-Alpes, CNRS, Grenoble-INP, GIPSA-lab, 38000 Grenoble, France

gerazov@feit.ukim.edu.mk, gerard.bailly@gipsa-lab.fr

Abstract

The Superposition of Functional Contours (SFC) prosody model decomposes the intonation and duration contours into elementary contours that encode specific linguistic functions. It is based on training a set of contour generators trained in an analysis-by-synthesis loop. The model has been proven to be able to extract these functional contours for multiple linguistic tasks and for multiple languages. It has also been successfully used to decompose and then synthesise visual prosody in terms of facial expression. PySFC is a fully functional prosody analysis system built around the SFC that is completely free software. It is created to ease access to the SFC model for the speech research community, and to facilitate further development of decompositional prosody models.

Index Terms: prosody, intonation, duration, model, superposition, free software

1. Introduction

The Superposition of Functional Contours (SFC) model is a prosody model that is based on the decomposition of prosodic contours into functionally relevant elementary contours [1, 2, 3]. It proposes a generative mechanism for encoding socio-communicative functions, such as syntactic structure and attitudes, through the use of prosody.

The SFC has been successfully used to model different linguistic levels, including: attitudes [4], dependency relations of word groups (dependency to the left/right of a verb group to its subject/object noun group) [5], cliticisation [3], narrow word focus [6], tones in Mandarin [7], and finally syntactic relations between operands and operators in math [8]. It has been also successfully applied to a number of languages including: French, Galician, German [9] and Chinese [7]. Recently, the SFC model has been extended into the visual prosody domain, where it has been used to analyse and generate facial expressions and head and gaze motion control, to communicate speaker attitude [10].

PySFC is a prosody analysis system based on the SFC model that was created with two goals: *i*) to make the SFC more accessible to the scientific community, and *ii*) to serve as a platform for future development of decompositional prosody models. This paper outlines the structure of the PySFC system and gives sample results generated with its use.

2. SFC model

2.1. Contour Generators

The SFC model is based on neural network contour generators (NNCGs) that are trained to learn elementary prosodic contours [11, 4]. Each NNCG is responsible for encoding one linguistic function on one given scope. The end multiparametric prosody contour is then obtained by overlapping and adding these elementary contours according to their scopes within the utterance.

The NNCGs used in the SFC comprise a shallow feedforward neural network with one hidden layer, with a default number of 15 neurons. Four syllable position ramps are used as input that capture absolute and relative position of the syllable within the global scope of the linguistic function and also locally, i.e. in the left and right scopes as divided by the central landmark in some functions. The NNCG uses these input ramps to generate pitch and duration coefficients for each of the syllables. The training algorithm comprises of an analysis-by-synthesis loop that distributes the error at every iteration among the contour generators that are active for each of the syllables [12, 2]. Their previous output is then adjusted by the error and used as a new target for backpropagation training.

2.2. Prosodic Contour Analysis

The SFC analyses the prosody in terms of the intonation contour and rhythm of the speech signal. The analysis is based on the segmentation of the speech signal into rhythmic units (RUs). In place of syllables, the SFC preferably splits speech into Inter Perceptual Centre Group (IPCG) units that encompass the interval between two consecutive vocalic onsets [3]. This allows for a clear RU definition in languages where syllabification is not straightforward. Also, it captures the different behaviour of the syllable coda in stretching [13].

In the original SFC the intonation contour is first stylised using a second-degree polynomial and then it sampled at three points in the vocalic nucleus of each rhythmic unit (RU) at 10%, 50% and 90% of its length [14]. The SFC model is trained on these pitch targets and it is pitch targets that it outputs for synthesis. A speech synthesis system would require interpolation in order to produce an intonation pattern for the whole utterance. The f_0 targets are expressed in quarter tones:

$$f_0[1/4 \text{ tones}] = 24 \log_2 \frac{f_0[\text{Hz}]}{\hat{f}_0[\text{Hz}]}\,, \tag{1}$$

where $\hat{f}_0$ is the speaker specific reference f_0, usually the median f_0.

