A Quasi-Newton Algorithm on the Orthogonal Manifold for NMF with Transform Learning

Pierre Ablin, Dylan Fagot, Herwig Wendt, Alexandre Gramfort, Cédric Févotte

A QUASI-NEWTON ALGORITHM ON THE ORTHOGONAL MANIFOLD FOR NMF WITH TRANSFORM LEARNING

Pierre Ablin[†]*, Dylan Fagot*[‡]*, Herwig Wendt*[‡]*, Alexandre Gramfort*[†] *and Cédric Févotte*[‡]

[†] Inria, Parietal team, Université Paris-Saclay, Saclay, France
[‡] IRIT, Université de Toulouse, CNRS, Toulouse, France

ABSTRACT

Nonnegative matrix factorization (NMF) is a popular method for audio spectral unmixing. While NMF is traditionally applied to off-the-shelf time-frequency representations based on the short-time Fourier or Cosine transforms, the ability to learn transforms from raw data attracts increasing attention. However, this adds an important computational overhead. When assumed orthogonal (like the Fourier or Cosine transforms), learning the transform yields a non-convex optimization problem on the orthogonal matrix manifold. In this paper, we derive a quasi-Newton method on the manifold using sparse approximations of the Hessian. Experiments on synthetic and real audio data show that the proposed algorithm outperforms state-of-the-art first-order and coordinate-descent methods by orders of magnitude in terms of speed. A Python package for fast TL-NMF is released online at `https://github.com/pierreablin/tlnmf`.

***Index Terms*—** Nonnegative matrix factorization (NMF), transform learning, source separation, non-convex optimization, manifolds, audio signal processing.

1. INTRODUCTION

Nonnegative matrix factorization (NMF) consists in decomposing a nonnegative data matrix $\mathbf{V} \in \mathbb{R}_+^{M\times N}$ into [1]:

$$\mathbf{V} \approx \mathbf{WH} \tag{1}$$

where $\mathbf{W} \in \mathbb{R}_+^{M\times K}$ and $\mathbf{H} \in \mathbb{R}_+^{K\times N}$ are two nonnegative matrices referred to as *dictionary* and *activation* matrix, respectively. The rank K of the factorization is generally chosen to be smaller than $\min(M, N)$ so that the approximation is low-rank. In audio signal processing, $\mathbf{V}$ is typically a magnitude $|\mathbf{X}|$ or power $|\mathbf{X}|^{\circ 2}$ spectrogram, where $\mathbf{X}$ is the short-time Fourier or Cosine transform of some signal $y(t)$ (the notation $\circ$ denotes element-wise operations throughout the paper). The short-time frequency transform $\mathbf{X}$ is computed by applying an orthogonal frequency transform $\mathbf{\Phi}$ to the *frames matrix* $\mathbf{Y} \in \mathbb{R}^{M\times N}$ which contains windowed segments of the original temporal signal $y(t)$ in its columns. M is the length of the window and N is the resulting number of time frames. As such, we have $\mathbf{X} = \mathbf{\Phi Y}$. Factorizing $\mathbf{V}$ as in (1) can lead to a meaningful decomposition where the dictionary $\mathbf{W}$ captures spectral patterns and the activation matrix $\mathbf{H}$ contains data decomposition coefficients. This decomposition can then be used to solve a variety of signal processing problems such as source separation [2, 3, 4] or music transcription [5, 6]. In the latter works, $\mathbf{V}$ is computed with a given off-the-shelf short-time frequency transform. This sets a limit to the accuracy of the factorization. To adress this issue, transform-learning NMF (TL-NMF) was introduced in [7, 8]. It computes an optimal transform from the input signal: the transform $\mathbf{\Phi}$ is learned *together* with the latent factors $\mathbf{W}$ and $\mathbf{H}$. TL-NMF has been employed successfully for source separation examples: it leads to better or comparable performance as compared with traditional fixed-transform NMF [7, 8, 9].

The contribution of this article is to propose a faster solver for TL-NMF. In [7], an orthogonal transform is learned using a projected gradient descent onto the orthogonal matrix manifold. In [8], a faster Jacobi approach (in which $\mathbf{\Phi}$ is searched as a product of Givens rotations) is proposed. In a different framework, [9] optimizes a nonsingular transform (not constrained to be orthogonal) with majorization-minimization (MM). In all cases, the cost of TL-NMF remains prohibitively large compared to standard NMF. The estimation of the transform is the computational bottleneck of the algorithms, and takes orders of magnitude more time than standard NMF. The present work aims at reducing the gap in terms of execution time between TL-NMF and traditional NMF in the orthogonal transform setting (which gently relaxes Fourier or Cosine transforms while still imposing orthogonality). To that purpose, we propose a quasi-Newton method on the orthogonal manifold.

