# NMF-BASED BLIND SOURCE SEPARATION USING A LINEAR PREDICTIVE CODING ERROR CLUSTERING CRITERION

*Xin Guo[1], Stefan Uhlich[2] and Yuki Mitsufuji[3]*

[1] École Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland
[2] Sony European Technology Center (EuTEC), Stuttgart, Germany
[3] Sony Corporation, Audio Technology Development Department, Tokyo, Japan

**ABSTRACT**

Non-negative matrix factorization (NMF) based sound source separation involves two phases: First, the signal spectrum is decomposed into components which, in a second step, are clustered in order to obtain estimates of the source signal spectra. The major challenge with this approach is the accuracy of the clustering algorithm in the second step, especially as most previously used clustering algorithms are focusing on the frequency part of NMF only and, hence, are missing the information of the time activation matrix. In this paper, we propose a novel clustering criterion which combines the frequency and time activation part of NMF. It is based on the linear predictive coding compression error and we show that it allows a good clustering of the NMF components while at the same time can be efficiently computed. Our new clustering criterion shows an overall improved performance compared with the current state-of-the-art clustering algorithms as we experiment on the TRIOS dataset.

***Index Terms***— Non-negative matrix factorization (NMF), Linear predictive coding (LPC), Blind source separation (BSS)

## 1. INTRODUCTION

Instantaneous, single-channel *blind source separation* (BSS) deals with the problem of obtaining $M$ estimates $\hat{s}_i(n), i = 1, \ldots, M$ of the $M$ source signals $s_i(n)$ if only a linear mixture $x(n) = \sum_{i=1}^{M} s_i(n)$ of them is given [1, 2]. One application for BSS is the separation of music into the individual instrument tracks such that an upmixing of the original content is possible [3, 4].

In this paper, we will use single-channel BSS using *non-negative matrix factorization* (NMF) as shown in Fig. 1. Such an approach usually consists of the following steps [5–7]: After the transformation of the mixture $x(n)$ into the time-frequency domain using a *short-time Fourier transform* (STFT), we apply NMF to its magnitude spectrum in order to obtain $L$ frequency basis vectors $\{\mathbf{w}_1, \ldots, \mathbf{w}_L\}$ and $L$ corresponding activation vectors $\{\mathbf{h}_1, \ldots, \mathbf{h}_L\}$ which we can summarize in the frequency matrix $\mathbf{W} \in \mathbb{R}_+^{F \times L}$ and activation matrix $\mathbf{H} \in \mathbb{R}_+^{L \times K}$ where $F$ denotes the number of frequency bins and $K$ the number of time frames. We will refer to the pair $\{\mathbf{w}_l, \mathbf{h}_l\}$ as $l$th NMF component. Using a suitable clustering, the $L > M$ components are grouped into $M$ clusters such that we can compute Wiener filter softmasks and use an inverse STFT to obtain the estimates $\hat{s}_i(n), i = 1, \ldots, M$.

Of these steps, the clustering is the most critical one as we only obtain good source separation results, if we can find a good grouping of the components $\{\mathbf{w}_l, \mathbf{h}_l\}$. Traditionally, the clustering is either manually done or the original sources are used as reference, for instance in [6, 8]. In [7], Spiertz and Gnann proposed a source-filter based method for blind clustering of $\mathbf{W}$ with either $k$means or a second NMF, where $\mathbf{W}$ is additionally transformed into *Mel-frequency cepstral coefficients* (MFCC). Their results showed significant improvement compared with previous unsupervised algorithms. Jaiswal et al. in [9] applied a constant Q transform on $\mathbf{W}$ and used shifted-NMF to separate harmonic instruments from the mixture. This method is based on western music theory and he could demonstrate considerably better quality than [7]. In [10], some efforts have been made to avoid the clustering step by introducing group sparsity into the NMF separation procedure. However, this method only works well if the overlapping of sources is less than roughly $66\%$, which is often not true for realistic applications. Another approach is [11], which combines information from the frequency basis and time activation matrices assuming that sources are monophonic. They showed a good performance on simple test files but the time disjointness assumption doesn't usually hold for real world cases, i.e., for polyphonic sources, and hence this approach fails for such mixtures.

Therefore, we propose in this paper a new clustering criterion which has the advantage that it not only takes into account the frequency basis vectors $\mathbf{w}_l$, but also the learned time activation patterns $\mathbf{h}_l$ by performing the clustering in the time domain. We will give an algorithm that minimizes the *linear predictive coding* (LPC) compression error of the source estimates as we can expect that a clustering where each estimate only contains one source can be compressed more than a clustering where estimates contain interference from other sources. Using this new clustering produces improved source separation results compared to the current state-of-the-art algorithms aforementioned.

The remainder of this paper is organized as follows: First, we review in Sec. 2 the NMF approach for single-channel BSS. Then, Sec. 3 motivates our compression error criterion and in Sec. 4 we show that we can efficiently compute it and use it for clustering. Finally, Sec. 5 gives results on the TRIOS dataset before we conclude this paper in Sec. 6.

