# FLEXIBLE HALS ALGORITHMS FOR SPARSE NON-NEGATIVE MATRIX/TENSOR FACTORIZATION

*Andrzej CICHOCKI*[*], *Anh Huy PHAN and Cesar CAIAFA*[†]

RIKEN Brain Science Institute, LABSP, Wako-shi, Saitama 351-0198, JAPAN

**ABSTRACT**

In this paper we propose a family of new algorithms for non-negative matrix/tensor factorization (NMF/NTF) and sparse nonnegative coding and representation that has many potential applications in computational neuroscience, multi-sensory, multidimensional data analysis and text mining. We have developed a class of local algorithms which are extensions of Hierarchical Alternating Least Squares (HALS) algorithms proposed by us in [1]. For these purposes, we have performed simultaneous constrained minimization of a set of robust cost functions called alpha and beta divergences. Our algorithms are locally stable and work well for the NMF blind source separation (BSS) not only for the over-determined case but also for an under-determined (over-complete) case (i.e., for a system which has less sensors than sources) if data are sufficiently sparse. The NMF learning rules are extended and generalized for $N$-th order nonnegative tensor factorization (NTF). Moreover, new algorithms can be potentially accommodated to different noise statistics by just adjusting a single parameter. Extensive experimental results confirm the validity and high performance of the developed algorithms, especially, with usage of the multi-layer hierarchical approach [1].

## 1. INTRODUCTION - PROBLEM FORMULATION

Non-negative Matrix Factorization (NMF) and its extension Non-negative Tensor Factorization (NTF) - models with non-negativity and sparsity constraints have been recently proposed as sparse and efficient representations of signals, images, or general multidimensional data [1, 2, 3, 4]. From a viewpoint of data analysis, NMF/NTF are very attractive because they take into account spatial and temporal correlations between variables and usually provide sparse common factors or hidden (latent) nonnegative components with physical or physiological meaning and interpretation [5, 2, 6].

In this paper, we consider at first a simple NMF model described as

$$\boldsymbol{Y} = \boldsymbol{A}\boldsymbol{X} + \boldsymbol{R}, \tag{1}$$

where $\boldsymbol{Y} = [y_{ik}] \in \mathbb{R}^{I\times K}$ is a known data matrix, $\boldsymbol{A} = [\boldsymbol{a}_1, \boldsymbol{a}_2, \ldots, \boldsymbol{a}_J] \in \mathbb{R}_+^{I\times J}$ is an unknown basis (mixing) matrix with vectors $\boldsymbol{a}_j$, $\boldsymbol{X} = [\underline{\boldsymbol{x}}_1^T, \underline{\boldsymbol{x}}_2^T, \ldots, \underline{\boldsymbol{x}}_J^T]^T \in \mathbb{R}_+^{J\times K}$ is a matrix representing unknown nonnegative components $\underline{\boldsymbol{x}}_j$ and $\boldsymbol{R} = [r_{ik}] \in \mathbb{R}^{I\times K}$ represents errors or noise. Our primary objective is to estimate the vectors $\boldsymbol{a}_j$ of the mixing (basis) matrix $\boldsymbol{A}$ and the sources $\underline{\boldsymbol{x}}_j$ (rows of the matrix $\boldsymbol{X}$), subject to non-negativity constraints[1]. The simple NMF model (1) can be naturally extended to the NTF (or nonnegative PARAFAC) as follows: For a given $N$-th order tensor $\underline{\boldsymbol{Y}} \in \mathbb{R}^{I_1\times I_2\times\cdots\times I_N}$ perform a nonnegative factorization (decomposition) into a set of $N$ unknown matrices: $\boldsymbol{U}^{(n)} = [\boldsymbol{u}_1^{(n)}, \boldsymbol{u}_2^{(n)}, \ldots, \boldsymbol{u}_J^{(n)}] \in \mathbb{R}_+^{I_n\times J}$, $(n = 1, 2, \ldots, N)$ representing the common (loading) factors, i.e., [3, 7]

$$\underline{\boldsymbol{Y}} \approx \sum_{j=1}^{J} \boldsymbol{u}_j^{(1)} \circ \boldsymbol{u}_j^{(2)} \circ \ldots \circ \boldsymbol{u}_j^{(N)} \tag{2}$$

$$= \sum_{j=1}^{J} [\![\boldsymbol{u}_j^{(1)}, \boldsymbol{u}_j^{(2)}, \ldots, \boldsymbol{u}_j^{(N)}]\!] = [\![\{\boldsymbol{U}\}]\!] = \underline{\boldsymbol{Z}}\,, \tag{3}$$

with $\|\boldsymbol{u}_{n,j}\|_2 = 1$ for $n = 1, 2, \ldots, N-1$, $\forall\, j = 1, 2, \ldots, J$, where $\circ$ means outer product of vectors and $\underline{\boldsymbol{Z}}$ is an estimated or approximated (actual) tensor. A residuum tensor defined as $\underline{\boldsymbol{R}} = \underline{\boldsymbol{Y}} - \underline{\boldsymbol{Z}}$ represents noise or errors depending on applications. This model can be referred as nonnegative version of CANDECOMP proposed by Carroll and Chang and PARAFAC independently by Harshman and Kruskal .