Duration is analysed in terms of the compression/expansion ratio coefficient of the RU [4, 15, 13]. The ratio is calculated according to:

$$RU_{ratio} = \frac{d_{RU} - \hat{d}_{RU}}{\hat{d}_{RU}}\,, \tag{2}$$

where d_{RU} is the measured and $\hat{d}_{RU}$ is expected duration of the RU calculated as:

$$\hat{d}_{RU} = (1-\alpha) \sum_{p_i \in RU} \hat{d}_{p_i} + \alpha D_{CLK}\,. \tag{3}$$

10.21437/SpeechProsody.2018-157

Here, D_{CLK} is the isochrony clock, i.e. the average RU duration for the database, and α is a coefficient that reflects the tendency towards isochrony of the language, i.e. the weight of the clock.

3. PySFC

The PySFC comprises a self-contained and fully functional ecosystem of tools for the analysis of prosody with the SFC at its core. These tools are, for the most part, used for working with the data and plotting. Python was chosen as an implementation language because of the powerful scientific computing environment that is completely based on free software and that has recently gained power and momentum in the scientific community. To enhance the accessibility of the PySFC, great attention was put on code readability, augmented with detailed functions docstrings, and comments. The system has been licensed as free software and has been made available on GitHub[1].

3.1. Modules

PySFC comprises the following modules:

- `sfc.py` – main module that controls the application of the SFC model to a chosen dataset. It is responsible for using all of the other modules to read the data, train the SFC model on it and output plots and statistics.
- `sfc_params.py` – parameter setting module that holds all the users settable parameters including:
  - data related parameters – type of input data, phrase and local level functional contours,
  - SFC hyperparameters – number of points to be sampled from the pitch contour at each vowel nucleus, pitch and duration coefficient scaling coefficients, number of iterations of analysis-by-synthesis, as well as the number of hidden units, learning rate, maximum number of iterations and regularisation coefficient for the NNCGs,
  - SFC execution parameters – use of preprocessed data, use of trained models, etc.
- `sfc_corpus.py` – holds functions used to work with the data corpus that the SFC model directly works with. This includes feature extraction from the input data and the output predictions of the SFC.
- `sfc_data.py` – comprises functions that read the input data files and calculate the f_0 and duration coefficients, as well as calculate intonation, and phone and syllable duration statistics.
- `sfc_learn.py` – holds the SFC training functions based on machine learning.
- `sfc_dsp.py` – holds DSP functions for smoothing the pitch contour.
- `sfc_plot.py` – holds the plotting functions that are used to plot the data analysis, synthesised SFC contours and SFC performance.

The size of these modules can be seen in terms of lines of code shown in Fig. 1. The plot shows that the core functionalities of the SFC are only a small part of the PySFC ecosystem, with the majority of code going into the necessary tools for working with the data, and to a lesser extent into the plotting functionalities.

[1] `https://github.com/bgerazov/PySFC`

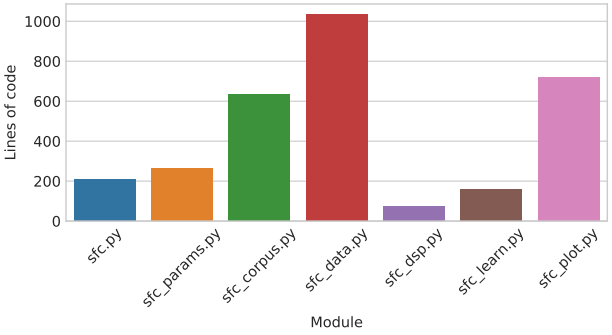

Figure 1: *Lines of code per module in the PySFC.*

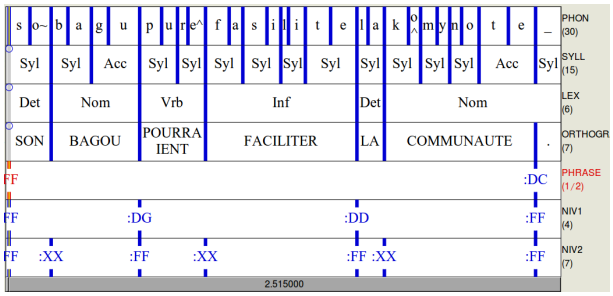

Figure 2: *Example Praat annotation of a French utterance: "Son bagou pourrait faciliter la communauté." containing functional contours: declaration (DC), dependency to the left/right (DG/DD), and cliticisation (XX).*