The following notation is used throughout this paper: $\mathbf{x}$ denotes a column vector and $\mathbf{X}$ a matrix where in particular $\mathbf{I}$ is the identity matrix. The matrix transpose and Euclidean norm are denoted by $(.)^T$ and $\|.\|$, respectively. Furthermore, $\mathbf{X}.*\mathbf{Y}$ denotes the element-wise (Hadamard) product of $\mathbf{X}$ with $\mathbf{Y}$ and $\mathbf{X}^{.\wedge 2}$ the element-wise squaring of $\mathbf{X}$.

## 2. SINGLE-CHANNEL NMF BASED BSS

In the following, we will review the basic steps for single-channel NMF based source separation which are also shown in Fig. 1. Let $x(n) \in \mathbb{R}$ with $n = 1, \ldots, N$ denote a mixture signal which is composed of $M$ source signals $s_1(n), \ldots, s_M(n)$, i.e., we have

$$x(n) = \sum_{i=1}^{M} s_i(n). \tag{1}$$

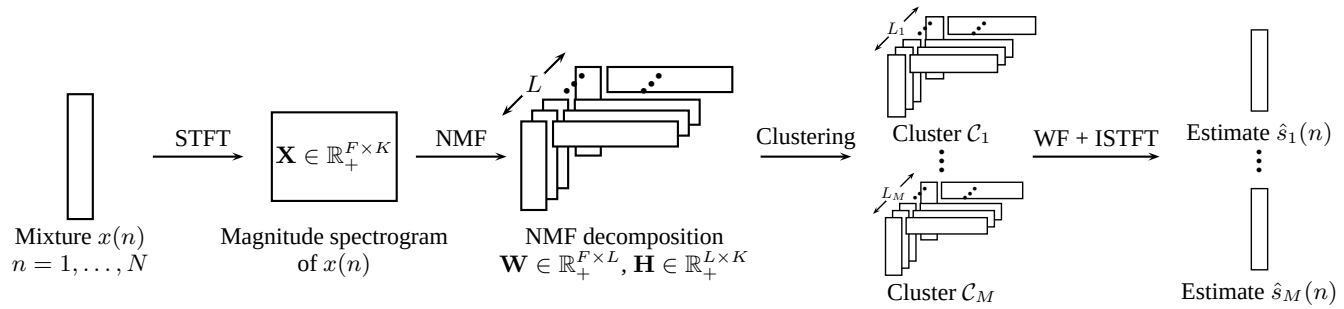


**Fig. 1**: Single-channel BSS using NMF with clustering in the frequency domain

Such signals can be observed in many fields and the task of BSS is to recover estimates $\hat{s}_1(n), \ldots, \hat{s}_M(n)$ of the source signals if only the mixture signal $x(n)$ is available. We will now give details for each step:

*(a) Time-Frequency representation:* In order to use NMF, we first compute the STFT of the mixture signal $x(n)$ and we obtain the spectrogram $S(f,k) = \text{STFT}\{x(n)\}$ where $f = 1, \ldots, F$ denotes the frequency bin index and $k = 1, \ldots, K$ the frame index. We can conveniently summarize the STFT of $x(n)$ in matrix notation as $\mathbf{S} = \mathbf{X}.*\mathbf{P} \in \mathbb{C}^{F \times K}$ where $\mathbf{X} \in \mathbb{R}_+^{F \times K}$ and $\mathbf{P} \in \mathbb{C}^{F \times K}$ denote the magnitude and phase spectrogram, respectively.

*(b) Non-negative matrix factorization:* Assuming that each source has a characteristic frequency spectrum which can be expressed by a small number of frequency basis vectors, we can use NMF to extract them. As we apply NMF to the magnitude spectrogram $\mathbf{X}$, we use the *Kullback-Leibler* (KL) divergence as cost function, see for example [12], and we obtain two non-negative matrices $\mathbf{W} \in \mathbb{R}_+^{F \times L}$ and $\mathbf{H} \in \mathbb{R}_+^{L \times K}$ with $\mathbf{X} \approx \mathbf{WH}$. Additionally, temporal continuity is enforced as proposed in [6] to obtain a NMF decomposition which is beneficial for the source separation.

*(c) Clustering:* Using a suitable clustering method, e.g., $k$means on the columns of the frequency basis matrix $\mathbf{W}$, we obtain $M$ clusters $\mathcal{C}_1, \ldots, \mathcal{C}_M$. Please see Sec. 1 for an overview of clustering methods that have been proposed so far.

*(d) Source reconstruction:* Finally, the source signals are found by using Wiener filtering to get source spectrogram estimates and applying an inverse STFT to them, i.e., we compute

$$\hat{s}_i(n) = \text{ISTFT}\left\{ \frac{\sum_{l \in \mathcal{C}_i} (\mathbf{w}_l \mathbf{h}_l^T)^{.\wedge 2}}{\sum_{l=1}^{L} (\mathbf{w}_l \mathbf{h}_l^T)^{.\wedge 2}} .* \mathbf{S} \right\} \tag{2}$$

for all $i = 1, \ldots, M$. Due to the Wiener filtering, we know that[1] $x(n) = \sum_{i=1}^{M} \hat{s}_i(n)$.