The article is organized as follows. Section 2 introduces the optimization problem behind TL-NMF and presents the standard MM updates used for $\mathbf{W}$ and $\mathbf{H}$. Section 3 starts with a brief introduction to optimization on the orthogonal manifold, introduces previous work and presents the new quasi-Newton algorithm. Finally, Section 4 describes comparative experiments with synthetic and real data. Exploiting the reduced computational load, we highlight a previously unnoticed energy concentration phenomenon of the learned transform, and study the structure of the local minima of the objective function.

Notation. Scalars are written in lower-case (e.g., $v \in \mathbb{R}$), vectors in bold lower-case (e.g., $\mathbf{v} \in \mathbb{R}^M$) and matrices in bold upper-case (e.g., $\mathbf{V} \in \mathbb{R}^{M\times N}$), while tensors are in calligraphic upper-case (e.g., $\mathcal{H} \in \mathbb{R}^{M\times M\times M\times M}$). Entry (m, n) of a matrix $\mathbf{V}$ is denoted as v_{mn} or $[\mathbf{V}]_{mn}$ while entry (i, j, k, l) of a tensor $\mathcal{H}$ is denoted as $\mathcal{H}_{ijkl}$. The identity matrix of size M is denoted as $\mathbf{I}_M$. The element-wise operations between two matrices $\mathbf{A}$ and $\mathbf{B}$ are written $\mathbf{A} \circ \mathbf{B}$ and $\frac{\mathbf{A}}{\mathbf{B}}$ for the multiplication and division while $\mathbf{A}^{\circ p}$ and $|\mathbf{A}|$ denote the element-wise exponentiation and modulus, respectively. The l_1 norm of a matrix $||H||_1$ is the sum of the coefficients of $|H|$. The orthogonal matrix set $\mathcal{O}_M$ is the set of $M \times M$ matrices such that $\mathbf{MM}^\top = \mathbf{I}_M$. The Frobenius scalar product is denoted as $\langle \mathbf{A}|\mathbf{B}\rangle = \sum_{i,j} a_{ij}b_{ij}$. Given a fourth-order tensor $\mathcal{H}$ of size $M \times M \times M \times M$, the weighted Frobenius inner product is $\langle \mathbf{A}|\mathcal{H}|\mathbf{B}\rangle = \sum_{i,j,k,l} \mathcal{H}_{ijkl}a_{ij}b_{kl}$. The tensor $\mathcal{H}$ can be seen as

[†] Supported by the Center for Data Science, funded by the IDEX Paris-Saclay, ANR-11-IDEX-0003-02, and the European Research Council (ERC SLAB-StG-676943). [‡] Supported by the European Research Council (ERC FACTORY-CoG-6681839).

Algorithm 1: Alternate minimization for TL-NMF

Input : Frames matrix $\mathbf{Y}$, dictionary size K, minimization algorithm for transform learning $\mathcal{A}$, number of iterations of the TL minimization L, total number of iterations N_{it}

Initialize $\mathbf{\Phi}, \mathbf{W}, \mathbf{H}$.

for $n = 1, \cdots, N_{it}$ **do**

  NMF

  Compute the current spectrogram $\mathbf{V} = |\mathbf{\Phi Y}|^{\circ 2}$

  Decrease $\mathcal{C}_\lambda$ w.r.t. $(\mathbf{W}, \mathbf{H})$ (step $\star$)

  TL

  Compute $\hat{\mathbf{V}} = \mathbf{WH}$

  Update $\Phi \leftarrow \mathcal{A}(\hat{\mathbf{V}}, \mathbf{Y}, \mathbf{\Phi}, L)$

end

Output: $\mathbf{\Phi}, \mathbf{W}, \mathbf{H}$

a $(M \times M) \times (M \times M)$ matrix acting on squares matrices seen as $(M \times M)$ vectors. The Itakura-Saito divergence is given by $d_{\text{IS}}(x, y) = \frac{x}{y} - \log(\frac{x}{y}) - 1$. Finally, δ_{ij} is the Kronecker delta function of (i, j) equal to 1 if $i = j$ and 0 otherwise.

