NMF-based method for drum separation

Dmitry Ulyanov[⋆]

Moscow State University, Moscow, Russia
dmitry.ulyanov.msu@gmail.com

Abstract. This paper addresses the problem of separating drums from polyphonic music. The method is based on Non-Negative Matrix Factorization (NMF) and coding ideas. At first basis vectors for each class are being built by separating the input spectrogram into components having a fixed spectrum and time-varying gain. Then the test sample is decomposed, using heuristically adopted NMF method. Performance evaluation has been carried out using mixtures generated from real-world polyphonic music. On some samples the results proved to be comparable to the best existing algorithms. Results are available at https://sites.google.com/site/drumseparation/

1 Introduction

The human auditory system possesses amazing abilities. More and more attempts to model this system appear with increasing computational power. One can split a sound into many sources easily, while it is still impossible for a computer to separate two simultaneously speaking people from a mono audio track.

Drum separation is a worthwhile research topic for Music Audio Retrieval nowadays. Splitting an input audio track into drum and harmonic parts is an important procedure because different methods are used to process these parts. With a quality separation algorithm it would be easier for musicians to make a backing track for their rehearsals.

Several approaches have been proposed to deal with the problem of drum extraction. The first group of algorithms is based on Blind Source Separation (BSS) methods such as Non-Negative Matrix Factorization (NMF) [4] and Independent Subspace Analysis (ISA) [8]. Finding an optimal component number is still an unsolved problem. Also it is unclear how to build the best features for determining the class of the component. Another way is called the "match and adapt". These methods start with a template of a drum sound (temporal [11] or spectral [10]), search for similar patterns and refine the template. The last approach can be considered as rule based. These methods use discriminative model between harmonic and drums sounds, which is usually based on some researchers prior knowledge about the difference between the two. [6] uses the fact, that the tonal components are presented on a spectrogram by horizontal lines, while the drum sounds are presented by vertical lines. The idea of sinusoidal modeling [3]

[⋆] Science advisor: Vladimir Chuchupal.

is to extract the most significant sinusoids from the signal supposing they are harmonics. Unfortunately, no large-scale comparison of existing methods has been made.

The algorithm introduced in this paper combines the first approach with the second one.This method was motivated by the curiosity, how well a computer can extract information from training set without any human work like feature selection.

1.1 Non-Negative Matrix Factorization

Let's consider the model, where we assume, that each signal can be represented as a sum of components, each corresponds either to drums, or to tonal instruments. It has been noticed that building the spectrogram as a sum of components, classifying these components as "drums" or "harmonic", applying Inverse Short Time Transform (ISTFT) to spectrogram, composed only from drum sources and original phases, can produce a perceptually good quality. Moreover, it is a lot easier than decomposing signal in time domain. The Non-Negative Matrix Factorization method [5] for drum separation implies that every column of spectrogram $\boldsymbol{X} \in \mathbb{R}^{T \times N}$ can be presented as a sum of source spectras $\boldsymbol{S} \in \mathbb{R}^{T \times P}$ (component per column) with time varying gains $\boldsymbol{A} \in \mathbb{R}^{P \times N}$.

$$\boldsymbol{X} \approx \boldsymbol{S}\boldsymbol{A} \tag{1}$$

The parameter P indicates the number of basis vectors. [1] suggests a method to determine the optimal value of the parameter, but usually it is chosen so that $P << N$. Both matrixes $\boldsymbol{S}$ and $\boldsymbol{A}$ should have non-negative elements to make sense as spectra and gain matrixes. To find $\boldsymbol{S}$ and $\boldsymbol{A}$ it is convenient to minimize the Frobenius norm:

$$L(\boldsymbol{X}; \boldsymbol{S}, \boldsymbol{A}) = \frac{1}{2} \|\boldsymbol{X} - \boldsymbol{S}\boldsymbol{A}\|_F = \frac{1}{2} \sum_{i}^{T} \sum_{j}^{N} \left(X_{ij} - [SA]_{ij} \right)^2 \tag{2}$$

Combined with Support Vector Machine, NMF has been successfully used for drum extraction in [4].

