BULLETIN OF THE POLISH ACADEMY OF SCIENCES
TECHNICAL SCIENCES, Vol. 60, No. 3, 2012
DOI: 10.2478/v10175-012-0051-4

Fast and unique Tucker decompositions via multiway blind source separation

G. ZHOU[1,2] and A. CICHOCKI[3,4,5]*

[1] Laboratory for Advanced Brain Signal Processing, RIKEN, Brain Science Institute, Wako-shi, Saitama 3510198, Japan
[2] School of Electronic and Information Engineering, South China University of Technology, Guangzhou 510641, China
[3] RIKEN Brain Science Institute, 2-1 Hirosawa Wako City, Saitarna 351-0198, Japan
[4] Electrotechnical Faculty, Warsaw University of Technology, 1 Politechniki Sq., 00-661 Warszawa, Poland
[5] System Research Institute, 6 Newelska St., 01-447 Warszawa, Poland

Abstract. A multiway blind source separation (MBSS) method is developed to decompose large-scale tensor (multiway array) data. Benefitting from all kinds of well-established constrained low-rank matrix factorization methods, MBSS is quite flexible and able to extract unique and interpretable components with physical meaning. The multilinear structure of Tucker and the essential uniqueness of BSS methods allow MBSS to estimate each component matrix separately from an unfolding matrix in each mode. Consequently, alternating least squares (ALS) iterations, which are considered as the *workhorse* for tensor decompositions, can be avoided and various robust and efficient dimensionality reduction methods can be easily incorporated to pre-process the data, which makes MBSS extremely fast, especially for large-scale problems. Identification and uniqueness conditions are also discussed. Two practical issues dimensionality reduction and estimation of number of components are also addressed based on sparse and random fibers sampling. Extensive simulations confirmed the validity, flexibility, and high efficiency of the proposed method. We also demonstrated by simulations that the MBSS approach can successfully extract desired components while most existing algorithms may fail for ill-conditioned and large-scale problems.

Key words: Multiway Blind Source Separation (MBSS), Multilinear Independent Component Analysis (MICA), Constrained tensor decompositions, Tucker models, Nonnegative Tucker Decomposition (NTD).

1. Introduction and problem statement

How to find informative and sparse/compact representations of massive experimental or measured multidimensional large-scale tensor data is a fundamental and challenging problem in data mining and data analysis. Although the basic models for tensor (i.e., multiway array) decompositions such as Canonical Polyadic (CP) and Tucker decomposition models were proposed a long time ago [1–4], they have only recently emerged as promising tools for exploratory analysis of multidimensional data in diverse applications, especially in dimensionality reduction, feature extraction, Independent Component Analysis (ICA), classification, prediction, multiway clustering, and data mining [4, 5]. By virtue of their multiway nature, tensors provide powerful tools for analysis and fusion of large-scale, multi-modal, massive data together with a mathematical backbone for the discovery of underlying hidden complex data structures [4, 7–9]. From the data analysis point of view, tensor decompositions are very attractive because they take into account spatial, temporal and spectral information, and provide links among the various extracted factors or latent variables with physical or physiological meanings and interpretations [5, 8, 10]. For example, tensor representations and decompositions allow us to investigate temporal, spatial and spectral independent components and links among them. Moreover, tensor decompositions are emerging techniques for data fusion, pattern recognition, object detection, classification, multiway clustering, Blind Source Separation (BSS), sparse representation and coding [11–17], etc.

Most of the existing tensor decomposition methods are focused on the minimum fitting error to the data. However, quite different from the matrix case, the optimal low-rank approximation may not exist at all [18] or, if it exists, may not be unique for high-order tensors [7, 19, 20]. Particularly, for Tucker decompositions the results are always non-unique due to rotational freedom. As a result the extracted factors often lack of physical meaning and are hard to interpret. To overcome this drawback, constrained tensor decompositions have received increasing interest in recent years. In this regard, several authors have proposed applying ICA to the CP and Tucker models, i.e., impose statistical independence in at least one mode [21–24]. For example, in the methods proposed in [21] and [22] ICA was combined with the CP model. However, the CP model and its associated algorithms are often too restrictive as the number of components in each mode is the same and there are no mutual interactions between components in different modes. Very recently Unkel *et al.* (2011) proposed a two-step method based on the Tucker-3 model in which ordinary Tucker decomposition is performed first and then statistical independence is imposed to refine the components in only one mode by exploiting the rotational freedom of the Tucker model [24]. Furthermore, Vasilescu and Terzopoulos (2005) proposed a multilinear ICA method which is also a two-step method but independence can be imposed on either all the temporal components or the mixing matrix (spatial components) [23]. However, this approach has not been investigated deeply and only ICA is considered.

*e-mail: cia@brain.riken.jp

On the other hand, in the (2D) matrix case, very efficient low-rank constrained matrix factorization techniques (also called penalized matrix factorizations) have been developed, such as principal component analysis (PCA), ICA [25–28], sparse component analysis (SCA) [29, 30], smooth component analysis (SmoCA) [5] and nonnegative matrix factorization (NMF) [5, 31], just to name a few. These matrix factorization techniques have their own bias, advantages, and are widely applied to blind source separation (BSS), dimensionality reduction, data compression and feature extraction, by exploiting various assumptions and *a priori* knowledge. We do not intend to duplicate these works which have been done for matrices. Instead, we would like to show that such well-established matrix factorization methods (especially, NMF, SCA and ICA) can be extended to tensor scenarios directly and uniformly. Motivated by efficiency and high level of flexibility of BSS and various Component Analysis methods, we investigate in this paper a multiway BSS (MBSS) approach to perform constrained tensor decompositions with various constraints and diversities[1], in order to provide more interpretable, essentially unique[2] components with physical meanings. To the best of our knowledge, till now there was no available systematic investigation of the validity and performance of this approach which could incorporate any powerful and flexible BSS techniques (beyond ICA[3]) to tensor decompositions. Finally, the MBSS method can be viewed as a generalization and extension of the wide variety of BSS techniques and algorithms to multiway data.

The remainder of the paper is organized as follows. In Sec. 2 basic models and concepts for Tucker decomposition are briefly introduced. In Sec. 3 the flexible and general scheme of Multiway BSS is developed, as well as the discussion on identifiability and uniqueness conditions are provided. In Sec. 4 a method for dimensionality reduction and estimation of the number of components is addressed. Finally, some simulation results are presented in Sec. 5 and conclusions are made in Sec. 6.

Basic notations. Tensors are denoted by underlined capital boldface letters, e.g., $\underline{\mathbf{Y}} \in \mathbb{R}^{I_1\times I_2\times\cdots\times I_N}$. The order of a tensor is the number of modes, also known as ways or dimensions (e.g., space, time, frequency, subjects, trials, classes, groups, and conditions). In contrast, matrices (two-way tensors) are denoted by boldface capital letters, e.g., $\mathbf{Y}$; vectors (one-way tensors) are denoted by boldface lowercase letters, e.g., the columns of a matrix $\mathbf{A}$ are denoted by $\mathbf{a}_j$, and scalars are denoted by lowercase letters, e.g., a_{ij}.

