The Flexible Audio Source Separation Toolbox Version 2.0

Yann Salaün, Emmanuel Vincent, Nancy Bertin, Nathan Souviraà-Labastie, Xabier Jaureguiberry, Dung T. Tran, Frédéric Bimbot

THE FLEXIBLE AUDIO SOURCE SEPARATION TOOLBOX VERSION 2.0

Yann Salaün[1], Emmanuel Vincent[1], Nancy Bertin[2], Nathan Souviraà-Labastie[3], Xabier Jaureguiberry[4], Dung T. Tran[1], and Frédéric Bimbot[2]

[1]Inria, Villers-lès-Nancy, F-54600, France
[2]CNRS, IRISA - UMR 6074, Rennes, F-35042, France
[3]Université Rennes 1, IRISA - UMR 6074, Rennes, F-35042, France
[4] Institut Mines-Télécom, Télécom ParisTech, CNRS LTCI, Paris, F-75014, France
yann.salaun@inria.fr

ABSTRACT

The Flexible Audio Source Separation Toolbox (FASST) is a toolbox for audio source separation that relies on a general modeling and estimation framework that is applicable to a wide range of scenarios. We introduce the new version of the toolbox written in C++, which provides a number of advantages compared to the first Matlab version: portability, faster computation, simplified user interface, more scripting languages. In addition, we provide a state-of-the-art example of use for the separation of speech and domestic noise. The demonstration will give attendees the opportunity to explore the settings and to experience their effect on the separation performance.

1. INTRODUCTION AND MOTIVATIONS

Source separation is one of the major topics in audio signal processing. Recent years have seen a move from blind to guided approaches incorporating more and more knowledge about the sources and/or the mixing process [1].

While most source separation methods are designed for a specific scenario, the flexible audio source separation framework in [2] introduced a compositional approach [3] where the mixture signal is modeled by composing multiple source models together. Each model is parameterized by a number of variables, which may be constrained by the user, trained from separate data or adapted to the considered mixture according to the available information. This framework has been applied to a wide variety of speech and music separation scenarios by exploiting information such as note spectra, cover music recordings, reference speech pronounced by another speaker, or target spatial direction [4–8]. It has also been used as a preprocessing step for instrument recognition, beat tracking, and automatic speech recognition [8, 9].

This framework was first implemented in Matlab as version 1.0 of the Flexible Audio Source Separation Toolbox (FASST) which was not demonstrated to the public. Although it facilitates quick prototyping of software modifications, this choice implied large computation time and limited diffusion to communities which do not routinely use Matlab. We introduce the version 2.0 of FASST in C++ which fixes these limitations and provides a simpler user interface. In addition, we provide a new example of use for speech denoising in the context of the 2nd CHiME Challenge [10].

The structure of the rest of the paper is as follows. Section 2 summarizes the framework behind FASST. Section 3 introduces the new implementation and user interface. Future developments are discussed in Section 4.

2. SCIENTIFIC AND TECHNICAL DESCRIPTION

FASST operates in the time-frequency domain. In each time-frequency bin (n, f), the vector $\mathbf{x}_{fn}$ of mixture STFT coefficients recorded at all microphones satisfies

$$\mathbf{x}_{fn} = \sum\nolimits_{j=1}^{J} \mathbf{y}_{jfn} \tag{1}$$

where $\mathbf{y}_{jfn}$ is the *spatial image* of the jth source. The sources are assumed to be zero-mean Gaussian distributed as

$$\mathbf{y}_{jfn} \sim \mathcal{N}(\mathbf{0}, v_{jfn}\mathbf{R}_{jf}) \tag{2}$$

where v_{jfn} denotes the short-term *power spectrum* of the jth source and $\mathbf{R}_{jf}$ its *spatial covariance matrix*.

The power spectrogram $\mathbf{V}_j$ of each source (i.e., the matrix whose (f, n)th entry is v_{jfn}) is further factored according to an excitation-filter model followed by two-level nonnegative matrix factorization (NMF). Overall,

$$\mathbf{V}_j = \left(\mathbf{W}_j^{\mathrm{ex}}\,\mathbf{U}_j^{\mathrm{ex}}\,\mathbf{G}_j^{\mathrm{ex}}\,\mathbf{H}_j^{\mathrm{ex}}\right) \odot \left(\mathbf{W}_j^{\mathrm{ft}}\,\mathbf{U}_j^{\mathrm{ft}}\,\mathbf{G}_j^{\mathrm{ft}}\,\mathbf{H}_j^{\mathrm{ft}}\right) \tag{3}$$

where the nonnegative matrices $\mathbf{W}_j^{\mathrm{ex}}$, $\mathbf{U}_j^{\mathrm{ex}}$, $\mathbf{G}_j^{\mathrm{ex}}$ and $\mathbf{H}_j^{\mathrm{ex}}$ encode the fine spectral structure, the spectral envelope, the temporal envelope and the temporal fine structure of the excitation, respectively $\mathbf{W}_j^{\mathrm{ft}}$, $\mathbf{U}_j^{\mathrm{ft}}$, $\mathbf{G}_j^{\mathrm{ft}}$ and $\mathbf{H}_j^{\mathrm{ft}}$ encode the same quantities for the resonance filter, and $\odot$ denotes entry-wise