Of these steps, the clustering step *(c)* is the most important one as single-channel NMF based BSS will only work if we can find a good clustering of the NMF components. In the following, we will introduce a new clustering criterion which has the advantage that it improves the source separation results compared to the current state-of-the-art methods.

## 3. LPC ERROR CLUSTERING CRITERION

### 3.1. Definition and Motivation

The clustering criterion that we want to propose is computed in the time domain and, therefore, we first use Wiener filtering to obtain $L$ time components $c_1(n), \ldots, c_L(n)$ with

$$c_l(n) = \text{ISTFT}\left\{ \frac{\left(\mathbf{w}_l \mathbf{h}_l^T\right)^{.\wedge 2}}{\sum_{l=1}^{L} (\mathbf{w}_l \mathbf{h}_l^T)^{.\wedge 2}} .* \mathbf{S} \right\} \tag{3}$$

where $\mathbf{w}_l$, $\mathbf{h}_l$ are the $l$th column and row of $\mathbf{W}$ and $\mathbf{H}$, respectively. In order to solve the source separation problem, we now need to group these $L$ signals into $M$ clusters $\mathcal{C}_1, \ldots, \mathcal{C}_M$ such that $\hat{s}_i(n) = \sum_{l \in \mathcal{C}_i} c_l(n)$ is a good estimate of the $i$th source signal $s_i(n)$. In this paper, we propose to use the compressibility of the signals $\hat{s}_i(n)$ in order to find a good clustering as our assumption is that we can compress an estimated source signal $\hat{s}_i(n)$ better if it only contains components from one source and, hence, our task is to find the clustering $\mathcal{C}_1, \ldots, \mathcal{C}_M$ which results in source estimates $\hat{s}_1(n), \ldots, \hat{s}_M(n)$ that can be most compressed. The compression method that we want to use is *linear predictive coding* (LPC) which we will outline in the following: In order to see how much we can compress an estimated source signal $\hat{s}_i(n)$, we compute the LPC error

$$e_i(n) = \hat{s}_i(n) - \left(h_{\hat{s}_i} \star \hat{s}_i\right)(n), \tag{4}$$

where $h_{\hat{s}_i}(n)$ denotes the causal impulse response of the optimal LPC filter of length $P$ for the signal $\hat{s}_i(n)$ with $h_{\hat{s}_i}(0) = 0$ such that $e_i(n)$ has minimum energy, i.e., at time instance $n$, we can view $\left(h_{\hat{s}_i} \star \hat{s}_i\right)(n)$ as an optimal estimate of $\hat{s}_i(n)$ given the past samples $\hat{s}_i(n-1), \ldots, \hat{s}_i(n-P)$ and $e_i(n)$ is the difference between the predicted value $(h_{\hat{s}_i} \star \hat{s}_i)(n)$ and the true value $\hat{s}_i(n)$. Combining the LPC errors $e_i(n)$ for all sources, we can compute the overall error signal $e(n) = \sum_{i=1}^{M} e_i(n)$ and, in order to solve the clustering problem, we are looking for the clustering $\mathcal{C}_1, \ldots, \mathcal{C}_M$ such that the energy of $e(n)$ is minimized, i.e., we want to solve

$$\min_{\mathcal{C}_1, \ldots, \mathcal{C}_M} \sum_{n=1}^{N} e(n)^2 \tag{5}$$

with

$$\begin{aligned} e(n) &= \sum_{i=1}^{M} e_i(n) = \sum_{i=1}^{M} \left(\hat{s}_i(n) - (h_{\hat{s}_i} \star \hat{s}_i)(n)\right) \\ &= x(n) - \sum_{i=1}^{M} (h_{\hat{s}_i} \star \hat{s}_i)(n). \end{aligned} \tag{6}$$

From (6), we can see that we subtract from the mixture signal $x(n)$ all those parts that we can "explain" by LPC and, hence, finding a clustering with minimum energy $\sum_{n=1}^{N} e(n)^2$ should result in a good source separation as we will also demonstrate in Sec. 5.

[1] Note that we neglect here and in the following any artifacts that may be due to the inverse STFT reconstruction [13].

### 3.2. Computation of the optimal LPC filter

In the following, we will briefly review linear predictive coding, please refer to [14] for a more detailed discussion. In linear predictive coding, we want to predict the sample $\hat{s}_i(n)$ given the last $P$ samples of it, i.e., we want to minimize the energy of the residual $e_i(n)$ where

$$e_i(n) = \hat{s}_i(n) - (h_{\hat{s}_i} \star \hat{s}_i)(n),\ n = 1, \dots, N, \tag{7}$$