2. NMF WITH TRANSFORM LEARNING

2.1. Objective function

TL-NMF consists in solving a NMF problem while learning a data-adapted transform [7]. This is done by minimizing some measure of fit between the transformed data $|\mathbf{\Phi Y}|^{\circ 2}$ and the factorized expression $\mathbf{WH}$ where we here assume that $\mathbf{\Phi}$ is a real-valued orthogonal matrix (of size $M \times M$). In addition, a penalty is added to promote sparsity of the activation coefficients. The TL-NMF problem thus writes:

$$\min_{\mathbf{\Phi},\mathbf{W},\mathbf{H}} \mathcal{C}_\lambda(\mathbf{\Phi}, \mathbf{W}, \mathbf{H}) = D_{IS}(|\mathbf{\Phi Y}|^{\circ 2} \mid \mathbf{WH}) + \lambda \frac{M}{K} ||\mathbf{H}||_1$$
$$\text{s.t. } \mathbf{W} \geq 0, \mathbf{H} \geq 0, \ \forall k, ||\mathbf{w}_k||_1 = 1, \mathbf{\Phi\Phi}^\top = \mathbf{I}_M, \quad (2)$$

where $\mathbf{w}_k$ is the k-th column of $\mathbf{W}$ and $D_{IS}(\cdot|\cdot)$ is the Itakura-Saito (IS) divergence defined as $D_{IS}(\mathbf{A}|\mathbf{B}) = \sum_{m,n} d_{IS}(a_{mn}|b_{mn}) = \sum_{m,n} \frac{a_{mn}}{b_{mn}} - \log \frac{a_{mn}}{b_{mn}} - 1$. Note that any other measure of fit could be used with no loss of generality. However, the IS divergence is particularly relevant for decomposing power spectrograms [10]. The M/K factor makes the measure of fit and the penalty term of comparable orders of magnitude.

The problem (2) is solved using *alternate minimization*, summarized in Algorithm 1. It alternates between two steps. In the NMF step, the current "spectrogram" $\mathbf{V} = |\mathbf{\Phi Y}|^{\circ 2}$ is fixed and the algorithm decreases $\mathcal{C}_\lambda$ with respect to (w.r.t.) $\mathbf{W}$ and $\mathbf{H}$. This is done using classical NMF MM update rules, described in the next section. In the transform-learning part, the factorization $\hat{\mathbf{V}} = \mathbf{WH}$ is fixed, and the algorithm decreases $\mathcal{C}_\lambda$ w.r.t. $\mathbf{\Phi}$, using an optimization algorithm denoted as $\mathcal{A}$. This article proposes a fast algorithm $\mathcal{A}$ for the minimization of $\mathcal{C}_\lambda$ w.r.t. $\mathbf{\Phi}$.

2.2. Majorization-minimization updates of W and H

We update $\mathbf{W}$ and $\mathbf{H}$ (step $\star$ in Algorithm 1) with the standard multiplicative updates derived from a majorization-minimization procedure [11]. The sum-to-one constraint on the columns of $\mathbf{W}$ (which is necessary to avoid degenerate solutions) can be rigorously enforced using a change of variable, like in [12, 13]. The updates read:

$$\mathbf{H} \leftarrow \mathbf{H} \circ \left[\frac{\mathbf{W}^T \left((\mathbf{WH})^{\circ -2} \circ |\mathbf{\Phi Y}|^{\circ 2} \right)}{\mathbf{W}^T (\mathbf{WH})^{\circ -1} + \lambda \frac{M}{K} \mathbf{1}_{K \times N}} \right]^{\circ \frac{1}{2}},$$

$$\mathbf{W} \leftarrow \mathbf{W} \circ \left[\frac{\left((\mathbf{WH})^{\circ -2} \circ |\mathbf{\Phi Y}|^{\circ 2} \right) \mathbf{H}^T}{(\mathbf{WH})^{\circ -1} \mathbf{H}^T + \lambda \frac{M}{K} \mathbf{1}_{M \times N} \mathbf{H}^T} \right]^{\circ \frac{1}{2}}.$$

They should be followed by a joint normalization of the columns of $\mathbf{W}$ and rows of $\mathbf{H}$ [12, 13].

