# A framework for music analysis/resynthesis based on matrix factorization

**Juan José Burred**
Paris, France
jjburred@jjburred.com

## ABSTRACT

Spectrogram factorization is a recent and promising alternative to sinusoidal or source/filter modeling for analysis/resynthesis systems aimed at musical creation. This paper presents a framework designed to perform a wide range of sound manipulations based on Non-negative Matrix Factorization (NMF), including a set of new techniques for creating artificial cross-components not present in the original analyzed sound. The system can process individual sounds by modifying their internal structure, or can be used for a flexible type of cross-synthesis between two input sounds. The different processing modules are illustrated by a collection of sound examples available on a companion website.

## 1. INTRODUCTION

Analysis/resynthesis is one of the most widely used paradigms of sound creation in computer and electronic music. It comprises the successive stages of sound analysis, modification of the extracted parameters, and resynthesis of a new sound from the modified parameters. In a wide sense, the resynthesis stage can be either fully electronic or computer-based, or performed by acoustical instruments following scores or instructions derived from the analyzed parameters. The latter approach was a central technique used by the first spectralist composers from the 1970's. In the present article however, analysis/resynthesis is interpreted as a fully computer-based processing chain: the user has access to the parameters and can modify them, but the analysis and resynthesis stages are automatic.

Many sound analysis techniques, with their corresponding resynthesis counterpart, have been proposed over the last decades. Arguably, the most popular one is sinusoidal analysis/resynthesis, in which the main parameters are the time-varying frequencies, amplitudes and phases of the sinusoidal partials contained in the analyzed sound. Analysis methods for such sinusoidal techniques include the Short Time Fourier Transform (STFT) and the Phase Vocoder [1]. Another important family of methods is source/filter analysis/resynthesis, which is based on the estimation and manipulation of spectral envelopes with methods such as Linear Predictive Coding (LPC) [2]. As an example of a more recent approach, sparse decomposition into time-frequency atoms has been used for this purpose [3].

Matrix factorization has been proposed in recent years as a new analysis technique applied to musical creation [4, 5, 6]. Due to its ability to reveal latent sound sources when applied to spectrograms, matrix factorization has been, and still is, the method of choice in sound source separation, and there is a huge body of literature in that area proposing many variations on such algorithms for the purpose of extracting instrumental sources from a mix. Nevertheless, the potential of matrix factorization for the creation of new sounds has been seldom explored.

When applied to a time-frequency matrix $\mathbf{X}$ of size $F \times T$ ($F$ frequency bands and $T$ temporal frames), such as a magnitude spectrogram, a matrix factorization algorithm yields two factor matrices ($\mathbf{W}$ and $\mathbf{H}$) that approximate the original matrix when multiplied: $\mathbf{X} \approx \mathbf{WH}$. Matrix $\mathbf{W}$ is of size $F \times K$ and matrix $\mathbf{H}$ is of size $K \times T$, where $K$ is a parameter set by the user. The factor matrices can be interpreted as follows: each column $\mathbf{w}_k$ of $\mathbf{W}$ is a spectrum of $F$ bins, and each row $\mathbf{h}_k$ of $\mathbf{H}$ is a temporal function (or *activation*) of $T$ frames. The outer product of each spectrum $\mathbf{w}_k$ with each activation $\mathbf{h}_k$ (denoted by $\mathbf{w}_k \otimes \mathbf{h}_k$) produces a $F \times T$ spectrogram that is called a *component*. Each component can be visualized as a time-frequency layer; when all $K$ layers are added, an approximation to the original matrix $\mathbf{X}$ is obtained. Fig. 1 illustrates the factorization of a magnitude spectrogram of a three-note piano melody with $K = 3$. The resulting activation functions are plotted horizontally above the spectrogram, and the spectra are plotted vertically to its left.