and $h_{\hat{s}_i}(n) \neq 0$ only for $1 \leq n \leq P$. For convenience, we will use the following matrix notation: Let $\mathbf{e}_i = \begin{bmatrix} e_i(1) & \cdots & e_i(N) \end{bmatrix}^T \in \mathbb{R}^N$, $\hat{\mathbf{s}}_i = \begin{bmatrix} \hat{s}_i(1) & \cdots & \hat{s}_i(N) \end{bmatrix}^T \in \mathbb{R}^N$ and $\mathbf{h}_{\hat{s}_i} = \begin{bmatrix} h_{\hat{s}_i}(1) & \cdots & h_{\hat{s}_i}(P) \end{bmatrix}^T \in \mathbb{R}^P$ denote the error, estimated source signal and LPC impulse response for the $i$th source, respectively. Then, we can write (7) as

$$\mathbf{e}_i = \hat{\mathbf{s}}_i - \hat{\mathbf{S}}_i \mathbf{h}_{\hat{s}_i} \tag{8}$$

with $\hat{\mathbf{S}}_i \in \mathbb{R}^{N \times P}$

$$\hat{\mathbf{S}}_i = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ \hat{s}_i(1) & 0 & \cdots & 0 \\ \hat{s}_i(2) & \hat{s}_i(1) & \cdots & 0 \\ \vdots & & \ddots & \\ \hat{s}_i(N-1) & \hat{s}_i(N-2) & \cdots & \hat{s}_i(N-P) \end{bmatrix} \tag{9}$$

and it is well known that the optimal prediction filter that minimizes $\|\mathbf{e}_i\|^2$ is given by (see [15])

$$\mathbf{h}_{\hat{s}_i} = (\hat{\mathbf{S}}_i^T \hat{\mathbf{S}}_i)^{-1} \hat{\mathbf{S}}_i^T \hat{\mathbf{s}}_i. \tag{10}$$

Plugging this into (8), we see that the residual $\mathbf{e}_i$ is given by

$$\mathbf{e}_i = \hat{\mathbf{s}}_i - \hat{\mathbf{S}}_i (\hat{\mathbf{S}}_i^T \hat{\mathbf{S}}_i)^{-1} \hat{\mathbf{S}}_i^T \hat{\mathbf{s}}_i \tag{11}$$

and the overall error is $\mathbf{e} = \sum_{i=1}^M \mathbf{e}_i = \mathbf{x} - \sum_{i=1}^M \hat{\mathbf{S}}_i (\hat{\mathbf{S}}_i^T \hat{\mathbf{S}}_i)^{-1} \hat{\mathbf{S}}_i^T \hat{\mathbf{s}}_i$ where $\mathbf{x} = \sum_{i=1}^M \hat{\mathbf{s}}_i$ is the vector representation of the mixture signal, i.e., $\mathbf{x} = \begin{bmatrix} x(1) & \cdots & x(N) \end{bmatrix}^T \in \mathbb{R}^N$. The overall error $\mathbf{e}$ has the energy, i.e., squared $l_2$-norm

$$\begin{aligned} \|\mathbf{e}\|^2 = \|\mathbf{x}\|^2 &- 2 \sum_{i=1}^M \mathbf{x}^T \hat{\mathbf{S}}_i (\hat{\mathbf{S}}_i^T \hat{\mathbf{S}}_i)^{-1} \hat{\mathbf{S}}_i^T \hat{\mathbf{s}}_i \\ &+ \sum_{i=1}^M \sum_{j=1}^M \hat{\mathbf{s}}_i^T \hat{\mathbf{S}}_i (\hat{\mathbf{S}}_i^T \hat{\mathbf{S}}_i)^{-1} \hat{\mathbf{S}}_i^T \hat{\mathbf{S}}_j (\hat{\mathbf{S}}_j^T \hat{\mathbf{S}}_j)^{-1} \hat{\mathbf{S}}_j^T \hat{\mathbf{s}}_j. \end{aligned} \tag{12}$$

In order to find a good source separation result, we want to cluster $c_1(n), \dots, c_L(n)$ such that $\|\mathbf{e}\|^2$ is minimized.

## 4. PROPOSED CLUSTERING APPROACH

### 4.1. Outline of the Method

We will now describe an iterative algorithm that we can use for minimizing the error energy (12). In each iteration, $R$ components are randomly chosen and they are assigned to the clusters such that the error energy (12) is smallest. Hence, the algorithm consists of the following three steps:

1.) Randomly generate an initial clustering $\mathcal{C}_1, \dots, \mathcal{C}_M$.

2.) Choose $R$ (for example $R = 2$) arbitrary components and compute for all $M^R$ possible clustering assignments the compression error. Assign the $R$ components to the clusters such that (12) is minimized.

3.) Repeat Step 2: For each epoch go through all possible subsets of size $R$ until the clustering is stable.

As this iterative approach is only performing a local optimization, i.e., we can not guarantee that we have converged to the clustering with the smallest LPC error, we have observed that we can further improve the source separation results by restarting the iterative algorithm several times with different initial clustering constellations and merging their results in an additional fourth step:

4.) Repeat Step 1.) - 3.) with different initial clusterings. Compute from the clustering results a similarity matrix $\mathbf{Q} \in \mathbb{R}^{L \times L}$ where the $(l_1, l_2)$th element gives the probability that the $l_1$th and $l_2$th component are in the same cluster and run $k$medoids on $\mathbf{Q}$ to output a final clustering.