3. QUASI-NEWTON UPDATE OF THE TRANSFORM Φ

3.1. Optimization on the orthogonal manifold

This section focuses on the minimization of $\mathcal{C}_\lambda$ with respect to $\mathbf{\Phi}$. In the following, we define $\hat{\mathbf{V}} = \mathbf{WH}$, and let $\mathcal{L}(\mathbf{\Phi}) = D_{IS}(|\mathbf{\Phi Y}|^{\circ 2}|\hat{\mathbf{V}})$. We may write:

$$\mathcal{L}(\mathbf{\Phi}) = \sum_{m=1}^{M} \sum_{n=1}^{N} f_{\hat{v}_{mn}}([\mathbf{\Phi Y}]_{mn}), \quad (3)$$

where we define $f_v(x) = d_{IS}(x^2, v) = \frac{x^2}{v} - 2\log(\frac{x}{v}) - 1$. The orthogonality constraint imposed to $\mathbf{\Phi}$ implies that (3) should be minimized on the *orthogonal matrix manifold* $\mathcal{O}_M$. This manifold appears in many optimization problems and its geometry is well-studied [14]. To derive an iterative algorithm that minimizes (3), we propose to parametrize the neighborhood of an iterate $\mathbf{\Phi}^t$ via the *matrix exponential* (following, e.g., [15]). We set:

$$\mathbf{\Phi}^{t+1} = \exp(\mathbf{E})\mathbf{\Phi}^t, \quad (4)$$

where $\mathbf{E}$ is an anti-symmetric matrix. If $\mathbf{\Phi}^t$ is orthogonal, this update enforces that $\mathbf{\Phi}^{t+1}$ remains orthogonal. It thus provides a natural framework for iterative optimization over the orthogonal manifold.

3.2. Previous methods

A projected gradient method is presented in [7]. Iterates are of the form:

$$\mathbf{\Phi} \leftarrow \Pi((\mathbf{I}_M - \eta \mathbf{G})\mathbf{\Phi}), \quad (5)$$

where $\mathbf{G}$ is the *natural gradient* [16] of $\mathcal{L}$, η is a step-size, and Π is the projection to the manifold, given by $\Pi(\mathbf{C}) = (\mathbf{CC}^\top)^{-\frac{1}{2}}\mathbf{C}$. The main drawback is that, as a first order method, it is hard to have a proper step size policy, and the convergence is at most linear [17].

A variant was proposed in [8] where the transform was updated using Givens rotations as:

$$\mathbf{\Phi} \leftarrow \mathbf{R}_{pq}(\theta)\mathbf{\Phi} \quad (6)$$

where $\mathbf{R}_{pq}$ is a unidirectional rotation matrix with axis (p, q) and angle θ. This update rule results in an acceleration because the single-axis rotations are cheap to compute. However, finding the best angle θ given an axis (p, q) was shown to involve a highly non-convex problem with the presence of many local minima. As such θ is selected by grid search which is not entirely satisfactory.

3.3. Derivatives of the objective function

In this section, the derivatives of $\mathcal{L}$ with respect to the parametrization (4) are computed. The gradient is an $M \times M$ matrix denoted as $\mathbf{G}$, and the Hessian is a $M \times M \times M \times M$ tensor denoted as $\mathcal{H}$. They are obtained from the following second-order Taylor expansion:

$$\mathcal{L}(\exp(\mathbf{E})\mathbf{\Phi}) = \mathcal{L}(\mathbf{\Phi}) + \langle \mathbf{G}|\mathbf{E} \rangle + \frac{1}{2}\langle \mathbf{E}|\mathcal{H}|\mathbf{E} \rangle + \mathcal{O}(||\mathbf{E}||^3). \quad (7)$$

Using $\mathbf{X} = \mathbf{\Phi Y}$, the gradient is given by

$$\mathbf{G}_{ij} = \sum_{n=1}^{N} f'_{\hat{v}_{in}}(x_{in})x_{jn} = 2\sum_{n=1}^{N}(\frac{x_{in}}{\hat{v}_{in}} - \frac{1}{x_{in}})x_{jn} \quad (8)$$

and the Hessian is given by

$$\mathcal{H}_{ijkl} = \delta_{ik}\sum_{n=1}^{N} f''_{\hat{v}_{in}}(x_{in})x_{jn}x_{ln} + \delta_{jk}\mathbf{G}_{il}. \quad (9)$$