The mode-n product $\underline{\mathbf{Y}} = \underline{\mathbf{G}} \times_n \mathbf{A}$ of a tensor $\underline{\mathbf{G}} \in \mathbb{R}^{J_1\times J_2\times\cdots\times J_N}$ and a matrix $\mathbf{A} = [a_{i,j_n}] \in \mathbb{R}^{I\times J_n}$ is a tensor $\underline{\mathbf{Y}} \in \mathbb{R}^{J_1\times\cdots\times J_{n-1}\times I\times J_{n+1}\times\cdots\times J_N}$, with elements $y_{j_1,j_2,\ldots,j_{n-1},i,j_{n+1},\ldots,j_N} = \sum_{j_n=1}^{J_n}(g_{j_1,j_2,\ldots,j_N})(a_{i,j_n})$. The symbol $\otimes$ denotes the Kronecker product, i.e., $\mathbf{A}\otimes\mathbf{B} = [a_{ij}\mathbf{B}]$, and the symbol $\odot$ denotes the Khatri-Rao product or column-wise Kronecker product, i.e., $\mathbf{A}\odot\mathbf{B} = [\mathbf{a}_1\otimes\mathbf{b}_1\cdots\mathbf{a}_J\otimes\mathbf{b}_J]$. Unfolding (matricization, flattening) of a tensor $\underline{\mathbf{Y}} \in \mathbb{R}^{I_1\times I_2\times\cdots\times I_N}$ in mode-n is denoted as $\mathbf{Y}_{(n)} \in \mathbb{R}^{I_n\times\Pi_{p\neq n}I_p}$, which consists of arranging all possible mode-n tubes (vectors) as the columns of the unfolded matrix [4]. For simplicity, we define $\breve{I}_n = \Pi_{p\neq n}I_p$, $\breve{J}_n = \Pi_{p\neq n}J_p$, $\bigotimes_{p\neq n}\mathbf{A}^{(p)} = \mathbf{A}^{(N)}\otimes\cdots\mathbf{A}^{(n+1)}\otimes\mathbf{A}^{(n-1)}\cdots\otimes\mathbf{A}^{(1)}$ and $\bigodot_{p\neq n}\mathbf{A}^{(p)} = \mathbf{A}^{(N)}\odot\cdots\mathbf{A}^{(n+1)}\odot\mathbf{A}^{(n-1)}\cdots\odot\mathbf{A}^{(1)}$. Readers are referred to [4, 5] for more details about the notations and tensor operations.

2. Tucker decomposition models

Tucker decomposition has been received intensive study in recent years as one of the most important and flexible tensor decomposition models. In Tucker decompositions, the data are decomposed as the product of a core tensor with N mode component matrices [2] (see Fig. 1), i.e., a given data tensor $\underline{\mathbf{Y}} \in \mathbb{R}^{I_1\times I_2\times\cdots\times I_N}$ is decomposed as

$$\begin{aligned}\underline{\mathbf{Y}} &= \underline{\mathbf{G}} \times_1 \mathbf{A}^{(1)} \times_2 \mathbf{A}^{(2)} \cdots \times_N \mathbf{A}^{(N)} + \underline{\mathbf{E}} = \underline{\widehat{\mathbf{Y}}} + \underline{\mathbf{E}} \\ &= \sum_{j_1=1}^{J_1}\cdots\sum_{j_N=1}^{J_N} g_{j_1 j_2\cdots j_N}\left(\mathbf{a}^{(1)}_{j_1}\circ\mathbf{a}^{(2)}_{j_2}\cdots\circ\mathbf{a}^{(N)}_{j_N}\right) + \underline{\mathbf{E}},\end{aligned} \tag{1}$$

where $\underline{\mathbf{G}} \in \mathbb{R}^{J_1\times J_2\times\cdots\times J_N}$ is the core tensor, $\mathbf{A}^{(n)} = [\mathbf{a}^{(n)}_1, \mathbf{a}^{(n)}_2, \ldots, \mathbf{a}^{(n)}_{J_n}] \in \mathbb{R}^{I_n\times J_n}$ is the mode-n component matrix for $n = 1, 2, \cdots, N$, and the tensor $\underline{\mathbf{E}} \in \mathbb{R}^{I_1\times I_2\times\cdots\times I_N}$ represents errors or noise. We may also use a shorter notation for (1) in the form $\underline{\widehat{\mathbf{Y}}} = [\![\underline{\mathbf{G}}; \mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \ldots, \mathbf{A}^{(N)}]\!]$ for simplicity [4].

To extract the latent factors $\mathbf{A}^{(n)}$ from $\underline{\mathbf{Y}}$, a Tucker-1 model is often useful

$$\underline{\mathbf{Y}} \approx \underline{\mathbf{G}}^{(-n)} \times_n \mathbf{A}^{(n)}, \tag{2}$$

where

$$\begin{aligned}\underline{\mathbf{G}}^{(-n)} = \underline{\mathbf{G}} &\times_1 \mathbf{A}^{(1)} \cdots \times_{n-1} \mathbf{A}^{(n-1)} \\ &\times_{n+1} \mathbf{A}^{(n+1)} \cdots \times_N \mathbf{A}^{(N)},\end{aligned}$$

or by using unfolding operations and matrix representation

$$\mathbf{Y}_{(n)} \approx \mathbf{A}^{(n)}\mathbf{G}^{(-n)}_{(n)} = \mathbf{A}^{(n)}[\mathbf{B}^{(n)}]^T, \tag{3}$$

where

$$\mathbf{G}^{(-n)}_{(n)} = \mathbf{G}_{(n)}\left[\bigotimes_{p\neq n}\mathbf{A}^{(p)}\right]^T \overset{\text{def}}{=} [\mathbf{B}^{(n)}]^T. \tag{4}$$

In other words, $\mathbf{A}^{(n)}$ and $\mathbf{G}_{(n)}$ are solutions to the following least-square problem[4]:

$$\min \left\|\mathbf{Y}_{(n)} - \mathbf{A}^{(n)}\mathbf{B}^{(n)T}\right\|^2_F, \qquad (n = 1, 2, \ldots, N). \tag{5}$$

[1]By diversity, we mean different characteristics, features or morphology of source signals or hidden latent variables [27].

[2]By essentially unique, we understand a unique decomposition with arbitrary scaling and permutation of components.

[3]By "beyond ICA", we understand that any BSS method can be applied, e.g., PCA/SVD, NMF, SCA.

[4]Alternatively, we can solve the least-square problem $\min\|\mathbf{Y}^{(-n)}_{(n)} - \mathbf{A}^{(n)}\mathbf{G}_{(n)}\|^2_F$ instead, where $\mathbf{Y}^{(-n)}_{(n)}$ is the mode-n unfolding of tensor $\underline{\mathbf{Y}}^{(-n)} = \underline{\mathbf{Y}} \times_1 [\mathbf{A}^{(1)\dagger}] \cdots \times_{n-1} [\mathbf{A}^{(n-1)\dagger}] \times_{n+1} [\mathbf{A}^{(n+1)\dagger}] \cdots \times_N [\mathbf{A}^{(N)\dagger}]$.