Most of the known algorithms for the NTF/NMF model are based on alternating least squares (ALS) minimization of the squared Euclidean distance [2, 3, 7]. Especially, for the NMF we minimize the following cost function:

$$D_F(\boldsymbol{Y}\|\boldsymbol{A}\boldsymbol{X}) = \frac{1}{2}\|\boldsymbol{Y} - \boldsymbol{A}\boldsymbol{X}\|_F^2, \tag{4}$$

[*] Also from Dept. EE Warsaw University of Technology and Systems Research Institute, Polish Academy of Science, POLAND

[†] On leave from Engineering Faculty, University of Buenos Aires, ARGENTINA

[1] Usually, a sparsity constraint is naturally and intrinsically provided due to nonlinear projected approach (e.g., half-wave rectifier or adaptive nonnegative shrinkage with gradually decreasing threshold).

and for NTF with the model (2)

$$D_F(\underline{\boldsymbol{Y}} \,||\, [\![\{\boldsymbol{U}\}]\!]) = \frac{1}{2}||\underline{\boldsymbol{Y}} - \sum_{j=1}^{J} (\boldsymbol{u}_j^{(1)} \circ \boldsymbol{u}_j^{(2)} \circ \ldots \circ \boldsymbol{u}_j^{(N)})||_F^2, \quad (5)$$

subject to nonnegativity constraints and often additional constraints such as sparsity or smoothness. A basic approach to the above formulated optimization problems is alternating minimization or alternating projection: The specified cost function is alternately minimized with respect to sets of parameters, each time optimizing one set of arguments while keeping the others fixed.

In fact, the NTF algorithms with global learning rules perform computation and update based on whole matrices. For tensor and its factorization problem, we often face large factors and tensors which have huge number of elements. Therefore, the global learning rules meet a lot of difficult problems in tensor factorization and decomposition such as limitation of memory resources, number of elements in vectors or matrices, etc. Moreover, global learning rules are not suitable for dynamic tensor factorization in which the number of components is unknown or changes in time and must be estimated during optimization process.

In this paper, we use a completely different and more sophisticated approach. Instead of minimizing one or two cost functions, we minimize a set of local cost functions (alpha and beta divergences) with a single parameter (alpha or beta). The proposed algorithms are suitable for large scale dataset due to fast processing speed, simplicity, local nature of learning rules and easy adaptation to changing the number of components.

The majority of known algorithms for NMF work only if the following assumption $K >> I \geq J$ is satisfied, where $J$ is the number of the nonnegative components. The NMF algorithms here proposed are suitable also for the underdetermined case, i.e., for $K > J > I$, if sources are sparse enough. Moreover, proposed algorithms are robust with respect to noise and suitable for large scale problems.

## 2. FLEXIBLE LOCAL ALGORITHMS USING ALPHA DIVERGENCE

For the NMF problem (1) we define the alpha-divergence as follows (as used in [8, 4]):

$$D_\alpha^{(j)}\left([\boldsymbol{Y}^{(j)}]_+) \,||\, \boldsymbol{a}_j \underline{\boldsymbol{x}}_j\right) =$$

$$\begin{cases} \sum_{ik} \left( ([y_{ik}^{(j)}]_+) \ln\left(\frac{[y_{ik}^{(j)}]_+}{z_{ik}}\right) - [y_{ik}^{(j)}]_+ + z_{ik} \right), & \alpha{=}1, \quad (6a) \\ \sum_{ik} \left( z_{ik} \ln\left(\frac{z_{ik}}{[y_{ik}^{(j)}]_+}\right) + [y_{ik}^{(j)}]_+ - z_{ik} \right), & \alpha{=}0, \quad (6b) \\ \sum_{ik} \left( \frac{[y_{ik}^{(j)}]_+}{\alpha(\alpha-1)} \left[ \left(\frac{[y_{ik}^{(j)}]_+}{z_{ik}}\right)^{\alpha-1} - 1 \right] - \frac{[y_{ik}^{(j)}]_+ - z_{ik}}{\alpha} \right), & \alpha \neq 0,1, \quad (6c) \end{cases}$$

where $[y_{ik}^{(j)}]_+ = \max\{0, y_{ik}^{(j)}\}$, $y_{ik}^{(j)} = [\boldsymbol{Y}]_{ik} - \sum_{r \neq j} a_{ir} x_{rk}$ and $z_{ik} = a_{ij} x_{jk}$ for $j = 1, 2, \ldots, J$.

The choice of the parameter $\alpha \in \mathbb{R}$ depends on the statistical distribution of noise and data. As special cases for $\alpha = 2, 0.5, -1$, we obtain the Pearson's chi squared, Hellinger and Neyman's chi-square distances, respectively, while for the cases $\alpha = 1$ and $\alpha = 0$ the divergence has to be defined by the limits of (6c) as $\alpha \to 1$ and $\alpha \to 0$, respectively. When these limits are evaluated one obtains the generalized Kullback-Leibler divergence defined by equations (6a) and (6b) [9, 4].