Newton's method. Newton method on the manifold would take $\mathbf{E} = -\Pi_A(\mathcal{H}^{-1}\mathbf{G})$, where Π_A is the projection onto the anti-symmetric matrices:

$$\Pi_A(\mathbf{C}) = \frac{\mathbf{C} - \mathbf{C}^\top}{2}. \quad (10)$$

Note that this projection is much cheaper to compute than Π. Newton's method provides fast convergence, but is not practical for several reasons. First, it requires the computation of the Hessian. The complexity of this operation is $O(M^3 \times N)$. Besides, the cost of computing a gradient is $O(M^2 \times N)$. Thus, a gradient method can roughly perform M iterations when Newton's method performs one. Second, because the problem is non-convex, the Hessian should be regularized to enforce its positive-definiteness, thereby guaranteeing that $-\mathcal{H}^{-1}\mathbf{G}$ is a descent direction. A standard regularization procedure consists in adding $\mu\mathbf{I}$ to the Hessian where $\mu > \max(0, -\lambda_{\text{min}})$ and where λ_{min} is the smallest eigenvalue of $\mathcal{H}$. The Hessian is sparse because of the δ_{ik} and δ_{jk} factors, but its sparsity structure does not help in computing the key quantity λ_{min}. As such one would have to compute the smallest eigenvalue of a $M^2 \times M^2$ matrix which is prohibitively expensive. Finally, solving the $M^2 \times M^2$ linear system $\mathcal{H}\mathbf{E} = -\mathbf{G}$ using, e.g., Gaussian elimination has complexity $O(M^6)$, which is orders of magnitude higher than the computation of the gradient.

3.4. A fast algorithm based on Hessian approximation

To derive a practical *quasi-Newton* algorithm, one can observe that the Hessian of $\mathcal{L}$ has two terms. The second term, $\delta_{jk}\mathbf{G}_{il}$, cancels when the algorithm is close to convergence, so we may ignore it. As an approximation of the first term, we impose that it cancels when $j \neq l$, leading to the following Hessian approximation:

$$\tilde{\mathcal{H}}_{ijkl} = \delta_{ik}\delta_{jl}\sum_{n=1}^{N} f''_{\hat{v}_{in}}(x_{in})x_{jn}^2 \quad (11)$$

$$= 2\delta_{ik}\delta_{jl}\sum_{n=1}^{N}(\frac{1}{\hat{v}_{in}} + \frac{1}{x_{in}^2})x_{jn}^2. \quad (12)$$

Our approximation provides an even sparser version of the true Hessian. Then, then proposed update for the transform reads:

$$\boxed{\mathbf{\Phi} \leftarrow \exp(-\eta\Pi_A(\tilde{\mathcal{H}}^{-1}\mathbf{G}))\mathbf{\Phi}} \quad (13)$$

where η is a step size. The step size is chosen to verify the Wolfe conditions [18] and is computed using the classical interpolation algorithm thoroughly described in [17, pp. 59-60]. Informally, Wolfe conditions guarantee that the objective function is sufficiently decreased by the step size, and that the projected gradient in the search direction is also decreased. These conditions are critical to obtain convergence of quasi-Newton methods, and in practice help in achieving fast convergence.

Algorithm 2: Algorithm $\mathcal{A}$: Fast transform learning

Input : Current factorization $\hat{\mathbf{V}}$, frames matrix $\mathbf{Y}$, current transform $\mathbf{\Phi}$, number of iterations L.
for *l=1,...,L* **do**
  Compute G and $\tilde{\mathcal{H}}$ using Eqs. (8), (12)
  Compute the search direction $\mathbf{E} = -\Pi_A(\tilde{\boldsymbol{H}}^{\circ -1} \circ \mathbf{G})$
  Compute a step size $\eta > 0$ satisfying the Wolfe conditions.
  Update $\mathbf{\Phi} \leftarrow \exp(\eta\mathbf{E})\mathbf{\Phi}$
end
Output: New transform $\mathbf{\Phi}$

Denote by $\tilde{\boldsymbol{H}}$ the matrix with coefficients $\tilde{h}_{ij} = 2\sum_{n=1}^{N}(\frac{1}{\hat{v}_{in}} + \frac{1}{x_{in}^2})x_{jn}^2$, so that $\tilde{\mathcal{H}}_{ijkl} = \delta_{ik}\delta_{jl}\tilde{h}_{ij}$. Our quasi-Newton's method solves all the aforementioned problems of Newton's method. The approximated Hessian is

- **cheap**: computing $\tilde{\mathcal{H}}$ has the same complexity as computing a gradient, i.e., $O(M^2 \times N)$.
- **positive definite:** the approximation boils down to a diagonal operator, i.e., $\tilde{\mathcal{H}}\mathbf{E} = \tilde{\boldsymbol{H}} \circ \mathbf{E}$. Hence, its eigenvalues are the coefficients $\tilde{h}_{ij}$, which are all nonnegative. As such, our method does not require Hessian regularization.
- **easy to invert:** because it boils down to a diagonal operator, we have $\tilde{\mathcal{H}}^{-1}\mathbf{G} = \tilde{\boldsymbol{H}}^{\circ -1} \circ \mathbf{G}$. Inversion is $O(M^2)$, which is negligible compared to the cost of computing the gradient.