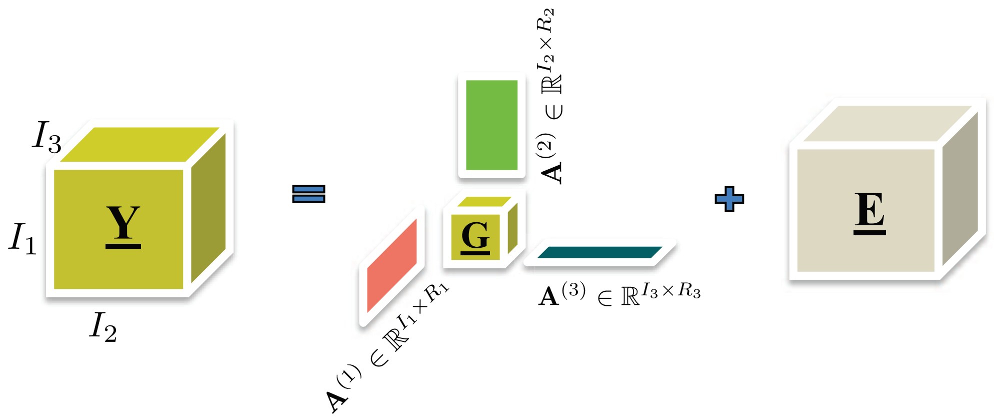

I_3
I_1
$\underline{\mathbf{Y}}$
I_2
$=$
$\mathbf{A}^{(2)} \in \mathbb{R}^{I_2\times R_2}$
$\underline{\mathbf{G}}$
$\mathbf{A}^{(1)} \in \mathbb{R}^{I_1\times R_1}$
$\mathbf{A}^{(3)} \in \mathbb{R}^{I_3\times R_3}$
$+$
$\underline{\mathbf{E}}$

Fig. 1. Illustration of a 3-way tensor decomposition using the Tucker-3 model. The objective is to estimate the components $\mathbf{a}_{j_n}^{(n)}$, i.e., columns of the component matrices $\mathbf{A}^{(n)} = [\mathbf{a}_1^{(n)}, \mathbf{a}_2^{(n)}, \dots, \mathbf{a}_{J_n}^{(n)}] \in \mathbb{R}^{I_n\times J_n}$, with desired diversities or statistical properties, and a possibly sparse core tensor $\underline{\mathbf{G}} \in \mathbb{R}^{J_1\times J_2\times J_3}$, typically with $J_n \ll I_n$, $(n = 1,2,3)$. Instead of applying the standard Alternating Least Squares (ALS) algorithms to the Tucker-3 model, we can apply the unfolding of the data tensor according to the Tucker-1 models and then perform constrained matrix factorizations for the unfolded matrices (multiway BSS) by imposing desired constraints (e.g., nonnegativity, sparseness, statistical independence, smoothness or decorrelation, etc)

Hence, $\mathbf{A}^{(n)}$ and $\underline{\mathbf{G}}$ can be updated sequentially by freezing all set of matrices except one

$$\begin{aligned}\mathbf{A}^{(n)} &\leftarrow \mathbf{Y}_{(n)}[\mathbf{B}^{(n)}]^{\dagger}, \qquad (n = 1,2,\dots,N),\\ \underline{\mathbf{G}} &\leftarrow \underline{\mathbf{Y}} \times_1 \mathbf{A}^{(1)\dagger} \times_2 \mathbf{A}^{(2)\dagger} \cdots \times_N \mathbf{A}^{(N)\dagger},\end{aligned} \tag{6}$$

where † denotes the Moore-Penrose pseudo inverse of matrices. To achieve the optimal fitting error, the above steps are repeated till convergence. These update rules are often referred to as alternating least-square (ALS) method.

From the above analysis, standard ALS methods involve frequently unfolding operations and matrix-matrix Kronecker/Khatri-Rao products. For example, the matrix $\mathbf{B}^{(n)} \in \mathbb{R}^{I_n\times \breve{I}_n}$ having a Kronecker structure in (4) is huge for large-scale problems. This makes the ALS methods quite time and memory consuming, and therefore not suitable for large-scale data.

3. Constrained Tucker decomposition using Multiway BSS

3.1. Motivation. The Tucker decomposition attempts to give optimal low-rank $(\{J_1, J_2, \dots, J_N\})$ approximations of the original data. However, quite different from the matrix case, the optimal low-rank approximation may not exist at all or, if it exists, may not be unique for high-order tensors [7, 18–20]. Particularly, unconstrained Tucker decompositions are always non-unique since

$$\begin{aligned}&\underline{\mathbf{Y}} \approx \underline{\mathbf{G}} \times_1 \mathbf{A}^{(1)} \times_2 \mathbf{A}^{(2)} \cdots \times_N \mathbf{A}^{(N)}\\ &= [\underline{\mathbf{G}} \times_1 \mathbf{Q}^{(1)\dagger} \times_2 \mathbf{Q}^{(2)\dagger} \cdots \times_N \mathbf{Q}^{(N)\dagger}]\\ &\quad \times_1(\mathbf{A}^{(1)}\mathbf{Q}^{(1)}) \cdots \times_N (\mathbf{A}^{(N)}\mathbf{Q}^{(N)}),\end{aligned} \tag{7}$$

where $\mathbf{Q}^{(n)} \in \mathbb{R}^{J_n\times J_n}$ is any nonsingular matrix. So the component matrices in unconstrained Tucker decompositions are with many degrees of freedom and usually have no specific physical meaning or interpretation.

Based on these facts, it is reasonable to consider constrained tensor decompositions, which achieves not necessarily the best fit but the most meaningful and featured components instead. Generally, a constrained Tucker decomposition problem can be formulated as minimization of the cost function:

$$\begin{aligned}&D_F\left(\underline{\mathbf{Y}} \| \widehat{\underline{\mathbf{Y}}}\right) =\\ &\|\underline{\mathbf{Y}} - \underline{\mathbf{G}} \times_1 \mathbf{A}^{(1)} \times_2 \mathbf{A}^{(2)} \cdots \times_N \mathbf{A}^{(N)}\|_F^2\\ &+\sum_{n=1}^{N} \alpha_n H_n(\mathbf{A}^{(n)}),\end{aligned} \tag{8}$$

where $\|\underline{\mathbf{Y}}\|_F = (\sum_{i_1=1}^{I_1} \cdots \sum_{i_N=1}^{I_N} y_{i_1\cdots i_N}^2)^{1/2}$, $\alpha_n \geq 0$ are penalty coefficients and $H_n(\mathbf{A}^{(n)})$ are penalty terms which are added to achieve specific properties of the components. For example, if we need to impose mutual independence constraints the penalty terms can take the form $H_n(\mathbf{A}^{(n)}) = \sum_{j=1}^{J_n} \sum_{p\neq j} \mathbf{a}_p^{(n)T} \phi_n(\mathbf{a}_j^{(n)})$, where $\phi_n(x)$ are suitable nonlinear functions.