2 Model and algorithm

2.1 Learning

Let's say we have a training set consisting out of L objects $\{\boldsymbol{X}_i, Y_i\}_{i=1}^{L}$, $Y = \{y^d, y^t\}$, $L = L^d + L^t$, where L^d and L^t is a number of objects, which correspond to classes $y^d = \{\text{drums}\}$, $y^t = \{\text{pitched}\}$ representatively. As an object we consider a spectrogram, obtained by applying STFT to a signal, $\boldsymbol{X} \in \mathbb{R}^{T \times N_X}$, for convenience with NMF designations we use T as the number of frequency bins and N as a spectrogram length.

$$\boldsymbol{X} \approx \boldsymbol{S}^d \boldsymbol{A}^d + \boldsymbol{S}^t \boldsymbol{A}^t \tag{3}$$

where $\boldsymbol{S}_d \in \mathbb{R}^{T \times P_d}$, $\boldsymbol{S}_t \in \mathbb{R}^{T \times P_t}$ are matrices of sources spectrum , corresponding to drums and tonal components representatively. We use the Frobenius norm as a loss function:

$$L(\boldsymbol{X}; \boldsymbol{S}^d, \boldsymbol{A}^d, \boldsymbol{S}^t, \boldsymbol{A}^t) = \frac{1}{2} \left\| \boldsymbol{X} - \boldsymbol{S}^d \boldsymbol{A}^d - \boldsymbol{S}^t \boldsymbol{A}^t \right\|_F \tag{4}$$

We seek to minimize average losses based on the training set, setting $\boldsymbol{A}_i^t = \boldsymbol{0}$ for each object with the label $y_i = y^d$ and, similarly, $\boldsymbol{A}_i^d = \boldsymbol{0}$ for each object with the label $y_i = y^t$. We assume $\boldsymbol{S}^t$, $\boldsymbol{S}^d$ are common for all objects of the same class. Let's denote the parameters as vector$\boldsymbol{\Pi} = \{\boldsymbol{S}^t, \boldsymbol{S}^d, \{\boldsymbol{A}_i^d\}_{i=1}^{L^d}, \{\boldsymbol{A}_i^t\}_{i=1}^{L^t}\}$ assuming elementwise non-negativity for each of them and minimize over this parameters vector.

$$\begin{aligned} Q(\boldsymbol{X}; \boldsymbol{\Pi}) &= \frac{1}{2} \sum_{i=1}^{L} \frac{1}{T_i} \left\| \boldsymbol{X}_i - \boldsymbol{S}^d \boldsymbol{A}_i^d - \boldsymbol{S}^t \boldsymbol{A}_i^t \right\|_F = \\ &= \frac{1}{2} \Big(\sum_{i=1}^{L^d} \frac{1}{T_i^d} \left\| \boldsymbol{X}_i^d - \boldsymbol{S}^d \boldsymbol{A}_i^d \right\|_F + \sum_{i=1}^{L^t} \frac{1}{T_i^t} \left\| \boldsymbol{X}_i^t - \boldsymbol{S}^t \boldsymbol{A}_i^t \right\|_F \Big) \to \min_{\boldsymbol{\Pi}} \end{aligned} \tag{5}$$

We can solve this analytically, but only without considering non-negativity constraints, so we use an iterative procedure. The update formulas for every parameter can be obtained by writing down partial derevatives and setting it to zero, then we force non-negativity by using slice function $[a]_+ = max(a, 0)$ (elementwise for matrices). See [2] for details on derivation for standard NMF.

$$\begin{aligned} \boldsymbol{A}_i^d &\leftarrow \left[\left((\boldsymbol{S}^d)^T \boldsymbol{S}^d \right)^{-1} (\boldsymbol{S}^d)^T \boldsymbol{X}_i^d \right]_+ \\ \boldsymbol{A}_i^t &\leftarrow \left[\left((\boldsymbol{S}^t)^T \boldsymbol{S}^t \right)^{-1} (\boldsymbol{S}^t)^T \boldsymbol{X}_i^t \right]_+ \\ \boldsymbol{S}^t &\leftarrow \Big[\left(\sum_{i=1}^{L^t} \frac{1}{T_i^t} \boldsymbol{X}_i^t (\boldsymbol{A}_i^t)^T \right) \left(\sum_{i=1}^{L^t} \frac{1}{T_i^t} \boldsymbol{A}_i^t (\boldsymbol{A}_i^t)^T \right)^{-1} \Big]_+ \\ \boldsymbol{S}^d &\leftarrow \Big[\left(\sum_{i=1}^{L^d} \frac{1}{T_i^d} \boldsymbol{X}_i^d (\boldsymbol{A}_i^d)^T \right) \left(\sum_{i=1}^{L^d} \frac{1}{T_i^d} \boldsymbol{A}_i^d (\boldsymbol{A}_i^d)^T \right)^{-1} \Big]_+ \end{aligned} \tag{6}$$

Since it is very expensive to compute error, iterative procedure is stopped when iteration number limit hit.