The gradient of the alpha divergence (6), for $\alpha \neq 0$, with respect to $a_{ij}$ and $x_{jk}$ can be expressed in a compact form as:

$$\frac{\partial D_\alpha^{(j)}}{\partial x_{jk}} = \frac{1}{\alpha} \sum_i a_{ij} \left[ 1 - \left( \frac{[y_{ik}^{(j)}]_+}{z_{ik}} \right)^\alpha \right], \quad (7)$$

$$\frac{\partial D_\alpha^{(j)}}{\partial a_{ij}} = \frac{1}{\alpha} \sum_k x_{jk} \left[ 1 - \left( \frac{[y_{ik}^{(j)}]_+}{z_{ik}} \right)^\alpha \right]. \quad (8)$$

However, instead of applying here the standard gradient descent, we use a rather projected (nonlinearly transformed) gradient approach (which can be considered as a generalization of the exponentiated gradient):

$$x_{jk} \leftarrow \Phi^{-1}\left( \Phi(x_{jk}) - \eta_{jk} \frac{\partial D_\alpha^{(j)}}{\partial x_{jk}} \right), \quad (9)$$

$$a_{ij} \leftarrow \Phi^{-1}\left( \Phi(a_{ij}) - \tilde{\eta}_{ij} \frac{\partial D_\alpha^{(j)}}{\partial a_{ij}} \right), \quad (10)$$

where $\Phi(x)$ is a suitable chosen function. It can be shown that using such nonlinear scaling or transformation provides stable solution and the gradients are much better behaved in $\Phi$ space [1, 10]. In our case, for $\Phi(x) = x^\alpha$ by choosing the learning rates as follows $\eta_{jk} = (\alpha x_{jk}^\alpha \sum_i z_{ik}^\alpha)/\sum_i a_{ij} z_{ik}^\alpha$ and $\tilde{\eta}_{ij} = (\alpha a_{ij}^\alpha \sum_k z_{ik}^\alpha)/\sum_k x_{jk} z_{ik}^\alpha$ after some simple mathematical manipulations we obtain a new multiplicative local alpha algorithm [1]:

$$\underline{\boldsymbol{x}}_j \leftarrow \left( \frac{\boldsymbol{a}_j^T \left( [\boldsymbol{Y}^{(j)}]_+ \right)^{.\alpha}}{\boldsymbol{a}_j^T \boldsymbol{a}_j^{.\alpha}} \right)^{.1/\alpha}, \quad \boldsymbol{a}_j \leftarrow \left( \frac{\left( [\boldsymbol{Y}^{(j)}]_+ \right)^{.\alpha} \underline{\boldsymbol{x}}_j^T}{\underline{\boldsymbol{x}}_j^{.\alpha} \underline{\boldsymbol{x}}_j^T} \right)^{.1/\alpha}, \quad (11)$$

where the "rise to the power" operations $\boldsymbol{x}^{.\alpha}$ are performed componentwise. The above algorithm can be generalized to the following form

$$\underline{\boldsymbol{x}}_j \leftarrow \Psi^{-1}\left( \frac{\boldsymbol{a}_j^T \, \Psi\left( [\boldsymbol{Y}^{(j)}]_+ \right)}{\boldsymbol{a}_j^T \, \Psi(\boldsymbol{a}_j)} \right), \quad \boldsymbol{a}_j \leftarrow \Psi^{-1}\left( \frac{\Psi\left( [\boldsymbol{Y}^{(j)}]_+ \right) \underline{\boldsymbol{x}}_j^T}{\Psi(\underline{\boldsymbol{x}}_j) \, \underline{\boldsymbol{x}}_j^T} \right),$$

[1] For $\alpha = 0$ instead of $\Phi(x) = x^\alpha$ we have used $\Phi(x) = \ln(x)$ [8].

**Algorithm 1 alpha-HALS NTF**

1: Initialize randomly all non-negative factors $\boldsymbol{U}^{(n)}$
2: Normalize all $\boldsymbol{u}_j^{(n)}$ for $n = 1, 2, ..., N-1$ to unit length
3: Compute residue tensor $\underline{\boldsymbol{R}} = \underline{\boldsymbol{Y}} - [\![\{\boldsymbol{U}\}]\!] = \underline{\boldsymbol{Y}} - \underline{\boldsymbol{Z}}$
4: **repeat**
5:  **for** $j = 1$ to $J$ **do**
6:  Compute $\underline{\boldsymbol{Y}}^{(j)} = \underline{\boldsymbol{R}} + [\![\boldsymbol{u}_j^{(1)}, \boldsymbol{u}_j^{(2)}, \ldots, \boldsymbol{u}_j^{(N)}]\!]$
7:  **for** $n = 1$ to $N$ **do**
8:  $\boldsymbol{u}_j^{(n)}$ as in (12)
9:  Normalize $\boldsymbol{u}_j^{(n)}$ to unit length vector if $n \neq N$
10:  **end for**
11:  Update $\underline{\boldsymbol{R}} = \underline{\boldsymbol{Y}}^{(j)} - [\![\boldsymbol{u}_j^{(1)}, \boldsymbol{u}_j^{(2)}, \ldots, \boldsymbol{u}_j^{(N)}]\!]$
12:  **end for**
13: **until** convergence criterion is reached

where $\Psi(\boldsymbol{x})$ is suitable chosen function, for example, $\Psi(\boldsymbol{x}) = \boldsymbol{x}^{\cdot\alpha}$, componentwise. [1]