matrix multiplication. The spatial covariance matrix is itself factored as

$$\mathbf{R}_{jf} = \mathbf{A}_{jf}\mathbf{A}_{jf}^{H} \quad (4)$$

where the dimension of $\mathbf{A}_{jf}$ governs the rank of $\mathbf{R}_{jf}$. Each of the 9 parameters in (3) and (4) may be either fixed or adaptive depending on the information available about, e.g., the spatial position of the sources and the harmonicity of their spectra.

Adaptive parameters are estimated in the maximum likelihood (ML) sense by iteratively fitting the empirical mixture covariance matrix $\widehat{\mathbf{R}}_{\mathbf{x},fn}$ using an expectation-maximization (EM) algorithm. The source spatial image signals are eventually obtained by multichannel Wiener filtering. For more details about the algorithm and example settings, see [2].

3. IMPLEMENTATION AND USE

FASST 2.0 is composed of two parts: binary executables written in C++ and user scripts written in Matlab and Python. Code and sound examples are distributed online under the QPL and Creative Commons open source licences[1].

3.1. C++ core

The core of FASST is composed of three C++ programs illustrated in Fig. 1:

1. *Representation*
   The first program takes as input a time-domain mixture WAV file and it computes its time-frequency domain empirical covariance matrix $\widehat{\mathbf{R}}_{\mathbf{x},fn}$. Since this matrix is Hermitian, we developed a binary format to save disk space by storing non-redundant entries only.

2. *Parameter estimation*
   The second program performs ML parameter estimation given the above time-frequency representation and a choice of model. Choosing the model consists of specifying the initial value and the fixed/adaptive character of each the 9 parameters in (3) and (4) for each source. Due to the hierarchical structure of this model, we store the input and output model structure and parameter values in an XML file, which can be edited using a wide variety of tools.

3. *Filtering*
   The third program performs Wiener filtering given the input mixture WAV file and the estimated parameters XML file.

These programs rely on standard third-party libraries: libsndfile, Qt, and Eigen. In addition to the portability of C++, FASST 2.0 offers two advantages. Firstly, the parameter estimation code takes full advantage of multicore hardware using OpenMP: computation time can reach a 3x speedup compared to the original implementation in Matlab. Secondly,

[1] http://bass-db.gforge.inria.fr/fasst/

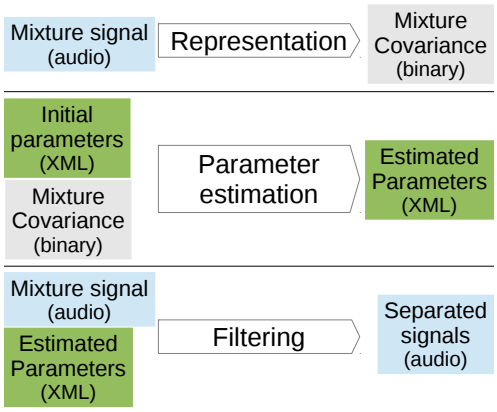

Fig. 1. The three core executables of FASST 2.0.

the user interface has been simplified. Specification of each of the 9 parameters of each source is not required anymore: only those parameters which are adaptive or different from the identity matrix must be specified (e.g., only the basis spectra $\mathbf{U}_j^{\text{ex}}$ and the activations $\mathbf{G}_j^{\text{ex}}$ in the case of conventional, one-level NMF) and the others are assumed to be fixed and equal to the identity matrix by default.

The demonstration will give attendees the opportunity to manipulate these parameters and to listen to the resulting separation performance.

3.2. User scripts

Similarly to speech recognition software, scripts must be written in order to glue the three core executables together as appropriate for a given source separation problem. The simplest workflow is to initialize the source model, to write it to an XML file, and to call the executables in the above order. More complicated workflows involve multiples call to the executables in order to learn models from separate data before applying them to the considered mixture. FASST 2.0 includes scripts allowing to import/export binary and XML files to/from Matlab and Python. This also increases the portability of the software compared to the original version which included Matlab scripts only.

3.3. Baseline for CHiME

As an advanced example, we provide new scripts for the separation of speech and real-world domestic noise as evaluated in Track 1 of the 2nd CHiME Challenge [10]. The workflow is depicted in Fig. 2. Speech models are trained on separate speaker-dependent data and kept fixed. Noise models are initialized by training from the background noise surrounding the considered mixture and reestimated from the mixture.