The resulting optimization procedure is described in Algorithm 2.

3.5. Relation to independent component analysis (ICA)

This objective function (3) is reminiscent of maximum-likelihood ICA where the maximum-likelihood objective is given by [19]:

$$\mathcal{L}(\mathbf{\Phi}) = -N\log|\det(\mathbf{\Phi})| + \sum_{m=1}^{M}\sum_{n=1}^{N} f([\mathbf{\Phi Y}]_{mn}), \quad (14)$$

where f is a pre-specified function. Under the orthogonal constraint, $\log|\det(\mathbf{\Phi})|$ becomes constant. As such, the ICA objective function shares the same dependency in $\mathbf{\Phi}$ with TL-NMF and the algorithm proposed in this paper is inspired by the ICA acceleration techniques proposed in [15].

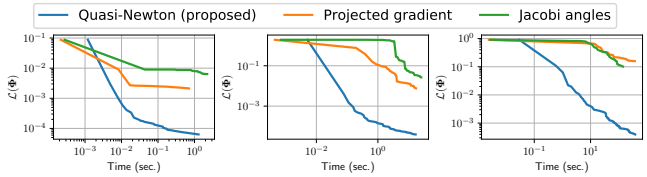
4. EXPERIMENTS

The following experiments are run on a single core of a laptop equipped with an Intel Core i7-6600U @ 2.6 GHz processor and 16 GB of RAM. The Python code is available online.[1]

4.1. Synthetic data

We first focus on the sole optimization of $\mathcal{L}$, and not on the full TL-NMF procedure. For this experiment, we generate random normal matrices $\mathbf{Y}$ of size $M \times N$, for $N = 1000$ and $M \in [10, 100, 500]$, and a random transform $\mathbf{\Phi}^* \in \mathcal{O}_M$. We set $\hat{\mathbf{V}} = |\mathbf{\Phi}^*\mathbf{Y}|^{\circ 2}$, so that the minimum of $\mathcal{L}(\mathbf{\Phi})$ is 0. Algorithms start from an orthogonal initialization $\mathbf{\Phi}^0$ in the vicinity of $\mathbf{\Phi}^*$. More precisely, we set

[1] https://github.com/pierreablin/tlnmf



(a) $M = 10$ (b) $M = 100$ (c) $M = 500$

Fig. 1: Convergence curves with synthetic data.

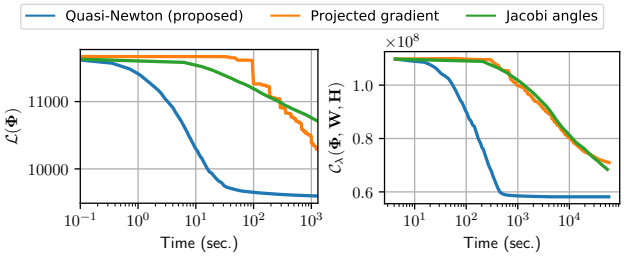


Fig. 2: Convergence curves with real data. Left: minimization of $\mathcal{L}(\mathbf{\Phi})$ only. Right: full TL-NMF optimization.

$\mathbf{\Phi}^0 = \exp(\mathbf{E})\mathbf{\Phi}^*$ where $\mathbf{E} = 10^{-3}\Pi_A(\mathcal{N}(0, \mathbf{I}_M))$. Fig. 1 shows the convergence curve of the proposed method, projected gradient [7] and Jacobi search [8]. The proposed quasi-Newton approach leads to a drastic improvement in speed of convergence.