In a similar way, for the $N$-order NTF problem (2), we have derived a novel learning rule referred here as the alpha HALS NTF learning rule [2]:

$$\boldsymbol{u}_j^{(n)} \leftarrow \Psi^{-1}\left(\frac{\Psi\left([\boldsymbol{Y}^{(j)}]_+\right) \,\overline{\times}_{-n}\, \{\boldsymbol{u}_j^T\}}{\prod_{l\neq n} \{\boldsymbol{u}_j^{(l)T}\, \Psi(\boldsymbol{u}_j^{(l)})\}}\right)_+, \tag{12}$$

where all nonlinear operations are performed componentwise and $\underline{\boldsymbol{Y}}^{(j)} = \underline{\boldsymbol{Y}} - \sum_{r\neq j} \boldsymbol{u}_r^{(1)} \circ \boldsymbol{u}_r^{(2)} \circ \cdots \circ \boldsymbol{u}_r^{(N)}$. Furthermore, from [7] we adopt the following shorthand notation for a multiplication of a tensor by a sequence of vectors in every mode except one:

$$\begin{aligned}\underline{\boldsymbol{Y}}^{(j)} \,\overline{\times}_{-n}\, \{\boldsymbol{u}_j^T\} = \underline{\boldsymbol{Y}}^{(j)} \overline{\times}_1 \boldsymbol{u}_j^{(1)T} \overline{\times}_2 \cdots \overline{\times}_{n-1} \boldsymbol{u}_j^{(n-1)T} \\ \overline{\times}_{n+1} \boldsymbol{u}_j^{(n+1)T} \overline{\times}_{n+2} \cdots \overline{\times}_N \boldsymbol{u}_j^{(N)T}.\end{aligned} \tag{13}$$

where $\overline{\times}_1$ means $n$-mode tensor by a vector multiplication.

## 3. FLEXIBLE HALS ALGORITHMS USING BETA DIVERGENCE

Beta divergence can be considered as a flexible and complementary cost function to the alpha divergence. In order to obtain local NMF algorithms we introduce the following definition of the beta-divergence (as used in [8, 4]):

$$D_\beta^{(j)}([\boldsymbol{Y}^{(j)}]_+ \,\|\, \boldsymbol{a}_j \underline{\boldsymbol{x}}_j) =$$

$$\begin{cases} \sum_{ik}\left(([y_{ik}^{(j)}]_+)\, \dfrac{[y_{ik}^{(j)}]_+^\beta - z_{ik}^\beta}{\beta} - \dfrac{[y_{ik}^{(j)}]_+^{\beta+1} - z_{ik}^{\beta+1}}{\beta+1}\right), & \beta > 0, \quad (14a) \\ \sum_{ik}\left(([y_{ik}^{(j)}]_+)\, \ln\left(\dfrac{[y_{ik}^{(j)}]_+}{z_{ik}}\right) - [y_{ik}^{(j)}]_+ + z_{ik}\right), & \beta{=}0, \quad (14b) \\ \sum_{ik}\left(\ln\left(\dfrac{z_{ik}}{[y_{ik}^{(j)}]_+}\right) + \dfrac{[y_{ik}^{(j)}]_+}{z_{ik}} - 1\right), & \beta{=}{-}1, \quad (14c) \end{cases}$$

where $[y_{ik}^{(j)}]_+ = \max\{0, y_{ik}^{(j)}\}$, $y_{ik}^{(j)} = y_{ik} - \sum_{r\neq j} a_{ir} x_{rk}$ and $z_{ik} = a_{ij} x_{jk}$ for $j = 1, 2, \ldots, J$. The choice of the $\beta$ parameter depends on the statistical distribution of data and the beta divergence corresponds to Tweedie models [4]. For example, if we consider the Maximum Likelihood (ML)approach (with no a priori assumptions) the optimal estimation consists in the minimization of the Beta Divergence measure when noise is Gaussian with $\beta = 1$, for the Gamma distribution is $\beta = -1$, for the Poisson distribution is $\beta = 0$, and for the compound Poisson $\beta \in (-1, 0)$. However, the ML estimation is not optimal in the sense of a Bayesian approach where a priori information of sources and mixing matrix (sparsity, non-negativity) can be imposed.

In order to derive a local learning algorithm, we compute the gradient of (14), with respect to elements to $x_{jk}$, $a_{ij}$:

$$\frac{\partial D_\beta^{(j)}}{\partial x_{jk}} = \sum_i \left(z_{ik}^\beta - ([y_{ik}^{(j)}]_+)\, z_{ik}^{\beta-1}\right) a_{ij}\,, \tag{15}$$

$$\frac{\partial D_\beta^{(j)}}{\partial a_{ij}} = \sum_k \left(z_{ik}^\beta - ([y_{ik}^{(j)}]_+) z_{ik}^{\beta-1}\right)\, x_{jk}\,. \tag{16}$$

By equating the gradient to zero, we obtain a new fixed point learning rule (referred here as the beta HALS NMF algorithm):

$$x_{jk} \leftarrow \frac{1}{\sum_{i=1}^I a_{ij}^{\beta+1}} \sum_{i=1}^I a_{ij}^\beta\, ([y_{ik}^{(j)}]_+)\,, \tag{17}$$

$$a_{ij} \leftarrow \frac{1}{\sum_{k=1}^K x_{jk}^{\beta+1}} \sum_{k=1}^K x_{jk}^\beta\, ([y_{jk}^{(j)}]_+)\,. \tag{18}$$

The above local HALS beta algorithm can be written in a generalized compact vector form as

$$\underline{\boldsymbol{x}}_j \leftarrow \frac{\Psi(\boldsymbol{a}_j^T)\, ([\boldsymbol{Y}^{(j)}]_+)}{\Psi(\boldsymbol{a}_j^T)\, \boldsymbol{a}_j}\,, \quad \boldsymbol{a}_j \leftarrow \frac{([\boldsymbol{Y}^{(j)}]_+)\, \Psi(\underline{\boldsymbol{x}}_j^T)}{\underline{\boldsymbol{x}}_j \Psi(\underline{\boldsymbol{x}}_j^T)}\,, \tag{19}$$

where $\Psi(\boldsymbol{x})$ is a suitably chosen convex function (e.g., $\Psi(\boldsymbol{x}) = \boldsymbol{x}^{\cdot\beta}$) and the nonlinear operations are performed elementwise.