Thus, after an analysis by factorization, the composer or

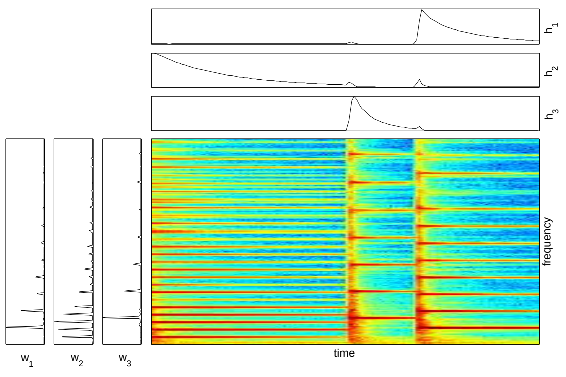


**Figure 1**. Illustration of spectrogram factorization by NMF: three-note piano melody.

sound designer has at his disposal a collection of $K$ spectra and $K$ activations, or alternatively a set of $K$ components obtained by multiplying pairs of spectra and activations. The value of $K$ and the spectra and activations are the parameters that can be modified for resynthesis purposes. This contrasts with sinusoidal analysis/resynthesis, where the elements are individual sinusoids, instead of full spectra, and the number of parameters to manipulate is usually much higher. Also, when compared to source/filter methods, factorization approaches have the ability to process internal sound components individually, instead of processing global features that act on the whole sound, such as spectral envelopes. Depending on the used algorithm and on the selected number of components $K$, the factorization components can correspond to individual resonances, transients, salient events or notes. The manipulation by the user of such sub-events is thus a potentially powerful new method of sound creation.

In their 2011 paper [4], Topel and Casey review several compositions based on matrix factorization, in particular based on Probabilistic Latent Component Analysis (PLCA). The works cited therein exploit different techniques, from manual arrangement and instrumentation based on the extracted parameters (in the spirit of spectralism) to automatic matching of live sounds to PLCA components based on a similarity measure. Sarver and Klapuri [5] propose the use of Non-Negative Matrix Factorization (NMF) for sound effects processing. The number and weights of each of the components are controlled to generate timbre modifications and compression and distortion effects.

Most of the cited approaches work by performing modifications at the component level: each spectrum $\mathbf{w}_k$ remains coupled with its corresponding activation $\mathbf{h}_k$ with the same index $k$. This is necessary if the final sound needs to be a close approximation to the input, and is thus essential in source separation or effects processing. For the purposes of sound synthesis however, it is possible to go one step further and decouple the spectra from their corresponding activations: by multiplying pairs of $\mathbf{w}_i$ and $\mathbf{h}_j$ such that $i \neq j$, new sounds are created that were not present in the original sound. These can be called *cross-components*. Such an approach was used in a previous work by this author [6], where automatic cross-synthesis based on NMF was proposed. In it, activations from a source sound are combined with spectra from a target sound, creating a hybrid sound where the temporal structure is provided by the source, and the timbre by the target.

In the present work, that idea is further explored, and the system has been extended with several new methods for automatic factorization-based sound creation and modification. The principle of automatic cross-component resynthesis has been extended from cross-synthesis of two sounds to the processing of single sounds, and thus the system has been generalized from a cross-synthesis framework to a full analysis/resynthesis framework. More flexibility has been added by introducing the possibility of automatic selection of spectra or activations based on objective features. Furthermore, the sound quality has been significantly improved by using a resynthesis stage based on Wiener filtering. All operations will be illustrated by several sound examples available online [1].

The ultimate goal of this line of research is to develop a comprehensive analysis/synthesis software framework that will allow composers and sound designers to exploit the many possibilities of factorization-based processing, including manual or automatic component-based, cross-component and cross-synthesis processing. A preliminary implementation, called *Factorsynth*, is available for download (Sect. 6).

The two main sections of the paper concern the two main synthesis modes implemented: Sect. 3 details several new modules to perform cross-component synthesis from an individual input sound. Sect. 4 briefly summarizes the cross-synthesis system previously introduced in [6], and introduces its new extensions, including the new activation mapping module and constrained cross-synthesis. Finally, Sect. 5 briefly discusses the new resynthesis module.