In principle, model (8) leads to a penalized or constrained ALS algorithms which allows us to find component matrices $\mathbf{A}^{(n)}$ and the associated core tensor $\underline{\mathbf{G}}$, as the ALS algorithms are considered as basic “workhorses” for tensor decompositions. However, as mentioned above, ALS iteration based algorithms have high computational complexity, and due to constraints, they may suffer often from slow convergence. For example, existing nonnegative Tucker decomposition methods often converge very slowly. Particularly, even if we optimize (8) in a global way, the ALS algorithms may stuck in local minima due to noncovexity of cost functions.

3.2. Multiway BSS. In this section, we investigate a simpler yet much more efficient and flexible approach by exploiting separately each mode-n unfolding matrix $\mathbf{Y}_{(n)}$ of the data tensor $\underline{\mathbf{Y}}$, according to the Tucker-1 decompositions (3), which allows us to directly apply suitable constrained matrix factorization methods to Tucker decompositions. Let $\underline{\mathbf{Y}} \approx [\![\underline{\mathbf{G}}; \mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \ldots, \mathbf{A}^{(N)}]\!]$. We have

$$\mathbf{Y}_{(n)} \approx \mathbf{A}^{(n)}\mathbf{B}^{(n)T}, \qquad (n = 1, 2, \ldots, N), \tag{9}$$

where the mixing matrices have a special Kronecker structure $\mathbf{B}^{(n)} = [\mathbf{G}_{(n)}^{(-n)}]^T$ defined by (4).

From (9), we note that the columns of the mode-n matricization of $\underline{\mathbf{Y}}$ are just linear mixtures of the columns of $\mathbf{A}^{(n)}$, $(n = 1, 2, \ldots, N)$. This suggests that we can use various BSS algorithms to directly extract the component matrices with specific properties and diversities, without consideration of the special Kronecker structure of the basis matrices $\mathbf{B}^{(n)}$ due to the essential uniqueness of BSS.

Assume that the component matrices of interest can be separated by a standard BSS algorithm with unavoidable scaling and permutation ambiguities:

$$\widehat{\mathbf{A}}^{(n)} = \Psi_n(\mathbf{Y}_{(n)}) = \Psi_n(\mathbf{A}^{(n)}\mathbf{B}^{(n)T}) = \mathbf{A}^{(n)}\mathbf{P}_n\mathbf{D}_n, \quad (n = 1, 2, \ldots, N), \tag{10}$$

where Ψ_n denotes symbolically a specific BSS method, the subindex n indicates the fact that for each mode different method and criteria can be employed, and for each mode we have different scaling $\mathbf{D}_n$ and permutation $\mathbf{P}_n$ ambiguity. From (10), for each mode different constraints can be imposed depending on the expected or known physical properties of the components. This is also one major difference between MBSS and the existing multilinear ICA algorithms where only ICA criteria are considered.

We have two basic ways to implement constrained Tucker decomposition in practice:

Independent Extraction of Factor Matrices. In this case each component matrix $\mathbf{A}^{(n)}$ is estimated from the mode-n matricization of $\underline{\mathbf{Y}}$ independently and separately by using (10). Once all desired component matrices $\mathbf{A}^{(n)}$ $(n = 1, 2, \ldots, N)$ have been estimated, the core tensor can be computed, for example, from:

$$\widehat{\underline{\mathbf{G}}} = \underline{\mathbf{Y}} \times_1 \mathbf{A}^{(1)\dagger} \times_2 \mathbf{A}^{(2)\dagger} \cdots \times_N \mathbf{A}^{(N)\dagger}. \tag{11}$$

Alternatively, we can apply multiplicative update formula proposed in [5, 16], e.g., if we wish to impose nonnegativity constraints on the components and the core tensor[5].

Sometimes only partial pre-specified factors, say $\mathbf{A}^{(K)}$, $K < N$, can be extracted by using BSS. For the Tucker decomposition, the remainder component matrices and the core tensor can be obtained, e.g., by using ordinary ALS iteration based methods, such as HOOI [7, 32]. In most cases, however, the remainder component matrices are simply of no interest because they do not carry any important information, or the information they carry can be simply absorbed into the core tensor. This often leads to a partial Tucker decomposition. Without loss of generality, let us assume that we are interested in extracting only the first K, with $K \leq N$ component matrices. In such a case, we can use a simplified Tucker-K model described as

$$\underline{\mathbf{Y}} = \breve{\underline{\mathbf{G}}} \times_1 \mathbf{A}^{(1)} \times_2 \mathbf{A}^{(2)} \cdots \times_K \mathbf{A}^{(K)} + \underline{\mathbf{E}}, \tag{12}$$

where the partial core tensor $\breve{\underline{\mathbf{G}}} \in \mathbb{R}^{J_1 \cdots \times J_K \times I_{K+1} \cdots \times I_N}$ is expressed as

$$\breve{\underline{\mathbf{G}}} = \underline{\mathbf{G}} \times_{K+1} \mathbf{A}^{(K+1)} \times_{K+2} \mathbf{A}^{(K+2)} \cdots \times_N \mathbf{A}^{(N)}. \tag{13}$$

Note that the Tucker-K model (12) can be represented equivalently by a set of K different matrix factorizations with three factors:

$$\mathbf{Y}_{(k)} \approx \mathbf{A}^{(k)}\, \breve{\mathbf{G}}_{(k)}\, \mathbf{Z}^{(k)}, \qquad (k = 1, 2, \ldots, K), \tag{14}$$

where $\mathbf{Z}^{(k)} = \left[\mathbf{A}^{(K)} \otimes \cdots \otimes \mathbf{A}^{(k+1)} \otimes \mathbf{A}^{(k-1)} \cdots \otimes \mathbf{A}^{(1)}\right]^T$. Again, $\mathbf{A}^{(k)}$ can be extracted by using proper BSS methods due to the linearity of (14). Finally, $\widehat{\breve{\underline{\mathbf{G}}}}$ can be computed from

$$\widehat{\breve{\underline{\mathbf{G}}}} = \underline{\mathbf{Y}} \times_1 \mathbf{A}^{(1)\dagger} \times_2 \mathbf{A}^{(2)\dagger} \cdots \times_K \mathbf{A}^{(K)\dagger} \tag{15}$$

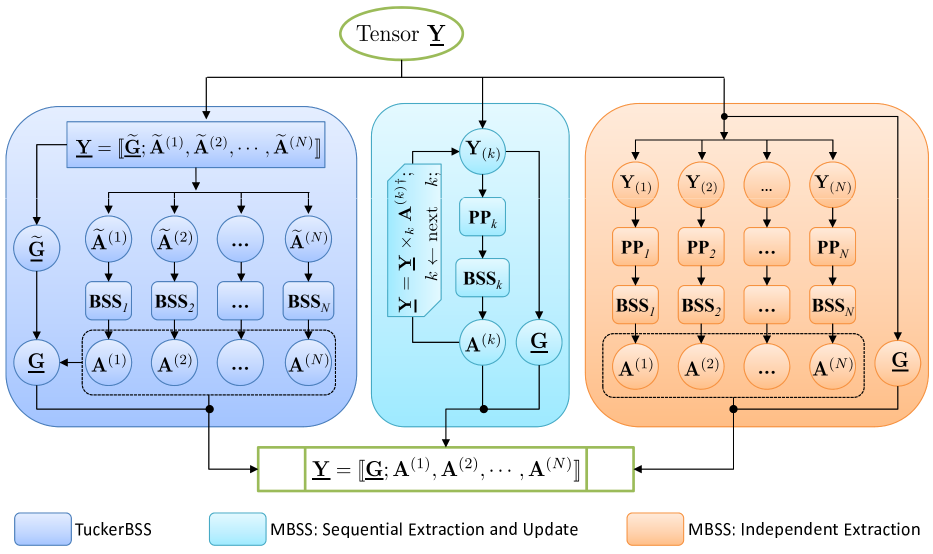
or some other proper methods. This procedure is illustrated in the rightmost diagram of Fig. 2.