Currently, PySFC supports the proprietary SFC `fpro` file format as well as standard Praat `TextGrid` annotations. An example Praat annotation is shown in Fig. 2. The levels that are important for the PySFC are the phonetic (PHON) and syllable (SYLL) interval tiers, which are used for determining the vocalic nuclei and RU boundaries, and the linguistic function tiers. The latter are point tiers in which each function's scope is marked by a start and end point (":FF") and the anchor RU is marked with a landmark (":" plus the function type, e.g. ":DC"). If the function has no left or right context, the landmark itself delimits the scope, e.g. for DC. In Fig. 2, there is a PHRASE tier that holds the attitude, NIV1 with an overlapped dependency to the left (DG) and a dependency to the right (DD), and NIV2 with three clitic (XX) contours. The PySFC decomposition of the example annotated here is shown later in Fig. 3.

Pitch is calculated based on Praat `PointProcess` pitch mark files, but integration of state-of-the-art pitch extractors [16] is planned for the future. PySFC also brings added value, by adding the possibility to adjust the number of samples to be taken from the pitch contour at each rhythmical unit vowel nucleus, and with its extended plotting capabilities for data and performance analysis.

4. Method

4.1. Databases

In order to assess the functionality of PySFC, two databases were used. One comprising emphatic speech in French focused on speaker attitudes uttered in various syntactic organisations and lengths of constituents, and the second comprising read Chinese:

- `db_morlec` – a database of 6 attitudes in French: declarative (DC), question (QS), exclamation (EX), incredulous question (DI), suspicious irony (SC) and obviousness (EV) [4]. In total 1932 utterances were recorded from a single speaker.
- `db_chen` – a database of two attitudes in Chinese: declarative (DC) and question (QS) [7]. In total 110 utterances were recorded from a single speaker.

4.2. Database analysis

PySFC incorporates tools to extract pitch, phone and syllable duration statistics from all files in a database. It then calculates the mean, median and Kernel Density Estimation (KDE) peak for each distribution. These can be used by the user to specify the f_{0ref} needed for normalising the f_0, and the D_{CLK} for the duration ratio coefficients, as well as for calculating the phone duration means.

5. Results

5.1. Intonation decomposition

Example plots of the PySFC decomposition for an utterance from `db_morlec` and `db_chen` is shown in Figs. 3 and 5. The plot shows the original intonation contour and its SFC reconstruction, as well as the decomposition of the intonation contour into its elementary functional contours.

5.2. Performance plots

Expansion plots for the functional contour generators can be used to assess the validity of the training results. Fig. 4 shows example expansion plots for `db_morlec` for dependency to the left (DG), while Fig. 6 shows expansions for the attitudes: declaration DC, question QS, incredulous question DI, and obviousness EV. In the expansion plot we can see how the generated contours change with expanding left and right contexts. The numbers next to the contours show how many scopes of that kind have been found in the database. We can see in Fig. 4 that there are two DG scopes that were not found in the data, but that are successfully generated by the trained NNCG.

PySFC also allows for plotting the loss in terms of error of the SFC reconstruction. Examples are shown in Fig. 7 for losses per iteration for each NNCG for the attitude DC for `db_chen`, and the final losses for all the NNCGs for each of the attitudes in `db_morlec`. The top plot shows how the losses decrease for each contour generator with each iteration, the error distribution being close to its asymptote after 10 iterations of the analysis-by-synthesis loop. The plot on the right gives a heat map of the final losses after 20 iterations for each of the functional NNCGs within each attitude.

6. Conclusions

PySFC is a self-contained prosody analysis system based on the SFC model implemented in the scientific Python ecosystem. The main goal of PySFC is enabling access to the SFC for the broader scientific community, but also serving as a platform to facilitate further development of decompositional prosody models. One improvement that is already incorporated is the possibility to choose the number of samples taken from the vocalic nuclei. Another is the inclusion of the concept of gradience that allows for different amplitudes of each contour generator in the final sum contour. Different ways can also be explored for the distribution of the error among the NNCGs in the analysis-by-synthesis loop. The PySFC system will facilitate the rapid development and assessment of these ideas.