## 2. ANALYSIS STAGE

The first processing step for each input sound $s(n)$ is the extraction of its magnitude spectrogram matrix as the absolute value of its STFT: $\mathbf{X} = |\mathrm{STFT}\{s(n)\}|$. The system accepts stereo or multichannel signals, but handles them by processing each channel separately, so notation denoting multiple channels can be ignored. The spectrogram is subjected to NMF, which requires that all elements on the input and output matrices have to be zero or positive. NMF is an iterative algorithm implemented as an optimization that minimizes the reconstruction error, given by the sum of an element-wise distance measure between observation $\mathbf{X}$ and approximation $\mathbf{WH}$:

$$D = \sum_{f=1}^{F} \sum_{t=1}^{T} d(\mathbf{X}_{(t,f)}, \mathbf{WH}_{(t,f)}). \tag{1}$$

Different NMF algorithms exist, depending on the choice of distance measure. Three possible choices were implemented and tested: the Frobenius norm (used in [6]), the Kullback-Leibler (KL) divergence [7] and the Itakura-Saito (IS) divergence [8]. After some informal listening, best-sounding results were obtained by the KL divergence, given by

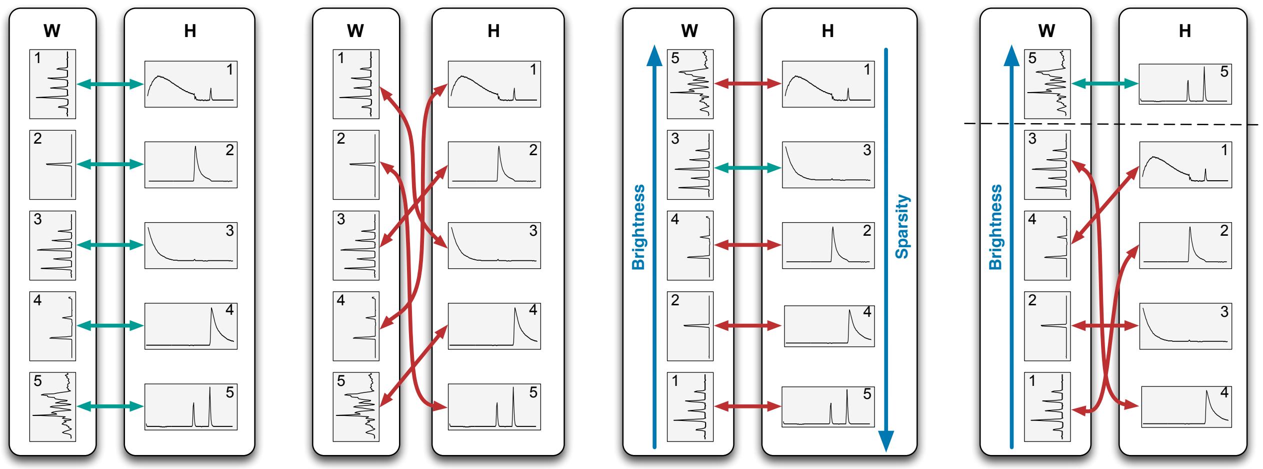
$$d_{KL}(x, y) = x \log \frac{x}{y} - x + y, \tag{2}$$

which was henceforth used [2]. The use of the KL divergence as error measure leads to a set of simple multiplicative update rules, first derived by Lee and Seung in their 2001 paper [7].

The crucial parameter at the analysis stage is the number of components $K$, since it determines the level of detail of the components. It is the single parameter having the most important timbral consequences on the result. Several guidelines on how to choose $K$ depending on the desired results are given on the previous work [6].

[1] `http://jjburred.com/research/icmc2014`

[2] More formal analysis and listening tests will be needed for a definitive choice.



(a) Reconstruction (no processing) (b) Scramble (c) Rank (d) Constrained scramble

**Figure 2**. Conceptual overview of single-sound factorization-based operations. Vertically plotted curves are spectra, horizontally plotted curves are temporal activations. The numbers are the original component indices after NMF. Green arrows correspond to components, red arrows to cross-components.

## 3. CROSS-COMPONENT PROCESSING OF INDIVIDUAL SOUNDS

After factorization by NMF, the sound is decomposed into a set of $K$ spectra (or *spectral bases*) and $K$ temporal activation functions. A distinction will be made between *components* and *cross-components*:

$$\text{Components}: \quad \mathbf{C}_k = \mathbf{w}_k \otimes \mathbf{h}_k, \tag{3}$$

$$\text{Cross-components}: \quad \mathbf{C}_{ij} = \mathbf{w}_i \otimes \mathbf{h}_j, \quad i \neq j \tag{4}$$

That is, components are obtained by multiplying spectra and activations of the same index, and are actual subentities of the original sounds. Cross-components, in contrast, are new sounds not present in the original input, created by artificially combining spectra and activations of different indexes. There are $K$ possible components and $K^2 - K$ possible cross-components (it should be noted that $\mathbf{C}_{ij} \neq \mathbf{C}_{ji}$).