[1] In practice, instead of half-wave rectifying we often use different transformations, e.g., real part of $\Psi(x)$ or adaptive nonnegative shrinkage function with gradually decreasing threshold till the noise threshold which is proportional to an estimate of the noise standard deviation.

[2] Due to limit of space we omit detailed derivation of all proposed algorithms.

**Algorithm 2 beta-HALS NTF**

1: Initialize randomly all non-negative factors $\boldsymbol{U}^{(n)}$
2: Normalize all $\boldsymbol{u}_j^{(n)}$ for $n = 1, 2, ..., N-1$ to unit length
3: Compute residue tensor $\underline{\boldsymbol{R}} = \underline{\boldsymbol{Y}} - [\![\{\boldsymbol{U}\}]\!] = \underline{\boldsymbol{Y}} - \underline{\boldsymbol{Z}}$
4: **repeat**
5: **for** $j = 1$ to $J$ **do**
6: Compute $\underline{\boldsymbol{Y}}^{(j)} = \underline{\boldsymbol{R}} + [\![\boldsymbol{u}_j^{(1)}, \boldsymbol{u}_j^{(2)}, \ldots, \boldsymbol{u}_j^{(N)}]\!]$
7: **for** $n = 1$ to $N-1$ **do**
8: $\boldsymbol{u}_j^{(n)} \Leftarrow \left[\underline{\boldsymbol{Y}}^{(j)} \,\overline{\times}_{-n}\, \{\Psi(\boldsymbol{u}_j^T)\}\right]_+$
9: Normalize $\boldsymbol{u}_j^{(n)}$ to unit length vector
10: **end for**
11: $\boldsymbol{u}_j^{(N)} \leftarrow \left[\frac{\underline{\boldsymbol{Y}}^{(j)} \,\overline{\times}_{-N}\, \{\Psi(\boldsymbol{u}_j^T)\}}{\prod_{l\neq N}\{\Psi(\boldsymbol{u}_j^{(l)})^T\, \boldsymbol{u}_j^{(l)}\}}\right]_+$
12: Update $\underline{\boldsymbol{R}} = \underline{\boldsymbol{Y}}^{(j)} - [\![\boldsymbol{u}_j^{(1)}, \boldsymbol{u}_j^{(2)}, \ldots, \boldsymbol{u}_j^{(N)}]\!]$
13: **end for**
14: **until** convergence criterion is reached

The above learning rule has been further generalized for the $N$-order NTF problem (2) as follows

$$\boldsymbol{u}_j^{(n)} \leftarrow \left[\frac{\underline{\boldsymbol{Y}}^{(j)} \,\overline{\times}_{-n}\, \{\Psi(\boldsymbol{u}_j^T)\}}{\prod_{l\neq n}\{\Psi(\boldsymbol{u}_j^{(l)})^T\, \boldsymbol{u}_j^{(l)}\}}\right]_+ . \tag{20}$$

Actually, the update rule (20) could be simplified to reduce computational cost by performing normalization of vectors $\boldsymbol{u}_j^{(n)}, (n = 1, \ldots, N-1)$ to unit length vectors after each iteration step.

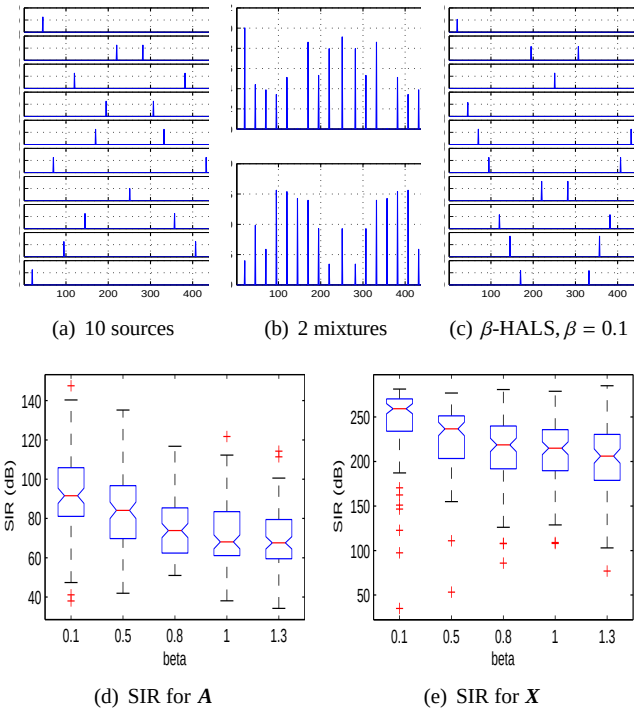
$$\boldsymbol{u}_j^{(n)} \leftarrow \left[\underline{\boldsymbol{Y}}^{(j)} \,\overline{\times}_{-n}\, \{\Psi(\boldsymbol{u}_j^T)\}\right]_+ , \tag{21}$$

$$\boldsymbol{u}_j^{(n)} \leftarrow \boldsymbol{u}_j^{(n)} / \|\boldsymbol{u}_j^{(n)}\|_2 . \tag{22}$$

The detailed pseudo-code of the beta HALS NTF algorithm is given below (Algorithm 2).