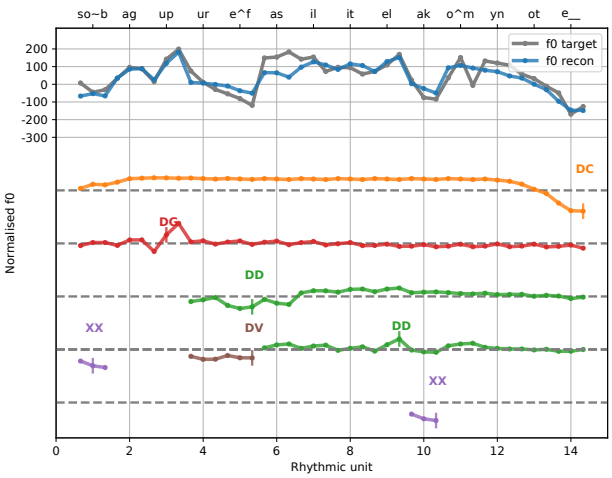

Figure 3: *Example PySFC intonation decomposition for the French utterance: "Son bagou pourrait faciliter la communauté." into constituent functional contours: declaration (DC), dependency to the left/right (DG/DD), and cliticisation (XX, DV).*

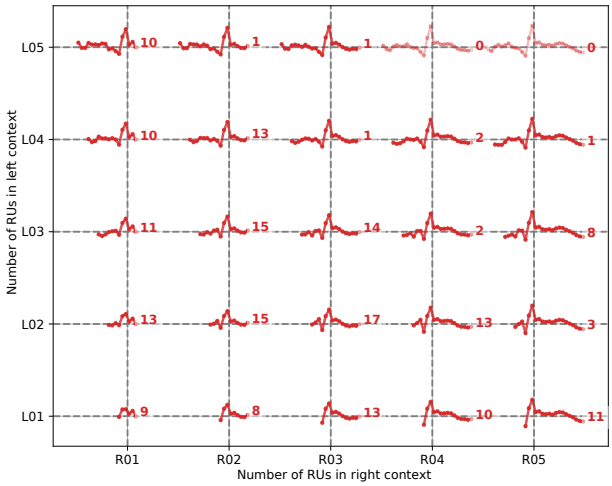

Figure 4: *Example PySFC expansion in left and right context for the dependency to the left (DG) functional contour, numbers next to the plots show the number of occurences of that scope in the data.*

7. Acknowledgements

This work has been conducted with the support of the Horizon Horizon 2020 Marie Skłodowska-Curie Actions Individual Fellowship Project, Call H2020-MSCA-IF-2016, under the project "ProsoDeep: Deep understanding and modelling of the hierarchical structure of Prosody".

8. References

[1] Gérard Bailly and Bleicke Holm, SFC: a trainable prosodic model," *Speech communication*, vol. 46, no. 3, pp. 348–364,