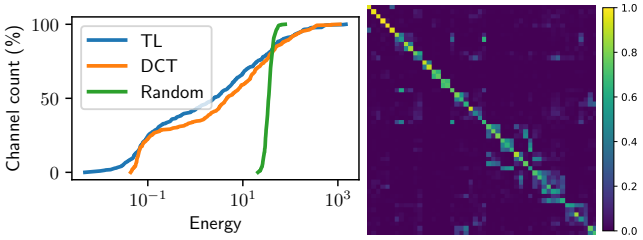
4.2. Real data

Experimental setup. We consider a 108 seconds-long excerpt from *My Heart (Will Always Lead Me Back To You)* by Louis Armstrong and His Hot Five. The sampling rate is $f_s = 11025$ Hz. Using a 40 ms-long analysis windows ($M = 440$) with 50% overlap between two frames, we obtain $N = 5407$. The rank of the decomposition is fixed to $K = 10$, which is known empirically to provide a satisfactory decomposition with traditional NMF [10].

Comparison of the algorithms performance. In a first experiment, we first run traditional IS-NMF on the DCT spectrogram of the input signal and store $\hat{\mathbf{V}}$. Then the three transform learning algorithms are run with fixed $\hat{\mathbf{V}}$ and from a random starting point for $\mathbf{\Phi}$. This provides a realistic setting to compare their performance in optimizing $\mathcal{L}(\mathbf{\Phi})$. Full TL-NMF (with free $\mathbf{W}$ and $\mathbf{H}$) are computed in a second experiment, using the same random starting points. The three different transform learning algorithms are run with $L = 5$. Results for the two experiments are shown in Fig. 2 and illustrate the superiority of the proposed quasi-Newton algorithm.

We now discuss some features of the transform learned with (full) TL-NMF using the quasi-Newton algorithm. We will refer to the rows $\boldsymbol{\phi}_1, \cdots, \boldsymbol{\phi}_M$ of $\mathbf{\Phi}$ as *atoms* (real-valued vectors of size M). The learned atoms are not shown here due to space limitation but are similar to those obtained in [7, 8].

Energy concentration. The contributed *energy* of a single atom $\boldsymbol{\phi}_i$ is defined as $e_i = \sum_{n=1}^{N} [\boldsymbol{\phi}_i \mathbf{Y}]_n^2$. Fig. 3a shows the cumulative distribution of the energies for three different transforms: $\mathbf{\Phi}$ estimated by TL-NMF, the DCT, and a random orthogonal matrix. The energy is evenly spread across the atoms of the random transform, while



(a) (b)

Fig. 3: **(a)**: Cumulative distribution of the atoms contributing energies for three transforms $\mathbf{\Phi}$. **(b)**: Similarity matrix between the 50 most-contributing atoms learnt from two different random initializations.

a few atom contain most of the energy for the DCT: this an *energy concentration* phenomenon. The energy concentration phenomenon is accentuated by transform learning. This behavior was observed with other music datasets as well.

Reliability of the learned transform. The problem solved by TL-NMF is non-convex, hence different initializations can lead to different local minima. We investigate the structure of the local minima returned by the proposed quasi-Newton algorithm using a technique similar to ICASSO in ICA [20]. It appears that a subset of atoms are reliably returned by the algorithm, regardless of initialization. To observe this behavior, we consider two transforms obtained from two random initializations. We select the 50 most-contributing atoms based of the values of e_i, yielding two matrices $\underline{\mathbf{\Phi}}^1$ and $\underline{\mathbf{\Phi}}^2$ of size 50×440. We compute the correlation matrix $\mathbf{T} = \underline{\mathbf{\Phi}}^1 \underline{\mathbf{\Phi}}^{2\top}$ of size 50×50 and find a permutation matrix $\mathbf{P}$ such that $\mathbf{PT}$ is as block-diagonal as possible. The absolute value of the resulting matrix is displayed in Fig. 3b. It is well structured and shows in particular that the first 6 atoms (top left) are the same. Furthermore, some pairwise couplings are also uncovered. The diagonal blocks in Fig. 3b correspond to sets of atoms such that $\text{Span}(\boldsymbol{\phi}_i^1, \boldsymbol{\phi}_j^1) = \text{Span}(\boldsymbol{\phi}_{i'}^2, \boldsymbol{\phi}_{j'}^2)$.

5. CONCLUSION

We introduced a quasi-Newton method on the orthogonal manifold to solve the TL-NMF problem. It relies on a sparse approximation of the Hessian. The proposed method outperforms the state-of-the-art methods by orders of magnitude. On the laptop used for the experiments, the whole estimation took about 10 minutes for a $\sim$2-minutes signal, while NMF without transform learning takes roughly 2 minutes. This work is thus a step towards making TL-NMF a practical tool for music signal processing. The shortened time of estimation also helps investigate properties of the learned transform without prohibitive computational burden. Results on the concentration of energy obtained by TL-NMF suggest that an algorithm that only learns a few atoms instead of M could remain informative, while drastically reducing the computational cost of TL-NMF, since the number of parameters would plummet. We intend to study this matter in a future work.