Sequential extraction and update of factor matrices. In this case, after $\mathbf{A}^{(n)}$ has been estimated by using (10), then the observation tensor $\underline{\mathbf{Y}}$ can be updated (reduced) as

$$\underline{\mathbf{Y}} \leftarrow \underline{\mathbf{Y}} \times_n \mathbf{A}^{(n)\dagger}. \tag{16}$$

After update, the size of new data $\underline{\mathbf{Y}}$ can be significantly reduced taking into account that $J_n \ll I_n$. For large-scale problems this can reduce the total computational complexity. After all component matrices have been estimated, we let $\underline{\mathbf{G}} = \underline{\mathbf{Y}}$. This way is often more efficient than the *Independent Extraction* approach if the order of data tensor is not very high. However, we have to carefully chose the BSS methods in this approach, because, first of all, current poor separation accuracy may deteriorate the subsequent separation accuracy. Second, the order of selection of modes should be carefully considered according to the constraints and the dimensionality of factor matrices. For example, it is suggested to estimate first nonnegative components since many standard BSS algorithms may often destroy the nonnegativity. Otherwise, the mode with the highest dimensionality may be considered first as it can reduce the data tensor $\underline{\mathbf{Y}}$ most significantly. See Fig. 2 (middle of the figure) for the diagram illustrating this approach.

[5]Since core tensors can be represented in unfolded form via matrix factorizations (see e.g., Eq. 14), we can apply any suitable BSS method to estimate constrained core tensors (e.g., NMF, SCA, ICA) with a little sacrifice in fit.



Tensor $\underline{\mathbf{Y}}$
$\underline{\mathbf{Y}} = [\![\underline{\tilde{\mathbf{G}}}; \tilde{\mathbf{A}}^{(1)}, \tilde{\mathbf{A}}^{(2)}, \cdots, \tilde{\mathbf{A}}^{(N)}]\!]$
$\underline{\tilde{\mathbf{G}}}$
BSS₁ BSS₂ ... BSS_N
$\underline{\mathbf{G}}$
$\mathbf{Y}_{(k)}$
PP_k
BSS_k
$\mathbf{A}^{(k)}$
$\underline{\mathbf{Y}} = \underline{\mathbf{Y}} \times_k \mathbf{A}^{(k)\dagger}$; $k \leftarrow$ next k;
$\mathbf{Y}_{(1)}$ $\mathbf{Y}_{(2)}$ $\mathbf{Y}_{(N)}$
PP₁ PP₂ PP_N
$\underline{\mathbf{Y}} = [\![\underline{\mathbf{G}}; \mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \cdots, \mathbf{A}^{(N)}]\!]$
TuckerBSS
MBSS: Sequential Extraction and Update
MBSS: Independent Extraction

Fig. 2. TuckerBSS versus MBSS. Here, PP$_n$ and BSS$_n$ denote the pre-processing procedure and the BSS method for mode-n, respectively, $(n = 1, 2, \cdots, N)$. Theoretically all the three approaches for noiseless data should give the same results but MBSS methods are more efficient and flexible

The methods described above are referred to as multiway blind source separation (MBSS) since multiple sets of signals with specific physical meaning are extracted from different modes of the data tensor by using BSS methods, based on the multilinear structures of Tucker model. The name MBSS is used to emphasize its difference from ordinary unconstrained Tucker decompositions.

It is worth noticing that there are two possible interpretations of the results using constrained Tucker decompositions for the MBSS. In the first approach the columns of component matrices $\mathbf{A}^{(n)}$ represent the desired components or latent variables, and the core tensor represents a kind of "mixing process". More precisely, the core tensor shows the links among the components from different modes, while the data tensor $\underline{\mathbf{Y}}$ represents a collection of multidimensional mixing signals. In the second approach, the core tensor represents the desired but unknown (hidden) N-dimensional signal (e.g., 3D MRI image or 4D video) and the component matrices represent specific dictionaries or transformations, e.g., time frequency transformations or wavelets dictionaries (mixing or filtering processes). In this case the data tensor $\underline{\mathbf{Y}}$ represents the observed N-dimensional signal, which may be distorted, transformed, compressed or mixed, depending on the specific applications. In this paper we only consider the first interpretation or approach.

Remark. The similar approach can be applied for constrained Candecomp/PARAFAC (CP) model, especially when components are highly collinear or problem is very ill conditioned or sample number in some modes are very small. However, for the CP model we need only to perform unfolding in a single (one) mode and apply a suitable standard BSS (e.g., ICA, NMF or SCA). On basis of the components in this mode we can compute uniquely components in all other models using SVD. Details can be found in our separate paper [33].

3.3. Identifiability conditions and uniqueness of MBSS.

Note that the separability of the MBSS depends on two conditions: full column rank of each mixing matrix $\mathbf{B}^{(n)}$ and suitable assumptions on the component matrices $\mathbf{A}^{(n)}$, for example, assuming that the components in specific modes are statistically independent, or nonnegative and/or sparse. These assumptions are generally application-dependent and are based on some *a priori* knowledge of expected features of $\mathbf{A}^{(n)}$. Here, we always assume that this *a priori* knowledge is available in order to choose suitable BSS algorithms and criteria[6].

For the constrained Tucker decompositions, we have the following proposition:

Proposition 3. If the elements of $\mathbf{A}^{(n)}$ and $\underline{\mathbf{G}}$ are drawn from independent Gaussian distributions, and $\min(\breve{I}_n, \breve{J}_n) \geq J_n$, then $\mathbf{B}^{(n)}$ given by (4) is of full column rank with probability one.

Proof. First $\operatorname{rank}(\mathbf{G}_{(n)}) = J_n$ and $\operatorname{rank}(\mathbf{A}^{(n)}) = \min(I_n, J_n)$, $n = 1, 2, \ldots, N$, hold with probability one [34]. Thus the matrix $\mathbf{G}_{(n)}$ is of full column rank with probability one. Moreover, $\operatorname{rank}(\bigotimes_{p\neq n} \mathbf{A}^{(p)}) = \Pi_{p\neq n}\operatorname{rank}(\mathbf{A}^{(p)})$. It can be easily verified that $\det(\mathbf{B}^{(n)T}\mathbf{B}^{(n)})$ is not identical to zero. The set satisfying $\det(\mathbf{B}^{(n)T}\mathbf{B}^{(n)}) = 0$ is therefore Lebesgue measure zero [35] (Theorem 5 A.2). In other words, $\mathbf{B}^{(n)}$ is of full column rank with probability one. This ends the proof.