Source separation and NMF-based effects processing as proposed in [5] both work by processing or filtering out components. Here, the focus is on cross-components, and three possible operations based on them are proposed in the following subsections: *scramble*, *rank*, and *constrained scramble*.

### 3.1 Scramble

The simplest way to obtain an output sound consisting entirely of cross-components is to do a "scramble" (random permutation) of either the spectra or activation indices before multiplication, keeping the other index set unchanged (in other words, the multiplication pairs are randomly chosen). This forces each and every spectra to be multiplied by an originally unrelated activation (see Fig. 2(b)). The main drawback of such an operation is obviously the lack of control and unpredictability of results (beyond the ability to control the decomposition level $K$).

The audible results of scrambling could perhaps be described as follows: the overall, *external* timbre and temporal discourse of the original sound are both recognizable, but the *internal* pitch (harmonic, resonance) contents is completely different.

The best is to illustrate this with some example sounds, available on the previously cited webpage. In the first example (Sound 1), the first few measures of Wagner's "Tristan und Isolde" are factorized into $K = 20$ components, and subsequently scrambled. In the resulting sound, it is possible to hear timbral elements from the original, arranged in such a way that the dynamic evolution is maintained (note the emphasis on the main chord). In the second example (Sound 2), an excerpt of the German Requiem by Brahms, the dynamic evolution is flatter and the texture is highly homophonic. The resulting sound keeps the timbral and texture contents, but completely alters the harmony.

### 3.2 Rank

A way of introducing some degree of control to automatic cross-component processing is to independently sort the spectra and activations following objective measures. It should be noted that NMF and other factorization algorithms suffer from the *permutation problem*: the ordering of the components is random [3], thus the value of the individual $k$s has no physical interpretation whatsoever.

Here, it was chosen to rank the spectra by brightness, measured by the spectral centroid:

$$SC_k = \frac{\sum_{n=1}^{F} f(n)\mathbf{w}_k(n)}{\sum_{n=1}^{F} \mathbf{w}_k(n)}, \tag{5}$$

[3] The random ordering of the components (indices $k$) produced by the permutation problem should not be confused with the random coupling between spectra and activations introduced by the scramble operation.

where $f(n)$ is the frequency at bin $n$, and the activations by sparsity, measured by the kurtosis:

$$K_k = \frac{\frac{1}{T}\sum_{t=1}^{T}(\mathbf{h}_k(t) - \mu_k)^4}{(\frac{1}{T}\sum_{t=1}^{T}(\mathbf{h}_k(t) - \mu_k)^2)^2}, \qquad (6)$$

where $\mu_k$ is the empirical mean of $\mathbf{h}_k(t)$. These were also the two measures used in [5] for the rearrangement of components, and were chosen here as well due to their simplicity and perceptual relevance.

The spectra and activations are independently sorted according to centroid and kurtosis, and then multiplications are carried out for creating the cross-components (see Fig. 2(c)). In some cases, the spectra will find themselves opposite their original activation partners after ranking. For instance, ranking the spectra by increasing brightness and the activations by increasing sparsity turned out in preliminary tests to produce output sounds close to the original (i.e., most of the resulting products were components, not cross-components). This indicates a high correlation between brightness and sparsity (in many music examples, darker sounds are slower and more sustained than impulsive sounds, which naturally tend to be brighter). Thus, more different output sounds were found by inverting one of the rankings (forcing brighter spectra multiply slower activations), which results in a sort of spectral inversion.

The difference between direct and inverse ranking operations is illustrated by the Kraftwerk excerpt on the webpage. The first example (Sound 3) corresponds to direct ranking: increasing brightness opposed to increasing sparsity. The output sound is, in character, quite close to the original (the drums and low bass notes are kept), but with several harmonic and slight timbral variations in the background. The second example (Sound 4) corresponds to inverse ranking. In it, some low sounds have been transferred to the upper registers and the drum set has been greatly altered.

### 3.3 Constrained scramble

One of the problems of the fully-random scramble operation described above is that some noisy components, initially present with very low energy in the original sound, or corresponding to short impulses such as consonants or drum hits, might get greatly amplified if they happen to get multiplied by a high-energy or highly sustained activation function. In some cases, this can produce unpleasant sounds with a high level of noise. For instance, in the Brahms example discussed, there is a prominent oscillating layer of noise on top many of the notes of the output.