2.2 Decomposing

The parameters that we were interested in at the training step are $\boldsymbol{S}^t$, $\boldsymbol{S}^d$. We use them now, during the decomposing step. The input object $\boldsymbol{X}$ is segmented into $\boldsymbol{X}^t = \boldsymbol{S}^t \boldsymbol{A}^t$, $\boldsymbol{X}^d = \boldsymbol{S}^d \boldsymbol{A}^d$ so that $\boldsymbol{X} \approx \boldsymbol{X}^d + \boldsymbol{X}^t$. For a fixed $\boldsymbol{S}^t$, $\boldsymbol{S}^d$ we solve an optimization problem, keeping in mind non-negativity constraints:

$$L(\boldsymbol{X}) = \frac{1}{2} \left\| \boldsymbol{X} - \boldsymbol{S}^d \boldsymbol{A}^d - \boldsymbol{S}^t \boldsymbol{A}^t \right\|_F \to \min_{\boldsymbol{A}^t, \boldsymbol{A}^d} \tag{7}$$

By analogy, setting gradient to zero, we get the same update rules:

$$\begin{aligned} \boldsymbol{A}^d &\leftarrow \left[\left((\boldsymbol{S}^d)^T \boldsymbol{S}^d \right)^{-1} (\boldsymbol{S}^d)^T \boldsymbol{X}^d \right]_+ \\ \boldsymbol{A}^t &\leftarrow \left[\left((\boldsymbol{S}^t)^T \boldsymbol{S}^t \right)^{-1} (\boldsymbol{S}^t)^T \boldsymbol{X}^t \right]_+ \end{aligned} \tag{8}$$

In the next section we discuss that succession of updates inside the iteration matters a lot.

3 Performance evaluation

3.1 Training set

Since there's no available dataset for testing drum separation algorithms, shared by authors of state-of-the-art methods, a new dataset has been created [1]. The dataset contains about 350 samples where only drums are present, 350 samples of drumless instrumental music and 350 samples containing vocals along with other not percussive instruments. Each sample is 15 seconds long.

3.2 Performance metric

We use signal-to-noise ratio (SNR) to evaluate the performance of the system described.

$$SNR = 10 \log_{10} \frac{\sum_t s(t)^2}{\sum_t (s(t) - \hat{s}(t))^2} \tag{9}$$

where $s(t)$ is the original drum signal and $\hat{s}(t)$ is the algorithm output.

3.3 Testing methods

The common parameters of an algorithm were selected to be similar to the parameters from other works on this topic ([4]). In our experiments we use STFT from the "Signal"[2] library with 40ms window multiplied by square root of Hann function with 50% overlap between consecutive frames. We use $P_d = 10$, $P_t = 30$. Other sets of P were tested, but these turned out to be fine. The number of iterations at the training step was set to 100. At decomposition step the optimal number of iterations depended heavily on the implementation way. Three ways were tested:

While evaluating it turned out that performance depends significantly on the succession of updated parameters. It has not been proven that the procedures converge, although in our experiments some kind of convergence presents in each variant.

[1] You can get it for evaluation purposes by contacting author.

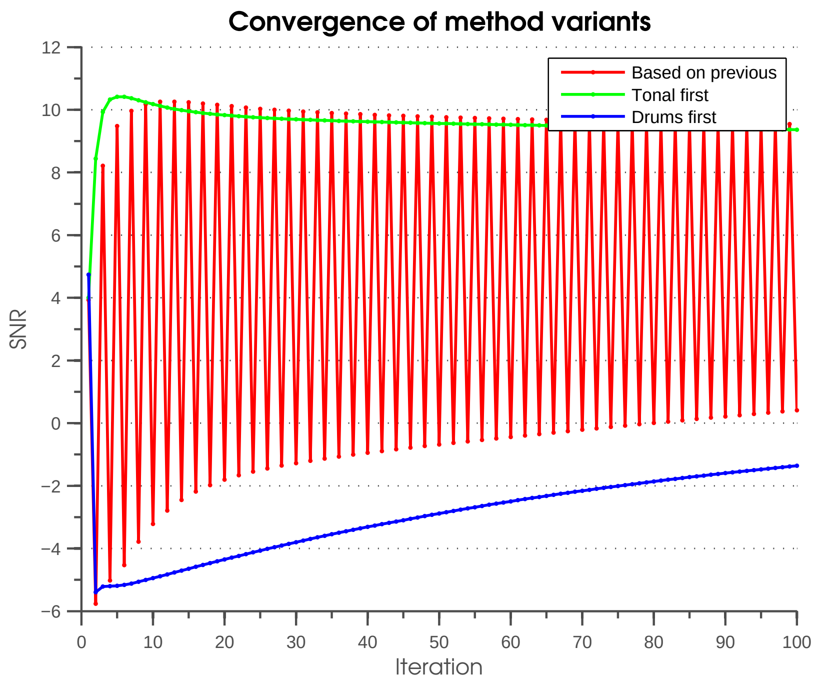
[2] `http://perso.telecom-paristech.fr/~liutkus/`

Algorithm 1 Drums first

```
for $i = 1, \dots, itNum$ do
    $A_i^d := Update(A_{i-1}^t)$
    $A_i^t := Update(A_i^d)$
end for
```