[6]For example, components in the frequency domain are usually nonnegative and smooth, while spatial components are usually sparse and temporal components are often mutually independent.

3.4. MBSS versus Tucker BSS. As a matter of fact, many authors have considered imposing constraints on the factor matrices to retrieve meaningful factors [21–24, 36]. For example, Beckamnn and Smith [21], and De Vos *et al.* [22] combined the CP model and ICA. Unkel *et al.* [24] proposed adding ICA algorithms to the Tucker-3 model and this method can be referred to as Tucker-ICA. However, MBSS allows us imposing various constraints in each mode. There is neither a theoretical nor experimental basis for the idea that statistical independence (ICA) is the uniquely correct concept to extract latent hidden components [37, 38]. In real world scenarios, latent (hidden) components may have various statistical properties and features. We often need to apply a fusion of strategies by employing several suitably chosen criteria and the associated learning algorithms to extract all desired components in specific modes [5, 6, 38]. Therefore, if instead the ICA alternative BSS algorithms are adopted, this two-step approach can be referred to as the TuckerBSS method. Figure 2 depicts the basic scheme of TuckerBSS. TuckerBSS looks quite intuitive, simple, and often gives meaningful results.

In the following we explain why we propose MBSS in more details. First of all, MBSS theoretically gives the consistent results with TuckerBSS in the ideal noiseless case, however it is usually more robust for noisy data.

Proposition 4. Suppose that we have two exact Tucker decompositions $\underline{\mathbf{Y}} = [\![\underline{\mathbf{G}}; \mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \ldots, \mathbf{A}^{(N)}]\!]$ and $\underline{\mathbf{Y}} = [\![\underline{\mathbf{H}}; \mathbf{X}^{(1)}, \mathbf{X}^{(2)}, \ldots, \mathbf{X}^{(N)}]\!]$ for the same tensor $\underline{\mathbf{Y}}$, obtained by an arbitrary Tucker decomposition algorithm, and $\mathrm{rank}(\mathbf{A}^{(n)}) = \mathrm{rank}(\mathbf{Y}_{(n)}) = J_n$. Then there holds that $\mathbf{X}^{(n)} = \mathbf{A}^{(n)}\mathbf{Q}^{(n)}$, where $\mathbf{Q}^{(n)} \in \mathbb{R}^{J_n \times J_n}$ is an invertible matrix, $(n = 1, 2, \ldots, N)$.

The proof is straightforward and is omitted here. Proposition 4 means that the Tucker decomposition just gives a linear mixture (i.e., the range) of $\mathbf{A}^{(n)}$. Indeed, any linear mixture of the columns of $\mathbf{A}^{(n)}$ is a solution. To retrieve the unique component matrices from their linear mixtures, the two-step TuckerBSS method applies specific BSS algorithms to the component matrices obtained by unconstrained Tucker decompositions:

$$\widehat{\mathbf{A}}^{(n)} = \Psi_n(\mathbf{X}^{(n)}) = \Psi_n(\mathbf{A}^{(n)}\mathbf{Q}^{(n)}) = \mathbf{A}^{(n)}\mathbf{P}_n\mathbf{D}_n, \quad (n = 1, 2, \ldots, N). \tag{17}$$

By using the MBSS approach, however, these constrained component matrices can be extracted directly from the unfolded matrices $\mathbf{Y}_{(n)}$, as $\mathbf{Y}_{(n)}$ itself is assumed to be a linear mixture of the columns of $\mathbf{A}^{(n)}$. In other words, the source separation and the unique Tucker decomposition can be performed simultaneously, see (10) and (17) for a comparison. Note also that the results are just the same as those obtained via TuckerBSS, because the separation results of BSS are independent of the mixing matrix $\mathbf{B}^{(n)}$ if it is full rank. In other words, although both MBSS and TuckerBSS are able to give unique, meaningful, and essentially consistent components, in MBSS the additional unconstrained Tucker decomposition is simply unnecessary.

In the following we summarize the advantages of MBSS versus TuckerBSS.

1. **Considerable flexibility and robustness**. In MBSS, any existing matrix factorization methods can be employed directly. Furthermore, various pre-processing procedures such as dimensionality reduction, source number estimation developed for matrices can be easily and straightforwardly incorporated, which can significantly improve the efficiency and performance of MBSS. However, in the TuckerBSS we have to carefully design different Tucker algorithms to adapt to different situations. Typically, most existing Tucker decomposition methods have assumed the noises are drawn from independent Gaussian distributions. However, if the noises are, e.g., very sparse and of very high amplitude, they lead to very high approximation error and thus TuckerBSS will achieve very low separation accuracy. In the MBSS, however, we can simply use, e.g., robust PCA proposed in [39] to remove the sparse noise and then extract the latent signals. This feature of MBSS will be illustrated in Simulation 2.
2. **High efficiency and simplicity**. In ALS based methods we have to unfold the tensor and perform matrix-matrix (Kronecker/Khatri-Rao) products frequently. These operations are often time and memory consuming, which severely hinders their applicability, especially for large-scale and noisy data. In the MBSS, we unfold the data tensor in each mode only once. Moreover, we do not need to consider the Kronecker/Khatri-Rao structure of $\mathbf{B}^{(n)}$ in (4) and (10) explicitly due to the essential uniqueness of BSS. This will significantly reduce the computational complexity. Moreover, as we will see later, by incorporating state of the art dimensionality reduction methods, the efficiency of MBSS can be further substantially improved. This feature will be illustrated by extensive simulations in Sec. 5.

Another subtle difference between MBSS and TuckerBSS is that in TuckerBSS the minimum fitting error is pursued first and then the feature information of the components is maximized, whereas in MBSS the feature information is maximized based on the multilinear structure and the feature information directs the components to the desired one. As pure pursuit of minimum fitting error has theoretical limitation [18], and existing Tucker decomposition methods have implicitly or explicitly assumed the noises are Gaussian, MBSS can be more flexible and practicable for real data analysis.

One may argue that MBSS is simply ordinary BSS. It is the truth if we only consider one mode[7]. However, we usually have to extract several factors from the data in different. These multiple factors are linked or associated via the core tensor, see (1), and provide extra information and facilities in data exploration, interpretation, projection, and transformation,

[7]Also, if we look only in one mode, tensor decomposition is simply ordinary constrained matrix factorization, by using matricization. The key point is that in tensor decomposition multiple factors from multiple modes are involved, and links between them are established.

etc. In summary, in MBSS the high-way data is explained and interpreted by a Tucker structure but this structure is realized by using ordinary BSS methods. The MBSS benefits from its delicate and comprehensive multilinear structures compared with ordinary BSS. Moreover, it provides more interpretable components with physical meanings, compared with unconstrained tensor decompositions.