The $k$medoids algorithm that we use here is a basic *partitioning around medoids* (PAM) algorithm, see [16].

### 4.2. Efficient computation of the LPC Error

As the LPC error (12) needs to be computed $M^R$ times in each iteration, it is important to efficiently compute it. This is possible as we can exploit the fact that all source estimates $\hat{s}_i(n)$ are computed from the $L$ components $c_l(n)$.

From (11), we see that we use the (cross-)correlation vector $\hat{\mathbf{S}}_i^T \hat{\mathbf{s}}_i$ and (auto-)correlation matrix $\hat{\mathbf{S}}_i^T \hat{\mathbf{S}}_i$ to compute the error energy $\|\mathbf{e}_i\|^2$. Computing both quantities requires either $O(NP)$ operations if it is computed directly or $O(N \log(N))$ operations if computed via the *fast Fourier transform* (FFT), which might be slow if the signal length $N$ is large. However, $\hat{\mathbf{S}}_i^T \hat{\mathbf{s}}_i$ and $\hat{\mathbf{S}}_i^T \hat{\mathbf{S}}_i$ can be efficiently computed as we know that $\hat{s}_i(n)$ can be expressed as $\hat{s}_i(n) = \sum_{l \in \mathcal{C}_i} c_l(n)$ and, hence,

$$\hat{\mathbf{S}}_i^T \hat{\mathbf{s}}_i = \sum_{l_1 \in \mathcal{C}_i} \sum_{l_2 \in \mathcal{C}_i} \mathbf{C}_{l_1}^T \mathbf{c}_{l_2}, \quad \hat{\mathbf{S}}_i^T \hat{\mathbf{S}}_i = \sum_{l_1 \in \mathcal{C}_i} \sum_{l_2 \in \mathcal{C}_i} \mathbf{C}_{l_1}^T \mathbf{C}_{l_2}, \tag{13}$$

where $\mathbf{C}_l \in \mathbb{R}^{N \times P}$ has the same Toeplitz structure as $\hat{\mathbf{S}}_i$ in (9) and is composed of $c_l(n)$ such that $\hat{\mathbf{S}}_i = \sum_{l \in \mathcal{C}_i} \mathbf{C}_l$ and $\mathbf{c}_l = \begin{bmatrix} c_l(1) & \cdots & c_l(N) \end{bmatrix}^T \in \mathbb{R}^N$ is the vector representation of the $l$th time component. If we precompute the products $\mathbf{C}_{l_1}^T \mathbf{c}_{l_2} \in \mathbb{R}^P$ and $\mathbf{C}_{l_1}^T \mathbf{C}_{l_2} \in \mathbb{R}^{P \times P}$ for all $1 \leq l_1, l_2 \leq L$ then we can quickly evaluate (13) at each iteration. Thus, the computation of $\mathbf{e}_i$ in (11) only consists of solving the linear system of equations $\hat{\mathbf{S}}_i^T \hat{\mathbf{S}}_i \mathbf{h}_{\hat{s}_i} = \hat{\mathbf{S}}_i^T \hat{\mathbf{s}}_i$ which can be done efficiently as $P$ is small, e.g., $P = 10$ in Sec. 5.

## 5. RESULTS

In the following, we will compare our proposed algorithm with the MFCC-based methods from [7] and shifted-NMF clustering from [9]. We will use the same pre-training procedure as mentioned in [17]: For every source $s_i(n)$, we run the regular KL-NMF algorithm separately which yields each time $L_i$ different frequency and activation vectors. Then, we concatenate the found frequency basis vectors for all $M$ sources such that we obtain the pre-trained frequency matrix $\mathbf{W} \in \mathbb{R}_+^{F \times L}$ with $L = L_1 + \cdots + L_M$. Finally, we run KL-NMF again on the mixture signal $x(n)$ where we keep $\mathbf{W}$ fixed and only update the activation matrix $\mathbf{H}$. This procedure has the advantage that we know the ground truth of the clustering and, hence, we can also give the performance of an oracle system that can conduct a perfect clustering. However, it is important to note that this pre-training step is only done for the sake of having an oracle clustering and the proposed compression error clustering can also be used if the NMF is learned from the mixture only.

The dataset that we use for our experiment is the same as the one that was used in [18]: it includes the whole TRIOS dataset [19] with several music mixtures of multiple harmonic instruments, the "Bach" quartet from Bach 10 dataset [20], and the "MIREX" quintet extracted from MIREX 2007 dataset which was also used in [21]. All