In order to avoid local minima we have developed also a simple heuristic hierarchical $\alpha$- $\beta$- HALS NTF algorithms combined with multi-start initializations using the ALS as follows:

1. Perform factorization of a tensor for any value of $\alpha$ or $\beta$ parameters (preferably, set the value of the parameter to 1 due to simplicity and high speed of algorithm for this value).
2. If the algorithm converged, but not achieved a desirable fit value (fit max), in order to overcome local minima we restart factorization but keep the previously estimated factors as the initial matrices for the ALS initialization (only one or two iterations).
3. If the algorithm does not converge change incrementally the value of the $\alpha$ or $\beta$ parameter. This step could help to jump over local minima.
4. Repeat procedure until we achieve a desired fit value or little or no change in the fit value or little or no change in the factor matrices, or a cost function value is at or near zero.




(a) 10 sources (b) 2 mixtures (c) $\beta$-HALS, $\beta = 0.1$

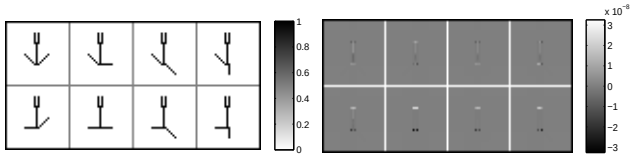
(d) SIR for $\boldsymbol{A}$ (e) SIR for $\boldsymbol{X}$

**Fig. 1**. Illustration of performance of the $\beta$-HALS NMF algorithm (a) 10 sparse sources assumed to be unknown, (b) two mixtures, (c) 10 estimated sources for $\beta = 0.1$. (d) & (e) SIR values for matrix $\boldsymbol{A}$ and sources $\boldsymbol{X}$ (respectively) obtained by the $\beta$-HALS NMF for $\beta = 0.1, 0.5, 0.8, 1, 1.3$ in the MC analysis of 100 trials.

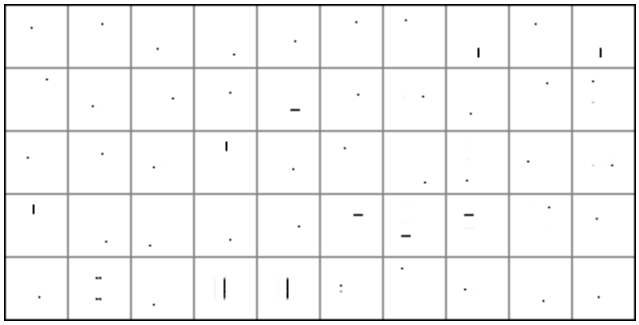
## 4. SIMULATION RESULTS

All algorithms presented in this paper have been tested for many difficult benchmarks with various statistical distributions and temporal structures of signals and images with additive noise on a 2.66 GHz Quad-Core Windows 64-bit PC with 8GB memory. For tensor factorization, results were compared with the following existing algorithms: the NMWF [6], the lsNTF [11] and also with two efficient implementations of PARAFAC ALS algorithm by Kolda and Bader [7] (denoted as ALS_K) and by Andersson and Bro [12] (denoted as ALS_B). To make a fair comparison we apply the same criteria and conditions: Maximum difference of fit value and we used three performance indexes: Peak Signal to Noise Ratio (PSNR) for all frontal slices, Signal to Interference Ratio (SIR) for each columns of factors and the explained variation ratio (i.e., how well the approximated tensor fit input data tensor) for a whole tensor. Due to space limitations we present here only four illustrative examples.

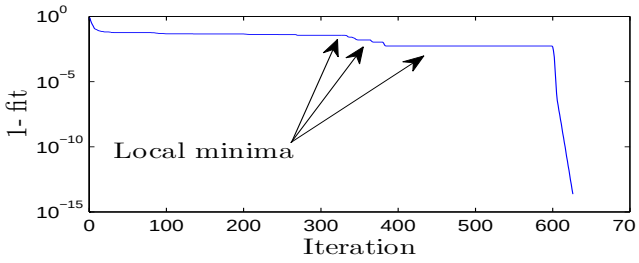
The results of **Example 1** presented in Fig.1 have been performed for the synthetic benchmark with sparse nonnegative 10 source components (Fig.1(a)). The sources have been mixed by the randomly generated full column rank



(a) 8 slices of `swimmer` (b) Residuum slices, $[-3:3\cdot 10^{-8}]$



(c) 50 basis components estimated by $\beta$-HALS NTF

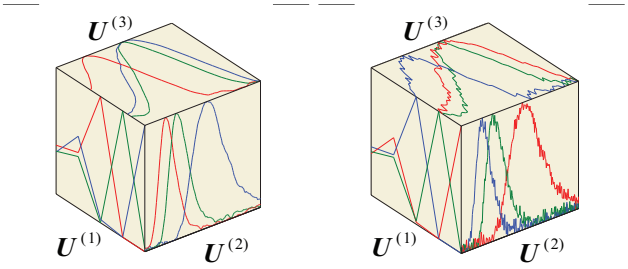



(d) Illustration of avoiding local minima with (logscale) graph 1-fit

**Fig. 2**. Illustration of perfect factorization of swimmer data set with 50 basis components of size $32 \times 32$.

matrix $\boldsymbol{A} \in \mathbb{R}_{+}^{2\times 10}$, so only two mixed signals are available (Fig.1(b)). The performance obtained with the new beta NMF HALS algorithm (8)-(11) for different values of the parameter $\beta$ are illustrated in Fig.1(d) and 1(e) with average Signal-to-Interference (SIR) level greater than 30 [dB]. Since the proposed algorithms (alternating techniques) perform a non-convex optimization, the estimated components are initial condition dependent. To estimate the performance in a statistical sense, we have performed a Monte Carlo (MC) analysis. Figures 1(d) and 1(e) present the histograms of 100 mean-$SIR$ samples for estimations of $\boldsymbol{A}$ and $\boldsymbol{X}$.