To avoid this, and to introduce another way of controlling cross-component output, the scramble operation can be constrained to be performed only on a subset of spectrum/activation pairs, leaving the rest coupled. For addressing the residual noise issue, it is highly effective to leave a percentage of the high-centroid spectra, as measured by Eq. 5, out of the random permutation (see Fig. 2(d)). In the new Brahms example re-processed by this kind of constrained scramble (Sound 5), the high-centroid components, mostly corresponding to the consonants, are left untouched, and the choral timbre is better preserved.

## 4. CROSS-SYNTHESIS

In cross-synthesis, two sounds (a source and a target) are individually subjected to NMF analysis, and the resulting spectra and activations from source and target can be combined in a wide range of different ways. Direct, random-order multiplication is possible, but is more likely to produce unsatisfactory results than single-sound scrambling, due to the even higher unpredictability of the results. Instead, user control and predictability demands a criterion-based selection of the cross-product pairings. Since no numerical ranking is needed (such as in Sect. 3.2), the use of better-performing multidimensional features is possible, and consequently, similarity amounts to proximity in a feature space.

Cross-synthesis is implemented in the system in one of two possible ways: one based on spectral similarity, and one based on temporal similarity.

### 4.1 Cross-synthesis based on spectral similarity

Here, the source and target spectra are first mapped[4] according to a measure of spectral similarity. Then, source spectra are replaced by the most similar spectra from the target, and multiplied by the source activations. This is indicated by the red arrows on Fig. 3.

This approach was originally presented in [6]. The features used for the computation of spectral similarity are the widely-used Mel Frequency Cepstral Coefficients (MFCC), which can be interpreted as a compact description of the spectral envelope. This type of cross-synthesis should be used when the emphasis is on keeping temporal structure of the source sound. Many processing details and options for controlling the matching are described in the cited paper. On the sound example webpage, some new cross-synthesis examples are included (Sounds 6-8).

### 4.2 Cross-synthesis based on temporal similarity

The current implementation of the system adds the dual process: the mapping can now also be performed in the space of activations, according to a measure of temporal similarity. Then, source activations are replaced by the most similar activations from the target, and multiplied by the source spectra (blue arrows on Fig. 3).

The emphasis now is on keeping the timbre of the source sound, adapting the temporal structure. Temporal similarity is based on the computation of a two-dimensional vector consisting of a sparsity measure as given by Eq. 6, and of a Dynamic Time Warping (DTW) minimum-cost alignment value, which can be understood as a measure of curve shape similarity independent of curve length. Feature vectors in this space (called here *shape space*) are compared by means of the Mahalanobis distance.

### 4.3 Constrained cross-synthesis

The issue of artificially high noise cross-components, which was discussed in Sect. 3.3 in the context of single-sound

[4] Here, *mapping* refers to the computation of a similarity matrix between all two possible feature vectors, followed by the assignment of each source feature vector to its closest target feature vector.