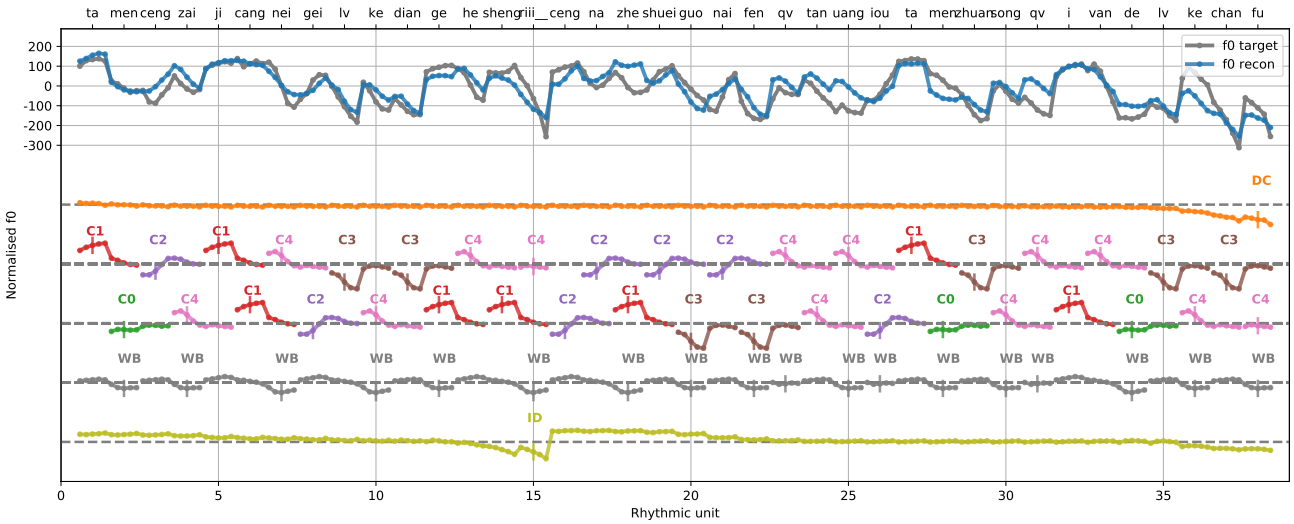

Figure 5: *Example PySFC intonation decomposition for the Chinese utterance: "Ta1 men ceng2 zai4 ji1 cang1 nei4 gei2 lv3 ke4 dian3 ge1 he4 sheng1 ri4, ceng2 na2 zhe1 shui2 guo3 nai2 fen3 qu4 tan4 wang4 you2 ta1 men zhuan3 song4 qu4 yi1 yuan4 de lv3 ke4 chan3 fu4." into constituent functional contours: declaration (DC), tones (C0-4), word boundaries (WB), and independence (ID).*

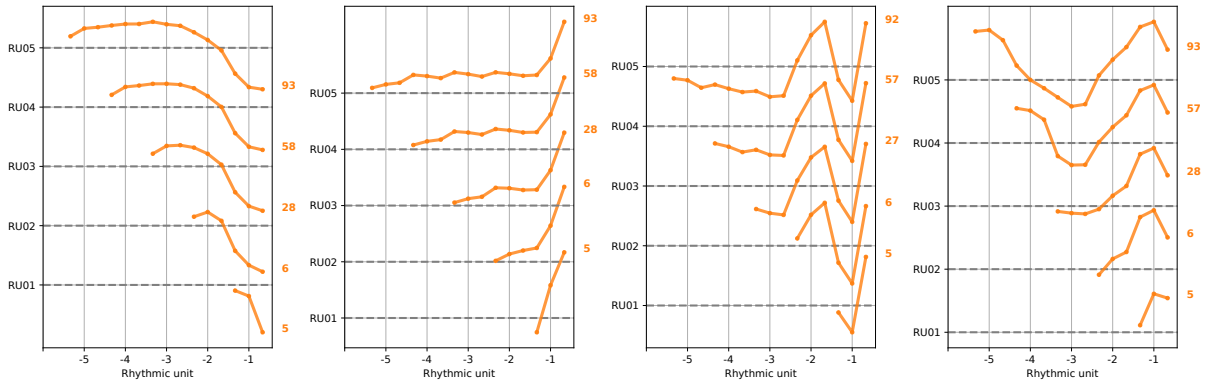

Figure 6: *Example PySFC expansion for: the assertion (DC), question (QS), incredulous question (DI), and obviousness (EV) attitudes contours, numbers next to the plots show the number of occurences of that scope in the data.*