In **Example 2**, we used a difficult benchmark `swimmer` dataset containing black-and-while stick figures satisfying the so called separable factorial articulation criteria. Each figure consists of a "torso" of 12 pixels in the center and four "limbs" of six pixels that can be in any one of four positions. With limbs in all possible positions, there are a total of 256 figures of dimension $32 \times 32$ pixels. Objective is to perform a sparse decomposition of images into basic parts [3]. Note that almost all standard NMF algorithms failed with this data set due to strong overlapping



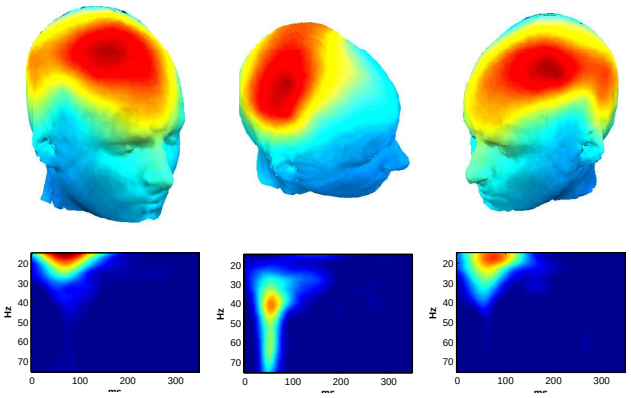

(a) Smoothed factors by $\beta$-HALS NTF (b) Factors by NMWF

**Fig. 3**. Illustration of estimated factors by the $\beta$-HALS NTF algorithm ($\beta = 1.2$) (a) in comparison to the NMWF algorithm (b) for three-way amino acid data.

patterns presenting in all slices. However, the HALS NTF algorithms can successfully decompose such images taking into account spatial information. The swimmer tensor was factorized into 50 rank-one tensors and returned the absolutely perfect results with 100% of the explained variation (Fit) by both $\alpha$- and $\beta$- HALS NTF algorithms with initial values $\alpha = 1$ and $\beta = 1$. The perturbation value of $\alpha$ and $\beta$ was set at 0.98, and the nonnegative ALS initialization was also used to avoid local minima. Fig.2(d) illustrates this technique over the 1-fit line. Fig.2(a) illustrates the first 8 images for this data set; whereas their residuum slices of the originals and their estimated slices by $\beta$-HALS NTF are shown in Fig.2(b), and 50 estimated very sparse basis components are shown in Fig.2(c). With the same dataset, the NMWF and the lsNTF algorithms achieved only 91.46%, 96.88% of Fit values, respectively. This means non-perfect reconstruction of original components.

In **Example 3**, we decomposed a real-world data: Amino acid fluorescence data from five samples containing tryptophan, phenylalanine, and tyrosine (`claus.mat`) [13] which were corrupted by Gaussian noise with SNR = 0 dB before reconstruction using $J = 3$ rank-one tensors. The $\beta$-HALS NTF was selected with $\beta = 1.2$, where for $\alpha$-HALS NTF we select $\alpha = 0.9$. All algorithms were set to process data with the same number of iterations (100 times). In this example, we applied smooth constraint for $\alpha$- and $\beta$- HALS NTF. Based on fit ratio (1), HALS algorithms returned better results than the existing algorithms. In comparison, HALS NTF's estimated factors (Fig.3(a)) are very smooth and explain 99.39% of the clean tensor; while the factors of the NMWF, lsNTF, ALS_K, ALS_B algorithms are dominated by noise, illustrated in Fig.3(b).

In **Example 4** we used real EEG data: `tutorialdataset2.zip` [14] which was pre-processed by complex Morlet wavelet. Tensor is represented by the inter-trial phase coherence (ITPC) for 14 subjects during a proprioceptive pull of left and right hand (28 files) with size $64 \times 4392 \times 28$. Exemplary results are shown in Fig.4 with scalp topo-



(a) Left hand stimuli (b) Gamma activity of both stimuli (c) Right hand stimuli

**Fig. 4**. EEG analysis using the $\beta$-HALS NTF for Example 4 with factor $\boldsymbol{U}^{(1)}$ for a scalp topographic map (first row), factor $\boldsymbol{U}^{(2)}$ for spectral (time-frequency) map (second row). Results are consistent with previous analysis [14] but run time is almost 8 times shorter and fit is slightly better.