Research has been carried regarding model initialization to favor convergence of EM to a relevant local optimum. The basis spectra are initialized by split vector quantization of the

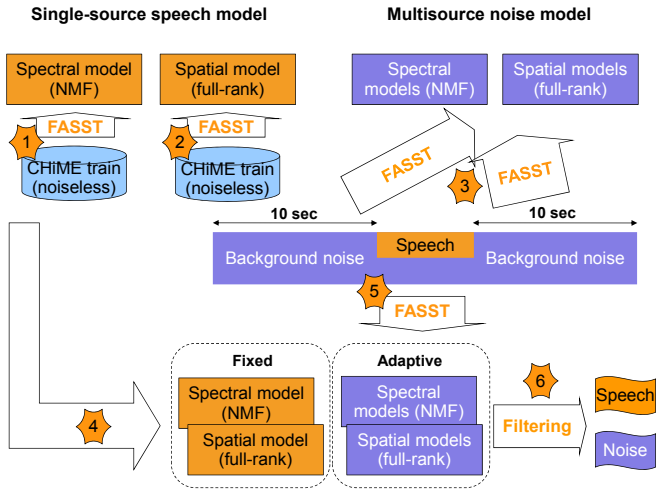

Fig. 2. Speech denoising workflow for the CHiME Challenge. Numbers denote successive steps and "FASST" denotes a call to "Representation" followed by "Parameter estimation".

Table 1. SDR (dB) on the CHiME Challenge subset in [11].

iSNR	-6 dB	-3 dB	0 dB	3 dB	6 dB	9 dB	avg.
FASST 2.0	6.0	7.8	5.2	12.7	10.5	12.4	9.1
Nesta [11]	5.2	6.2	5.6	9.2	11.6	10.5	8.0

input short-term spectra, the spatial parameters are initialized by splitting the space of directions-of-arrival into regions of equal surface, and the activations on the considered mixture are initialized by the mean activations on the training data. On average, the overall separation quality measured by the signal-to-distortion ratio in decibels (dB) outperforms the best quality reported so far in [11], as shown in Table 1.

4. CONCLUSION AND FUTURE DEVELOPMENT

Thanks to its general modeling and estimation framework, FASST is to our knowledge the most widely applicable source separation software currently available, as exemplified by its use for different scenarios in [4–9]. The demonstration will introduce the advantages of FASST 2.0 compared to the original version in Matlab, which had not been demonstrated to the public so far, and it will allow the audience to interact with the settings. Future planned developments include extending the code for real-time operation and integrating it with automatic speech recognition software using uncertainty propagation.

5. ACKNOWLEDGMENT

This work was supported by the Inria ADT FASST project. We would like to thank O. Rochel, J. Espiau de Lamaëstre, D. Jouvet and G. Gravier for their advice.

6. REFERENCES

[1] E. Vincent, N. Bertin, R. Gribonval, and F. Bimbot, "From blind to guided audio source separation," *IEEE Signal Processing Magazine*, to appear.

[2] A. Ozerov, E. Vincent, and F. Bimbot, "A general flexible framework for the handling of prior information in audio source separation," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 4, pp. 1118 – 1133, 2012.

[3] T. Virtanen, J. F. Gemmeke, B. Raj, and P. Smaragdis, "Compositional models for audio processing," *IEEE Signal Processing Magazine*, to appear.

[4] T. Gerber, M. Dutasta, and L. Girin, "Professionally-produced music separation guided by covers," in *Proc. 13th Int. Society for Music Information Retrieval Conf.*, 2012, pp. 85–90.

[5] Y.-H. Yang, "On sparse and low-rank matrix decomposition for singing voice separation," in *Proc. 20th ACM Int. Conf. on Multimedia*, 2012, pp. 757–760.

[6] K. Han and D. L. Wang, "Towards generalizing classification based speech separation," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 1, pp. 168–177, 2013.

[7] L. Le Magoarou, A. Ozerov, and N. Q. K. Duong, "Text-informed audio source separation using nonnegative matrix partial co-factorization," in *Proc. 2013 IEEE Int. Workshop on Machine Learning for Signal Processing*, 2013, pp. 1–6.

[8] D. T. Tran, E. Vincent, D. Jouvet, and K. Adiloğlu, "Using full-rank spatial covariance models for noise-robust asr," in *Proc. 2nd Int. Workshop on Machine Listening in Multisource Environments*, 2013, pp. 31–32.

[9] J. J. Bosch, J. Janer, F. Fuhrmann, and P. Herrera, "A comparison of sound segregation techniques for predominant instrument recognition in musical audio signals," in *Proc. 13th Int. Society for Music Information Retrieval Conf.*, 2012, pp. 559–564.

[10] E. Vincent, J. Barker, S. Watanabe, J. Le Roux, F. Nesta, and M. Matassoni, "The second 'CHiME' speech separation and recognition challenge: An overview of challenge systems and outcomes," in *Proc. 2013 IEEE Workshop on Automatic Speech Recognition and Understanding*, 2013, pp. 162–167.

[11] F. Nesta and M. Matassoni, "Blind source extraction for robust speech recognition in multisource noisy environments," *Computer Speech and Language*, vol. 27, no. 3, pp. 703–725, 2013.