| Recording | Instrument | Oracle clustering | | | Proposed method | | | MFCC $k$means [7] | | | Mel NMF [7] | | | Shifted-NMF [9] | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | SDR | SIR | SAR | SDR | SIR | SAR | SDR | SIR | SAR | SDR | SIR | SAR | SDR | SIR | SAR |
| Brahms | Horn | 7.20 | 9.94 | 10.91 | **7.07** | 13.86 | 8.27 | 3.87 | 5.76 | 9.41 | 4.17 | 5.83 | 10.17 | 2.95 | 3.34 | 15.20 |
| | Piano | 5.73 | 8.48 | 9.59 | 0.64 | 1.04 | 13.63 | 3.30 | 4.42 | 10.76 | −0.10 | 0.21 | 14.39 | **3.78** | 5.59 | 9.50 |
| | Violin | 10.14 | 19.24 | 10.76 | **10.14** | 19.24 | 10.76 | −8.35 | −7.89 | 10.21 | 9.69 | 19.79 | 10.19 | 7.66 | 10.96 | 10.74 |
| Lussier | Bassoon | 8.08 | 11.76 | 10.79 | **8.08** | 11.76 | 10.79 | 1.85 | 11.67 | 2.61 | 0.15 | 0.75 | 11.72 | −0.83 | −0.60 | 15.43 |
| | Piano | 7.15 | 10.93 | 9.83 | **7.15** | 10.93 | 9.83 | 4.66 | 6.28 | 10.64 | 4.56 | 8.00 | 7.83 | 2.54 | 5.14 | 7.16 |
| | Trumpet | 10.10 | 16.87 | 11.22 | **10.10** | 16.87 | 11.22 | −1.73 | −1.29 | 12.18 | 8.46 | 18.12 | 9.05 | 6.57 | 7.39 | 14.95 |
| Mozart | Clarinet | 14.44 | 22.99 | 15.91 | **14.44** | 22.99 | 15.91 | 13.82 | 23.37 | 14.35 | 13.82 | 23.37 | 14.35 | 10.62 | 11.78 | 17.18 |
| | Piano | 9.02 | 5.85 | 11.37 | **9.02** | 5.85 | 11.37 | −3.80 | −3.51 | 13.21 | 5.12 | 6.58 | 11.43 | 4.40 | 5.25 | 13.03 |
| | Viola | 2.62 | 14.44 | 5.31 | **2.62** | 14.44 | 5.31 | −4.83 | −3.80 | 7.28 | −8.25 | −7.92 | 11.66 | 2.41 | 6.55 | 5.39 |
| Schubert | Cello | 7.01 | 16.26 | 7.66 | **7.01** | 16.26 | 7.66 | −6.48 | −6.04 | 10.70 | −7.57 | −7.08 | 9.97 | −0.09 | 0.41 | 12.34 |
| | Piano | 10.64 | 14.67 | 12.98 | **10.64** | 14.67 | 12.98 | 8.85 | 17.92 | 9.49 | 4.05 | 5.84 | 9.77 | 4.44 | 5.06 | 14.35 |
| | Violin | 6.38 | 10.27 | 9.05 | **6.38** | 10.27 | 9.05 | −2.34 | −1.61 | 9.60 | 0.33 | 1.64 | 8.42 | −1.39 | −0.46 | 9.02 |
| Bach | Bassoon | 5.15 | 8.31 | 8.62 | −15.95 | −15.09 | 6.73 | **3.72** | 5.96 | 8.64 | 0.63 | 1.64 | 9.73 | −0.44 | 0.35 | 10.15 |
| | Clarinet | 5.29 | 7.84 | 9.47 | −0.69 | 0.01 | 10.57 | **2.70** | 4.11 | 9.72 | 0.25 | 1.87 | 7.49 | 1.10 | 2.02 | 10.42 |
| | Saxophone | 5.81 | 11.26 | 7.58 | **5.81** | 11.26 | 7.58 | −0.55 | 0.49 | 8.93 | −1.25 | −0.48 | 9.95 | 3.16 | 4.52 | 10.17 |
| | Violin | 7.79 | 14.13 | 9.10 | **8.62** | 15.73 | 9.67 | 5.74 | 8.15 | 10.08 | 7.13 | 13.93 | 8.32 | 4.19 | 5.79 | 10.31 |
| Take Five | Kick | 12.38 | 20.91 | 13.08 | −11.53 | −5.19 | −4.04 | **12.38** | 20.91 | 13.08 | **12.38** | 20.91 | 13.08 | 11.14 | 16.14 | 12.89 |
| | Piano | 4.06 | 6.79 | 8.19 | −0.97 | −0.33 | 10.82 | **2.83** | 6.37 | 6.28 | −1.35 | −0.83 | 11.53 | 2.66 | 4.30 | 9.07 |
| | Ride | 5.67 | 12.59 | 6.89 | **5.67** | 12.59 | 6.89 | −32.39 | −30.38 | 2.31 | −0.48 | 12.56 | −0.02 | −2.63 | −1.23 | 6.64 |
| | Saxophone | 9.47 | 17.72 | 10.25 | 7.66 | 16.69 | 8.34 | 6.84 | 18.12 | 7.24 | 5.39 | 19.53 | 5.60 | **8.26** | 10.78 | 12.17 |
| | Snare | −0.08 | 9.11 | 0.98 | 1.41 | 11.82 | 2.10 | **1.82** | 11.96 | 2.54 | −1.56 | 1.70 | 3.47 | −24.21 | −23.20 | 5.89 |
| MIREX | Bassoon | 7.74 | 14.92 | 8.81 | **7.74** | 14.92 | 8.81 | 1.49 | 3.65 | 7.12 | 1.20 | 3.40 | 6.83 | 0.35 | 1.23 | 10.19 |
| | Clarinet | 6.54 | 9.43 | 10.14 | −1.35 | −0.62 | 10.12 | **1.85** | 2.95 | 10.12 | −1.04 | −0.43 | 11.01 | 1.54 | 2.19 | 12.16 |
| | Flute | 9.76 | 16.02 | 11.05 | **10.07** | 18.88 | 10.74 | 4.71 | 5.79 | 12.30 | 7.50 | 10.36 | 11.05 | 6.30 | 6.95 | 15.63 |
| | Horn | 5.61 | 10.18 | 7.87 | **1.93** | 5.50 | 5.52 | −0.79 | 2.00 | 4.58 | −0.69 | 2.15 | 4.57 | 1.68 | 2.64 | 10.61 |
| | Oboe | −6.22 | −4.43 | 4.26 | −12.97 | −12.54 | 10.05 | **−8.86** | −8.18 | 8.30 | −10.89 | −10.62 | 12.27 | −9.23 | −8.30 | 6.82 |