**Table 1**. Comparison of Performance of NTF Algorithms for Examples 3-5

| | **Fit** (%) | | | **Time** (second) | |
|---|---|---|---|---|---|
| **Example No.** | 3 | 4 | 5 | 4 | 5 |
| $\alpha$**-NTF** | **100** | **99.44** | 52.33 | 4.52 | 13.02 |
| $\beta$**-NTF** | **100** | 99.39 | **52.33** | **1.74** | **7.08** |
| **NMWF** [1] | 91.46 | 98.76 | 52.31 | 3.16 | 58.19 |
| **lsNTF** | 96.88 | 98.06 | 51.33 | 3.30 | 4029.84 |
| **ALS_B** | | 98.53 | 53.17 | 2.52 | 67.24 |
| **ALS_K** | | 98.53 | 53.17 | 1.78 | 66.39 |

graphic maps and their corresponding IPTC time-frequency measurements and performance comparisons are given in Table 1. The components of the first factor $\boldsymbol{U}^{(1)}$ are relative to location of electrodes, and they are used to illustrate the scalp topographic maps (the first row in Fig.4); whereas the 2-nd factor $\boldsymbol{U}^{(2)}$ represents frequency-time spectral maps which were vectorized, presented in the second row. Each component of these factors corresponds to specific stimulus (left, right and the both hands actions).

Computer simulation for these data confirmed that the proposed algorithms give similar results to that obtained using the known "state of the arts" NTF algorithms, but our algorithms seems to be slightly faster and more efficient for this data set (see Table 1).

[1] In fact, the NMWF failed for very noisy data due to large negative entries. We enforced the estimated components to have non-negative values by half-wave rectifying.

## 5. CONCLUSIONS AND DISCUSSION

The main objective and motivations of this paper is to derive local NMF/NTF algorithms which are suitable both for under-determined (over-complete) and over-determined cases. We have applied novel cost (loss) functions that allow us to derive a family of flexible and efficient NMF and NTF algorithms, where sources may have different temporal structures and distributions and/or different sparsity profiles.

This is the unique extension of standard NMF/NTF algorithms, and to the authors' best knowledge, the first time such algorithms have been applied to multi-way NTF/NMF models.

We have implemented the discussed algorithms in MATLAB in the toolboxes NMFLAB/NTFLAB and they will be available soon free for researchers. The performance of the proposed algorithms are compared with the ordinary NMF and also FOCUSS+ algorithms. The proposed algorithms are shown to be superior in terms of the performance, speed and convergence properties. The approach can be extended for other applications such as dynamic MRI imaging and it can be used as an alternative or improved reconstruction method to: k-t BLAST, k-t SENSE or k-t SPARSE, because our approach relaxes the problem of getting stuck to in local minima and provides usually better performance than the standard FOCUSS algorithms.

## 6. REFERENCES


[1] A. Cichocki, R. Zdunek, and S. Amari, "Hierarchical als algorithms for nonnegative matrix and 3d tensor factorization," in *Lecture Notes in Computer Science*, 2007, vol. 4666, pp. 169–176.

[2] M. Berry, M. Browne, A. Langville, P. Pauca, and R. Plemmons, "Algorithms and applications for approximate nonnegative matrix factorization," *Computational Statistics and Data Analysis*, vol. 52, no. 1, pp. 155–173, 2007.

[3] T. Hazan, S. Polak, and A. Shashua, "Sparse image coding using a 3D non-negative tensor factorization," in *International Conference of Computer Vision (ICCV)*, 2005, pp. 50–57.

[4] A. Cichocki, R. Zdunek, S. Choi, R. Plemmons, and S. Amari, "Non-negative tensor factorization using Alpha and Beta divergencies," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP07)*, Honolulu, Hawaii, USA, April 15–20 2007, vol. III, pp. 1393–1396.

[5] A. Smilde, R. Bro, and P. Geladi, *Multi-way Analysis: Applications in the Chemical Sciences*, John Wiley and Sons, New York, 2004.

[6] M. Mørup, L. K. Hansen, J. Parnas, and S. M. Arnfred, "Decomposing the time-frequency representation of EEG using non-negative matrix and multi-way factorization," Tech. Rep., 2006.

[7] T. G. Kolda and B.W. Bader, "Tensor decompositions and applications," *SIAM Review*, June 2008.

[8] A. Cichocki, S. Amari, R. Zdunek, R. Kompass, G. Hori, and Z. He, "Extended SMART algorithms for non-negative matrix factorization," *Springer LNAI*, vol. 4029, pp. 548–562, 2006.

[9] S. Amari, *Differential-Geometrical Methods in Statistics*, Springer Verlag, 1985.

[10] A. Cichocki, A-H. Phan, R. Zdunek, and L.-Q. Zhang, "Flexible component analysis for sparse, smooth, nonnegative coding or representation," in *Lecture Notes in Computer Science*. 2008, vol. 4984, pp. 811–820, Springer.

[11] M. P. Friedlander and K. Hatz, "Computing nonnegative tensor factorizations," Tech. Rep. TR-200621, Dept. Computer Science, University of British Columbia, Vancouver, December 2007, To appear in *Optimization Methods and Software*.

[12] C. A. Andersson and R. Bro, "The *N*-way Toolbox for MATLAB," *Chemometrics Intell. Lab. Systems*, vol. 52, pp. 1–4, 2000.

[13] R. Bro, "PARAFAC. Tutorial and applications," in *Special Issue 2nd Internet Conf. in Chemometrics (INCINC'96)*. 1997, vol. 38, pp. 149–171, Chemom. Intell. Lab. Syst.

[14] M. Mørup, L. K. Hansen, and S. M. Arnfred, "ERPWAVELAB a toolbox for multi-channel analysis of time-frequency transformed event related potentials," *Journal of Neuroscience Methods*, vol. 161, pp. 361–368, 2007.