4. Dimensionality reduction and estimation of the number of components

In this section we discuss two important pre-processing procedures, that is, dimensionality reduction and estimation of the number of components. The matrices $\mathbf{Y}_{(n)}$ and $\mathbf{B}^{(n)}$ have the large sizes of $I_n \times \breve{I}_{(n)}$ and $\breve{I}_{(n)} \times J_n$, respectively, typically with $\breve{I}_n \gg J_n$. Therefore, problems we encounter here are usually large-scale, highly over-determined and thus could be challenge in practice. To reduce the computational complexity, the dimensionality reduction step should be performed first. Moreover, the adaptive estimation of the number of components J_n in each specific mode of interest is another important issue. Note that at first, we have assumed that the noises are Gaussian, which is the standard assumption for most tensor decomposition methods. After that we will briefly discuss how to deal with the cases when have different non-Gaussian distributions.

In order to perform dimensionality reduction, we can apply standard PCA (e.g., using truncated SVD) to each unfolding matrix $\mathbf{Y}_{(n)}$. In more detail, we perform eigenvalue decompositions $\mathbf{Y}_{(n)}\mathbf{Y}_{(n)}^T = \mathbf{U}_{J_n}\mathbf{\Lambda}\mathbf{U}_{J_n}^T$ first (supposing that $\breve{I}_n > I_n$), where $\mathbf{U}_{J_n}$ consists of the first J_n leading eigenvectors. Then, we run BSS on the dimensionality reduced matrix $\mathbf{U}_{J_n}^T$ directly[8]. This way involves the eigenvalue decomposition of an $I_n \times I_n$ matrix, and gives the optimal low-rank approximation of the observations, in the sense of least square. Therefore, PCA is preferred for some ordinary BSS tasks if the scale of the problem is moderate. However, it suffers from a heavy computation load and huge memory use for large-scale problems, i.e., both $\breve{I}_n$ and I_n are very large [40]. Moreover, PCA/SVD will not preserve nonnegativity constraints, thus it is not directly suitable for the cases in which nonnegative components are desired.

Note that each column of $\mathbf{Y}_{(n)}$, namely $\mathbf{y}_{(n)}$, is exactly a linear combination of the columns of $\mathbf{A}^{(n)}$. Thus, we can estimate $\mathbf{A}^{(n)}$ from a new observation matrix whose columns are sampled from the columns of $\mathbf{Y}_{(n)}$ since BSS is generally independent of the specific mixing matrix. By thus, the dimensionality of observations can be significantly reduced, and the nonnegativity can be preserved. In ideal noise free case, even only J_n columns would be sufficient to estimate $\mathbf{A}^{(n)}$. In general, we want to use as small as possible number of columns to approximate the original huge observation matrix, then run BSS on the significantly reduced observation matrix. The CUR method presented in [41] confirms that a huge matrix can be approximated by suitably sampling its columns and/or rows. Based on this, we can run BSS on the sampled columns of $\mathbf{Y}_{(n)}$ without accessing the whole tensor[9]. By using this approach the computational efficiency can be significantly improved and the use of memory can be reduced as well.

Another important fact is that the columns of $\mathbf{Y}_{(n)}$ are simply built up from the mode-n fibers[10] of $\underline{\mathbf{Y}}$. Using MATLAB notations, the mode-n fibers of an N-way tensor $\underline{\mathbf{Y}}$ are denoted as $\mathbf{y}_{i_1 i_2 \cdots i_{n-1},:,i_{n+1}\cdots i_N}$, or in short, $\mathbf{y}_{(n)}$ [32]. In the MATLAB environment, thanks to the support for multidimensional arrays, we can access and sample the fibers directly from the tensor, and the sampled fibers form a reduced observation matrix, say $\widetilde{\mathbf{Y}}_{(n)}$, without the need to explicitly construct the full unfolding matrix $\mathbf{Y}_{(n)}$ in advance. Consequently, the sampling procedure can be very efficient and it is very similar to the Fiber Sampling Tensor Decomposition (FSTD) method [42], which is a generalization of CUR decomposition for tensors (see Fig. 3). The FSTD is based on the following theoretical results [42]:

$$\underline{\mathbf{Y}} \approx \underline{\widetilde{\mathbf{G}}} \times_1 \widetilde{\mathbf{Y}}_{(1)} \times_2 \widetilde{\mathbf{Y}}_{(2)} \cdots \times_N \widetilde{\mathbf{Y}}_{(N)}, \tag{18}$$

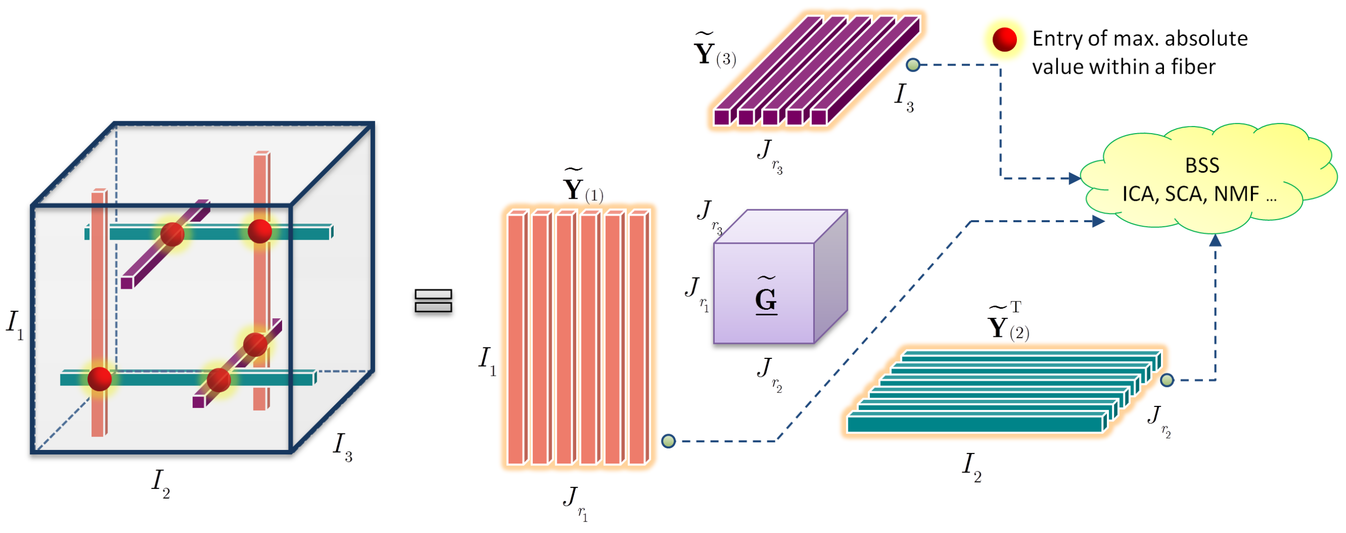
where $\widetilde{\mathbf{Y}}_{(n)} \in \mathbb{R}^{I_n \times J_{rn}}$ are matrices consisting of mode-n fibers sampled from the data tensor $\underline{\mathbf{Y}}$ directly, $I_n \gg J_{r_n} \geq J_n$. The value of J_{r_n} depends on the level of noise and in practice it is often sufficient that $J_{r_n} \geq 5J_n$ [5]. It should be noted that each column of $\widetilde{\mathbf{Y}}_{(n)}$ is a linear combination of the columns of $\mathbf{A}^{(n)}$. Hence, we can directly apply the FSTD procedure in the MBSS, i.e., run BSS on $\widetilde{\mathbf{Y}}_{(n)}$ and retrieve $\mathbf{A}^{(n)}$ from the linear mixtures $\widetilde{\mathbf{Y}}_{(n)}$, as illustrated in Fig. 3. However, in our MBSS approach there are some major simplifications and modification with comparison to the original FSTD:

- The computation of the core tensor $\underline{\widetilde{\mathbf{G}}}$ is unnecessary, since it is not essential for the estimation of $\mathbf{A}^{(n)}$, $n = 1, 2, \ldots, N$.
- We do not need to sample all the modes simultaneously. Instead we sample each mode sequentially and independently.
- The numbers of components, i.e., J_n, are usually unknown in practice. In such case we need to determine the value of J_n (and thus J_{r_n}).