**Table 1**: Single-channel NMF separation results for the TRIOS dataset, all values are given in dB

music files are down-sampled to 16 kHz to speed up our method and the BSS Eval toolbox is utilized to evaluate the obtained source separation results, see [22]. Furthermore, we use the following experimental settings: The magnitude spectrogram of the mixture signal is obtained using a STFT with a 75% overlapping Hamming window and a window length of size 4096. For the pre-training, each source is separated into $L_i = 5$ components such that the overall number of components is $L = 5M$.

The obtained source separation results are summarized in Table 1 where we compare the following approaches:

- *"Oracle clustering":* The clustering is done according to the ground truth, i.e., $\mathcal{C}_i = \{1+5(i-1), \ldots, 5i\}$ as we use $L_i = 5$.
- *"Proposed method":* As introduced in Sec. 3 and 4, we use the LPC error criterion for the clustering with an LPC filter length $P = 10$ and in each iteration, we update $R = 2$ components. Overall, we restart the algorithm 50 times in order to calculate the similarity matrix $\mathbf{Q}$ and run $k$medoids on it to obtain the final clustering.
- *"MFCC $k$means [7]":* This approach uses a Mel filter bank of size 30 which is applied to the frequency basis vectors in $\mathbf{W}$ in order to compute MFCCs. These are then clustered via $k$means. For the Mel filter bank, we use the implementation [23] of [24].
- *"Mel NMF [7]":* This approach also applies a Mel filter bank of size 30 to the original frequency basis vectors and uses a second NMF to perform the clustering.
- *"Shifted-NMF [9]":* For this approach, we use a constant Q transform matrix with minimal frequency 55 Hz and 24 frequency bins per octave. The shifted-NMF decomposition is allowed to use a maximum of 24 shifts. The constant Q transform source code was kindly provided by D. FitzGerald and we use the Tensor toolbox [25,26] for the shifted-NMF implementation.

In Table 1, we give *signal-to-distortion ratio* (SDR), *signal-to-interference ratio* (SIR) and *signal-to-artifact ratio* (SAR) values for all instruments [22]. For each instrument, the separation giving highest SDR is emphasized in bold-face. From the results, we can see that our algorithm outputs the source separation with the best average SDR result on all three-source music files ("Brahms", "Lussier", "Mozart" and "Schubert") and "MIREX". For "Bach", although our method is slightly worse than the others, it separates the best saxophone and violin part from the mixture and for "Take Five" the proposed algorithm is only worse than the Mel NMF method. The reason why it fails to work fine for "Take Five" is that there are two percussive sources, i.e., "Kick" and "Snare", which are not well suited for the LPC compression error and we are therefore planning to extend our method, see Sec. 6.

The advantage of our proposed clustering method lies in the combination of frequency and time information. A good example of this advantage can be seen for "Schubert": It consists of three sources (cello, piano and violin) where the cello and the violin have similar frequency basis vectors and, hence, the other three algorithms fail to separate these two sources. However, the time activation of these two instruments is different and our method can take advantage of this information.

Finally, it is interesting to note that in some situations, the SDR of the estimated source is even better than for the oracle clustering, e.g., MFCC $k$means produces better results for the snare of "Take Five" than oracle clustering. The reason is that during the procedure of generating separated components, we update only the time activation matrix $\mathbf{H}$ while fixing the frequency basis matrix $\mathbf{W}$. If two sources have a similar frequency basis, interference between instruments can be introduced through the time activation matrix and the quality of sources with relatively small amplitude will be damaged. We could see this phenomenon for the flute of "MIREX", the snare of "Take Five" and the violin of "Bach". Thus, the local optimum clustering for each instrument might not be the oracle clustering, however, oracle clustering provides us the best clustering in general.

## 6. CONCLUSIONS AND FUTURE WORK

In this paper, we have proposed a new clustering criterion for single-channel NMF source separation. As it is computed in the time-domain, it combines information from the frequency basis vectors and their corresponding activations and we could show that it performs superior to the current state-of-the-art algorithms.