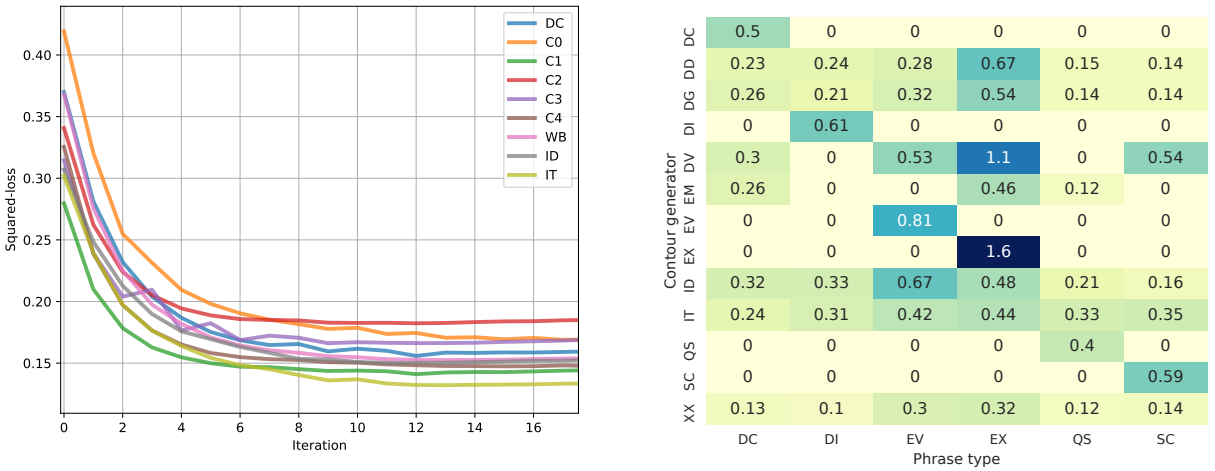

Figure 7: *Example PySFC plots of f_0 reconstruction losses for all NNCGs for attitude DC per iteration for Chinese (left) and final losses for all NNCGs for all attitudes for French as a heat map (right).*

2005.

[2] B Holm and Gérard Bailly, Learning the hidden structure of intonation: implementing various functions of prosody," in *Speech Prosody 2002, International Conference*, 2002.

[3] Gérard Bailly and Bleike Holm, Learning the hidden structure of speech: from communicative functions to prosody," *Cadernos de Estudos Linguisticos*, vol. 43, pp. 37–54, 2002.

[4] Yann Morlec, Gérard Bailly, and Véronique Aubergé, Generating prosodic attitudes in French: data, model and evaluation," *Speech Communication*, vol. 33, no. 4, pp. 357–371, 2001.

[5] Yann Morlec, Albert Rilliard, Gérard Bailly, and Véronique Aubergé, Evaluating the adequacy of synthetic prosody in signalling syntactic boundaries: methodology and first results," in *Proceedings of the first International Conference on Language Resources and Evaluation. Granada, Spain*, 1998, pp. 647–650.

[6] C. Brichet and V. Aubergé, La prosodie de la focalisation en français: faits perceptifs et morphogénétiques," *Journées d'Etudes sur la Parole, Nancy-France*, pp. 33–36, 2004.

[7] Gao-Peng Chen, Gérard Bailly, Qing-Feng Liu, and Ren-Hua Wang, A superposed prosodic model for Chinese text-to-speech synthesis," in *Chinese Spoken Language Processing, 2004 International Symposium on*. IEEE, 2004, pp. 177–180.

[8] Bleicke Holm, Gérard Bailly, and Colette Laborde, Performance structures of mathematical formulae," in *Proceedings of the International Congress of Phonetic Sciences*, 1999, vol. 2, pp. 1297–1300.

[9] Gérard Bailly and Ian Gorisch, Generating German intonation with a trainable prosodic model," in *Interspeech*, 2006, pp. 2366–2369.

[10] Adela Barbulescu, *Generation of audio-visual prosody for expressive virtual actors*, Ph.D. thesis, Université de Grenoble, 2015.

[11] Yann Morlec, *Génération multiparamétrique de la prosodie du français par apprentissage automatique*, Ph.D. thesis, 1997.

[12] Bleicke Holm and Gérard Bailly, Generating prosody by superposing multi-parametric overlapping contours.," in *INTERSPEECH*, 2000, pp. 203–206.

[13] W Nick Campbell, Syllable-based segmental duration," *Talking machines: Theories, models, and designs*, pp. 211–224, 1992.

[14] Stéphanie de Tournemire, Identification and automatic generation of prosodic contours for a text-to-speech synthesis system in French," in *Fifth European Conference on Speech Communication and Technology*, 1997, pp. 191–194.

[15] Plínio Barbosa and Gérard Bailly, Characterisation of rhythmic patterns for text-to-speech synthesis," *Speech Communication*, vol. 15, no. 1-2, pp. 127–137, 1994.

[16] Pegah Ghahremani, Bagher BabaAli, Daniel Povey, Korbinian Riedhammer, Jan Trmal, and Sanjeev Khudanpur, A pitch extraction algorithm tuned for automatic speech recognition," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. IEEE, 2014, pp. 2494–2498.