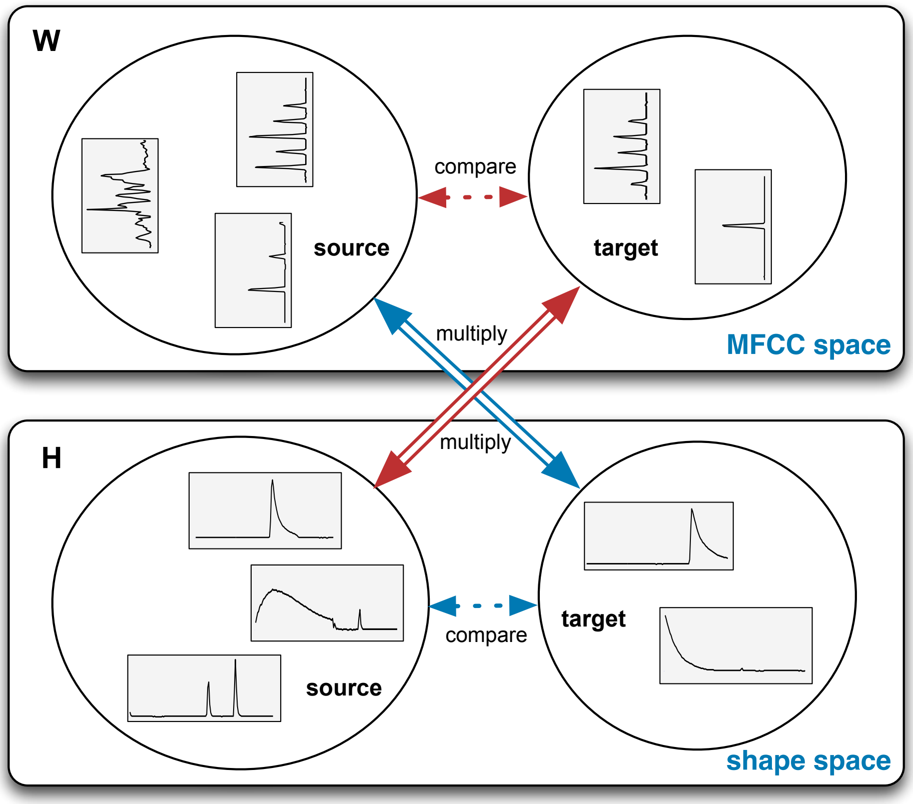


**Figure 3**. Conceptual overview of cross-synthesis operations. Red arrows denote cross-synthesis based on spectral similarity, blue arrows denote cross-synthesis based on temporal similarity.

processing, is also present here. Therefore, the possibility to use numerical ranking for discarding some spectra (or activations) from the processing has been added. For instance, in cross-synthesis based on spectral similarity, target spectra having high centroid values can be flagged to be ignored during the mapping stage. An example on the webpage (Sound 9) compares cross-synthesis with and without such a high-centroid discard condition.

## 5. WIENER-BASED RESYNTHESIS

Once the magnitude spectrogram of the output sound has been obtained by adding the generated components or cross-components, it has to be resynthesized back to the audio domain. NMF processing does not take into account phase information. Therefore, for resynthesis, phase has to be either directly taken from the input complex STFT (usually via Wiener time-frequency masking) or estimated from the magnitude spectrogram.

The latter approach was used in the work previous to this article [6], based on the Griffin and Lim phase estimation algorithm. Therein, it was argued that time-frequency masking was not appropriate for a non-subtractive task where artificial cross-components are present. Indeed, Wiener filtering works by generating a mask matrix $\mathbf{M}$ that filters out the undesired sounds from the input sound. For example, the STFT of a particular component $k$ can be obtained by

$$\mathbf{S}_k = \mathbf{M} \circ \mathbf{S} = \frac{\mathbf{w}_k \otimes \mathbf{h}_k}{\mathbf{WH}} \circ \mathbf{S}, \tag{7}$$

where $\mathbf{S}$ is the input STFT (complex), "$\circ$" denotes element-wise multiplication, and the division is also element-wise. Since

$$\mathbf{WH} = \sum_{k=1}^{K} \mathbf{w}_k \otimes \mathbf{h}_k, \tag{8}$$

the elements of the Wiener mask $\mathbf{M}$ are guaranteed to be between 0 and 1, and thus the mask acts as a filter. In the present case however, the artificial layers created by the cross-components $\mathbf{w}_i \otimes \mathbf{h}_j$ do not add to the approximation matrix $\mathbf{WH}$, and so the "mask" is no longer bounded and can take values significantly higher than one. It will filter out some time-frequency points but at the same time it will enhance others. The final processed STFT is thus given now by

$$\mathbf{S}_{out} = \frac{\sum_{i,j} \mathbf{w}_i \otimes \mathbf{h}_j}{\mathbf{WH}} \circ \mathbf{S}. \tag{9}$$

However, even if this is not the usual implementation and interpretation of Wiener filtering, it was found that the quality of the resynthesis produced by Eq. 9 was significantly higher (and more computationally efficient) than the one obtained by Griffin and Lim estimation, mostly due to the better definition of transients. A sound comparison is available on the website (Sounds 10-11). Finally, the last step is to invert $\mathbf{S}_{out}$ back to the time domain via standard overlap-add.

## 6. IMPLEMENTATION

The current analysis/synthesis framework exists as a software tool called *Factorsynth*, which is currently available in two different implementations: as a command-line tool and as a Max/MSP external object called *factorsynth∼*. Both are available online for download [5]. In the current version, both command-line tool and Max/MSP object use offline processing and are not real-time capable.