Due to the properties of linear predictive coding, our method works well for harmonic sources but may fail for percussive ones. We therefore plan to incorporate a harmonic/percussive separation in the first place in order to separate the percussive from the harmonic sources, e.g., using the method [27], before applying the NMF source separation with LPC clustering.

## 7. REFERENCES


[1] P. Comon and C. Jutten, Eds., *Handbook of Blind Source Separation: Independent Component Analysis and Applications*, Academic Press, 2010.

[2] G. R. Naik and W. Wang, Eds., *Blind Source Separation: Advances in Theory, Algorithms and Applications*, Springer, 2014.

[3] D. FitzGerald, "Upmixing from mono - a source separation approach," *Proc. 17th International Conference on Digital Signal Processing*, 2011.

[4] D. FitzGerald, "The good vibrations problem," *134th AES Convention, e-brief*, 2013.

[5] D. FitzGerald, M. Cranitch, and E. Coyle, "Shifted non-negative matrix factorization for sound source separation," *Workshop on Statistical Signal Processing*, pp. 1132–1137, 2005.

[6] T. Virtanen, "Monaural sound source separation by non-negative matrix factorization with temporal continuity and sparseness criteria," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 15, pp. 1066–1074, 2007.

[7] M. Spiertz and V. Gnann, "Source-filter based clustering for monaural blind source separation," *Proc. Int. Conference on Digital Audio Effects*, 2009.

[8] B. Wang and M. D. Plumbley, "Investigating single-channel audio source separation methods based on non-negative matrix factorization," *Proceeding of the ICA Research Network International Workshop*, pp. 17–20, 2007.

[9] R. Jaiswal, D. FitzGerald, D. Barry, E. Coyle, and S. Rickard, "Clustering NMF basis functions using shifted NMF for monaural sound source separation," *Proc. IEEE Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 245–248, 2011.

[10] A. Lefevre, F. Bach, and C. Fevotte, "Itakura-Saito non-negative matrix factorization with group sparsity," *Proc. IEEE Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 21–24, 2011.

[11] K. Murao, M. Nakano, Y. Kitano, N. Ono, and S. Sagayama, "Monophonic instrument sound segregation by clustering NMF components based on basis similarity and gain disjointness," *11th International Society for Music Information Retrieval Conference*, pp. 375–380, 2010.

[12] D. Lee and H. S. Seung, "Algorithms for non-negative matrix factorization," *Advances in neural information processing systems*, pp. 556–562, 2000.

[13] B. Yang, "A study of inverse short-time Fourier transform," *Proc. IEEE Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 3541–3544, 2008.

[14] S. Haykin, *Adaptive Filter Theory*, Prentice-Hall, 4th edition, 2002.

[15] S. M. Kay, *Fundamentals of Statistical Signal Processing, Volume 1: Estimation Theory*, Prentice-Hall, 1993.

[16] S. Theodoridis and K. Koutroumbas, *Pattern Recognition*, Academic Press, 2006.

[17] E. M. Grais and H. Erdogan, "Single channel speech music separation using nonnegative matrix factorization and spectral mask," *IEEE International Conference on Digital Signal Processing (DSP)*, pp. 1–6, 2011.

[18] J. Fritsch and M. Plumbley, "Score informed audio source separation using constrained nonnegative matrix factorization and score synthesis," *Proc. IEEE Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 888–891, 2013.

[19] J. Fritsch, "High quality musical audio source separation," *Master's Thesis, UPMC / IRCAM / Telecom ParisTech*, 2012.

[20] Z. Duan and B. Pardo, "Soundprism: An online system for score-informed source separation of music audio," *IEEE Journal of Selected Topics in Signal Processing*, vol. 5, no. 6, pp. 1205–1215, 2011.

[21] J. Fritsch, J. Ganseman, and M. D. Plumbley, "A comparison of two different methods for score-informed source separation," *5th International Workshop on Machine Learning and Music*, 2012.

[22] E. Vincent, R. Gribonval, and C. Févotte, "Performance measurement in blind audio source separation," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 14, no. 4, pp. 1462–1469, 2006.

[23] P. Brady, "Matlab Mel filter implementation," `http://www.mathworks.com/matlabcentral/fileexchange/23179-melfilter`, 2014, [Online].

[24] F. Zheng, G. Zhang, and Z. Song, "Comparison of different implementations of MFCC," *Journal of Computer Science and Technology*, vol. 16, no. 6, pp. 582–589, 2001.

[25] B. W. Bader and T. G. Kolda, "MATLAB tensor toolbox version 2.5," `http://www.sandia.gov/~tgkolda/TensorToolbox/`, 2012, [Online].

[26] B. W. Bader and T. G. Kolda, "Algorithm 862: MATLAB tensor classes for fast algorithm prototyping," *ACM Transactions on Mathematical Software*, vol. 32, no. 4, pp. 635–653, 2006.

[27] D. FitzGerald, "Harmonic/percussive separation using Median filtering," *International Conference on Digital Audio Effects*, 2010.