Algorithm 2 Tonal first

```
for $i = 1, \dots, itNum$ do
    $A_i^t := Update(A_{i-1}^d)$
    $A_i^d := Update(A_i^t)$
end for
```



Algorithm 3 Based on previous

```
for $i = 1, \dots, itNum$ do
    $A_i^t := Update(A_{i-1}^d)$
    $A_i^d := Update(A_{i-1}^t)$
end for
```

Taking convergence into account, we used "Tonal first" in all our evaluations as most stable. After decomposition we take an ISTFT of $\boldsymbol{X}^d = \boldsymbol{S}^d \boldsymbol{A}^d$ with the phases from original signal. Depending on the testing sample, performance varied from $SNR = -4db$ to $SNR = +12db$ [3]. The method performs well for samples with one or two tonal instruments playing along with drums. Performance gets worse when there is a bass mixed in the sample, so the bass is usually extracted with the drums. This is because in a 40ms window bass and kick drum spectras are almost the same. Although the quality of output tracks is sometimes low in terms of SNR, the algorithm can be used as a preprocessing step to tonal instrument separation algorithms, since the output never contains voice or lead instruments.

4 Future work

It should be mentioned that there is some kind of inconsistency between SNR and Frobenius norm which the algorithm minimizes, so in practice having lower Frobenius norm does not guarantees higher SNR. Besides, higher SNR does not guarantees better perceptual quality, so exploring other metrics would be an important research.

Another problem, as said above is the locality of the algorithm, so that algorithm: it cannot distinguish bass from kick drum. One of the extensions of NMF called "Non-Negative Matrix Factor Deconvolution" [1] should deal with this problem.

At first it was planned to use some regularization summand in the loss function to make classes training not independent of each other. Several regularizators were tested, but non of them gave a better result, than without regularization. It would also be interesting to try other regularizators.

5 Conclusion

In this paper, the method for drum extraction has been presented. This method uses an iterative NMF-like procedure to learn basis spectra for drum and tonal sounds and a decomposition procedure to separate the drums from other instruments in the track.

References

1. Ron Weiss Juan Bello. Identifying repeated patterns in music using sparse convolutive non-negative matrix factorization. 2010.
2. Pierre Comon and Christian Jutten. *Handbook of Blind Source Separation: Independent Component Analysis and Applications.* Academic Press, 1st edition, 2010.
3. Olivier Gillet and Gaël Richard. Extraction and remixing of drum tracks from polyphonic music signals. In *Applications of Signal Processing to Audio and Acoustics, 2005. IEEE Workshop on*, pages 315–318. IEEE, 2005.

[3] You can find samples here: `https://sites.google.com/site/drumseparation/`

4. Marko Helen and Tuomas Virtanen. Separation of drums from polyphonic music using non-negative matrix factorization and support vector machine. In *In: Proc. EUSIPCO2005. (2005*, 2005.
5. Daniel D. Lee and H. Sebastian Seung. Algorithms for non-negative matrix factorization. In *In NIPS*, pages 556–562. MIT Press, 2000.
6. Nobutaka Ono, Kenichi Miyamoto, Jonathan Le Roux, Hirokazu Kameoka, and Shigeki Sagayama. Separation of a monaural audio signal into harmonic/percussive components by complementary diffusion on spectrogram. In *Proc. EUSIPCO*, 2008.
7. François Rigaud, Mathieu Lagrange, A Robel, and Geoffroy Peeters. Drum extraction from polyphonic music based on a spectro-temporal model of percussive sounds. In *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*, pages 381–384. IEEE, 2011.
8. Christian Uhle, Christian Dittmar, and Thomas Sporer. Extraction of drum tracks from polyphonic music using independent subspace analysis. In *Proc. ICA*, pages 843–847. Citeseer, 2003.
9. Emmanuel Vincent and Xavier Rodet. Music transcription with isa and hmm. In CarlosG. Puntonet and Alberto Prieto, editors, *Independent Component Analysis and Blind Signal Separation*, volume 3195 of *Lecture Notes in Computer Science*, pages 1197–1204. Springer Berlin Heidelberg, 2004.
10. Kazuyoshi Yoshii, Masataka Goto, and Hiroshi G Okuno. Automatic drum sound description for real-world music using template adaptation and matching methods. In *Proceedings of the 5th International Conference on Music Information Retrieval (ISMIR2004)*, 2004.
11. Aymeric Zils, François Pachet, Olivier Delerue, and Fabien Gouyon. Automatic extraction of drum tracks from polyphonic music signals. In *Web Delivering of Music, 2002. WEDELMUSIC 2002. Proceedings. Second International Conference on*, pages 179–183. IEEE, 2002.