6. REFERENCES

[1] Daniel D. Lee and H. Sebastian Seung, "Learning the parts of objects by non-negative matrix factorization," *Nature*, vol. 401, no. 6755, pp. 788–791, 1999.

[2] Paris Smaragdis, Cédric Févotte, Gautham J Mysore, Nasser Mohammadiha, and Matthew Hoffman, "Static and dynamic source separation using nonnegative factorizations: A unified view," *IEEE Signal Processing Magazine*, vol. 31, no. 3, pp. 66–75, 2014.

[3] Emmanuel Vincent, Tuomas Virtanen, and Sharon Gannot, *Audio source separation and speech enhancement*, John Wiley & Sons, 2018.

[4] Andrzej Cichocki, Rafal Zdunek, Anh Huy Phan, and Shun-ichi Amari, *Nonnegative matrix and tensor factorizations: applications to exploratory multi-way data analysis and blind source separation*, John Wiley & Sons, 2009.

[5] Paris Smaragdis and Judith C. Brown, "Non-negative matrix factorization for polyphonic music transcription," in *Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, 2003.

[6] Emmanuel Vincent, Nancy Bertin, and Roland Badeau, "Harmonic and inharmonic nonnegative matrix factorization for polyphonic pitch transcription," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2008, pp. 109–112.

[7] Dylan Fagot, Herwig Wendt, and Cédric Févotte, "Nonnegative matrix factorization with transform learning," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 2431–2435.

[8] Herwig Wendt, Dylan Fagot, and Cédric Févotte, "Jacobi algorithm for nonnegative matrix factorization with transform learning," in *Proc. European Signal Processing Conference (EUSIPCO)*, 2018.

[9] Kazuyoshi Yoshii, Koichi Kitamura, Yoshiaki Bando, Eita Nakamura, and Tatsuya Kawahara, "Independent low-rank tensor analysis for audio source separation," in *Proc. European Signal Processing Conference (EUSIPCO)*, 2018.

[10] Cédric Févotte, Nancy Bertin, and Jean-Louis Durrieu, "Nonnegative matrix factorization with the Itakura-Saito divergence: With application to music analysis," *Neural computation*, vol. 21, no. 3, pp. 793–830, 2009.

[11] Cédric Févotte and Jérôme Idier, "Algorithms for nonnegative matrix factorization with the β-divergence," *Neural computation*, vol. 23, no. 9, pp. 2421–2456, 2011.

[12] Augustin Lefevre, Francis Bach, and Cédric Févotte, "Itakura-Saito nonnegative matrix factorization with group sparsity," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2011, pp. 21–24.

[13] Slim Essid and Cédric Févotte, "Smooth nonnegative matrix factorization for unsupervised audiovisual document structuring," *IEEE Transactions on Multimedia*, vol. 15, no. 2, pp. 415–425, 2013.

[14] Pierre-Antoine Absil, Robert Mahony, and Rodolphe Sepulchre, *Optimization algorithms on matrix manifolds*, Princeton University Press, 2009.

[15] Pierre Ablin, Jean-François Cardoso, and Alexandre Gramfort, "Faster ICA under orthogonal constraint," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018.

[16] Shun-Ichi Amari, "Natural gradient works efficiently in learning," *Neural computation*, vol. 10, no. 2, pp. 251–276, 1998.

[17] Jorge Nocedal and Stephen J Wright, *Numerical Optimization*, Springer, 1999.

[18] Philip Wolfe, "Convergence conditions for ascent methods," *SIAM review*, vol. 11, no. 2, pp. 226–235, 1969.

[19] Dinh Tuan Pham and Philippe Garat, "Blind separation of mixture of independent sources through a quasi-maximum likelihood approach," *IEEE Trans. on Signal Processing*, vol. 45, no. 7, pp. 1712–1725, 1997.

[20] Johan Himberg, Aapo Hyvärinen, and Fabrizio Esposito, "Validating the independent components of neuroimaging time series via clustering and visualization," *Neuroimage*, vol. 22, no. 3, pp. 1214–1222, 2004.