Concerning computational requirements, the current implementation completes the processing in roughly 60% to 80% of the total length of the input sounds (measured on a 2.3 GHz Intel Core i7 CPU with 4 GB of RAM). For instance, for a 20s input sound, it will complete processing in around 13s. For cross-synthesis, the sum of lengths of both input sounds has to be considered. This will vary a little depending on the operations performed and on the number of components chosen, but for most operations this was a consistent figure (it is assumed that the STFT analysis parameters are always fixed).

For future implementations, the possibility of implementing a graphical interface will be explored. Users could use a graphical representation of spectra and activations to do manual connections or selections, similar in concept to Figs. 2 and 3, or to correct cross-components created by the automatic algorithms. Or alternatively, to navigate the full matrix of cross-components.

As another improved aspect of future versions, the feasibility of a real-time implementation will be assessed. This will require the use of online factorization algorithms [9]. For accelerating cross-synthesis, a possibility would be to pre-compute and store the target spectra and activations, and perform the source factorization and source-target mapping in real time.

## 7. CONCLUSIONS

The proposed framework implements some of the new sound manipulation possibilities offered by spectrogram factorization methods. By using relatively little control data and computational requirements, it is possible to obtain a wide range of complex sounds by manipulating their internal structure. The user can control the sound complexity and overall structure, as well as how much of the timbre and temporal structure of the original sound is kept.

The focus of the current article was on the creation of cross-components, which are artificial sounds not present in the input, obtained by multiplying originally unrelated spectra and activations. Another, still unexplored possibility, would be the individual processing of separate spectra or activations. For instance, it will be possible to implement a selective time-stretching of only certain activations, or a selective pitch-shifting of only certain spectra.

Another line of improvement would be to consider more advanced factorization models. The current system handles stereo or multichannel signals by processing each channel independently. Instead, explicit multichannel factorization models [10] can improve the extraction of the components and minimize the reconstruction error. Another path to explore is the use of factorization models based on the source/filter paradigm, such as the one proposed in [11].

[5] `http://jjburred.com/software/factorsynth`

## 8. REFERENCES

[1] J. Laroche and M. Dolson, "New phase vocoder technique for pitch-shifting, harmonizing and other exotic effects," in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, New Paltz, USA, 1999.

[2] J. Moorer, "The use of linear prediction of speech in computer music applications," *Journal of the Audio Engineering Society*, vol. 27, no. 3, pp. 134–140, 1979.

[3] N. Collins and B. Sturm, "Sound cross-synthesis and morphing using dictionary-based methods," in *Proc. ICMC*, Huddersfield, UK, 2011.

[4] S. Topel and M. Casey, "Elementary sources: Latent component analysis for music composition," in *Proc. ISMIR*, Miami, USA, 2011.

[5] R. Sarver and A. Klapuri, "Application of non-negative matrix factorization to signal-adaptive audio effects." in *Proc. DAFX*, Paris, France, 2011.

[6] J. J. Burred, "Cross-synthesis based on spectrogram factorization," in *Proc. ICMC*, Perth, Australia, 2013.

[7] D. Lee and H. Seung, "Algorithms for non-negative matrix factorization," in *Neural Information Processing Systems*, Denver, USA, 2001.

[8] C. Févotte, N. Bertin, and J.-L. Durrieu, "Nonnegative Matrix Factorization with the Itakura-Saito Divergence. With Application to Music Analysis," *Neural Computation*, vol. 21, pp. 793–830, 2009.

[9] A. Lefévre, F. Bach, and C. Févotte, "Online algorithms for nonnegative matrix factorization with the Itakura-Saito divergence," in *Proc. WASPAA*, New Paltz, USA, 2011.

[10] H. Sawada, H. Kameoka, S. Araki, and N. Ueda, "Multichannel extensions of non-negative matrix factorization with complex-valued data," *IEEE Trans. on Audio, Speech and Language Processing*, vol. 21, no. 5, pp. 971–982, 2013.

[11] J.-L. Durrieu, G. Richard, B. David, and C. Févotte, "Source/filter model for unsupervised main melody extraction from polyphonic audio signals," *IEEE Trans. on Audio, Speech and Language Processing*, vol. 18, no. 3, pp. 564–575, 2010.