[8]If occasionally $I_n > \breve{I}_n$, the eigenvalue decomposition of $\mathbf{Y}_{(n)}^T\mathbf{Y}_{(n)} = \mathbf{V}_{J_n}\mathbf{\Sigma}\mathbf{\Lambda}_{J_n}^T$ should be computed instead. Then we apply a suitable BSS method for low-rank approximation matrix $\mathbf{V}_{J_n}^T\mathbf{Y}_{(n)}^T$.

[9]This sampling procedure only reduces the redundant mixed (observed) signals without destroying any property (e.g., temporal property) of source signals. Therefore, we can run BSS methods on the sampled mixtures to estimate the sources.

[10]A *fiber* of a tensor $\underline{\mathbf{Y}}$ is a vector defined by fixing every subscript of $\underline{\mathbf{Y}}$ but one.



$\widetilde{\mathbf{Y}}_{(3)}$
Entry of max. absolute value within a fiber
I_3
J_{r_3}
BSS
ICA, SCA, NMF ...
$\widetilde{\mathbf{Y}}_{(1)}$
J_{r_3}
J_{r_1}
$\widetilde{\mathbf{G}}$
J_{r_2}
$\widetilde{\mathbf{Y}}^T_{(2)}$
I_1
I_1
I_2
I_3
J_{r_1}
J_{r_2}
I_2

Fig. 3. Illustration of FSTD and MBSS based on fiber sampling of a 3D tensor. Note that on one hand, the columns of $\widetilde{\mathbf{Y}}_{(n)}$ are sampled from the fibers of the original data tensor $\underline{\mathbf{Y}}$; on the other hand, these columns/fibers are the linear combinations of the columns of factors $\mathbf{A}^{(n)}$. Thus we can run various BSS algorithms (optionally, with PCA) on the sampled fibers directly to extract $\mathbf{A}^{(n)}$

Now, we will discuss how to determine the number of fibers to be sampled. Assume that the additive noises are drawn from independent Gaussian distributions with zero mean, and a total of J_{r_n} fibers are sampled and stored in $\widetilde{\mathbf{Y}}_{(n)}$. Let $\widetilde{\mathbf{Y}}^T_{(n)}\widetilde{\mathbf{Y}}_{(n)} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^T$ be the eigenvalue decomposition of $\widetilde{\mathbf{Y}}^T_{(n)}\widetilde{\mathbf{Y}}_{(n)}$ with the eigenvalues $\lambda_i,\ i = 1, 2, \cdots, J_{r_n}$. It is well known that $\widehat{\lambda}_i = \lambda_i + \sigma_\varepsilon^2$ for $i = 1, 2, \cdots, J_n$, and $\widehat{\lambda}_i = \sigma_\varepsilon^2$ for $i > J_n$, where λ_i corresponds to the signal space and σ_ε^2 measures the level of noise. Without loss of generality, assume that [9]

$$\widehat{\lambda}_1 \geq \widehat{\lambda}_2 \cdots \geq \widehat{\lambda}_{J_n} > \widehat{\lambda}_{J_n+1} \approx \cdots \approx \widehat{\lambda}_{J_{r_n}} \approx \sigma_\varepsilon^2. \qquad (19)$$

Intuitively, to estimate the number of components, i.e., J_n, we only need to locate the GAP (jump) between the eigenvalues corresponding to the signal space (with noise) and the pure noise space. Based on (19), a so-called Second ORder sTatistic of the Eigenvalues (SORTE) method is developed to locate this GAP [9]. Let $\nabla\widehat{\lambda}_i = \widehat{\lambda}_i - \widehat{\lambda}_{i+1}$ denote the difference of neighbor eigenvalues and $\widehat{\sigma}_p^2$ be the variance of $\{\nabla\widehat{\lambda}_i :\ i = p, p+1, \cdots, J_{r_n} - 1\}$. That is,

$$\widehat{\sigma}_p^2 = \text{var}\left[\{\nabla\widehat{\lambda}_i\}_{i=p}^{J_{r_n}-1}\right]$$
$$= \frac{1}{J_{r_n} - p}\sum_{i=p}^{J_{r_n}-1}\left[\nabla\widehat{\lambda}_i - \frac{1}{J_{r_n} - p}\sum_{i=p}^{J_{r_n}-1}\nabla\widehat{\lambda}_i\right]^2. \qquad (20)$$

The SORTE method estimates the number of components, i.e., J_n, by locating the maximal GAP between the eigenvalues of $\widehat{\lambda}_i$ as follows:

$$J_n = \arg\min_p \text{GAP}(p) = \arg\min_p \frac{\widehat{\sigma}_{p+1}^2}{\widehat{\sigma}_p^2}, \qquad (21)$$
$$p = 1, 2, \ldots, J_{r_n} - 3.$$

The SORTE implicitly depends on a reliable and stable estimation of the eigenvalues of the covariance matrix, which means that the number of samples should be sufficiently large. Therefore, we keep sampling the columns of $\mathbf{Y}_{(n)}$ until a satisfactory estimate of $\widehat{\lambda}_i$, i.e., a stable estimate of J_n, is reached, as demonstrated by Fig. 4. In Fig. 4 the data (observation) matrix $\mathbf{Y} \in \mathbb{R}^{100\times1000}$ with rank 10 is contaminated by white Gaussian noise with SNR=20dB, r is the rank estimated by SORTE from the J_r sampled columns, and ρ is defined as

$$\rho = \frac{Err_s - Err^*}{\|\mathbf{Y}\|_F}, \qquad (22)$$

where Err_s is the best rank-r approximation error by running PCA on the J_r sampled columns, whereas Err^* is the optimal rank-r fitting error achieved by PCA using the matrix $\mathbf{Y}$. From Fig. 4, with the sampled columns J_r increasing, the number of components r estimated by SORTE increases and fluctuates at first. However, after $J_r \geq 20$, r does not increase any more. Moreover, the approximation is gradually closer to the optimal low-rank approximation. Thus, the number of components is estimated, and at the same time, the observation matrix is approximated by a much smaller submatrix $\widetilde{\mathbf{Y}}$. In the next section simulations will show that this sampling procedure achieves a good tradeoff between efficiency and accuracy. Based on above analysis, a fast Fiber Sampling-based SORTE (FSSORTE) method is proposed and implemented to perform estimation of the number of components and dimensionality reduction, see Algorithm 4 for details. (In the algorithm, freq(r_n) computes the frequency of occurrence of r_n).

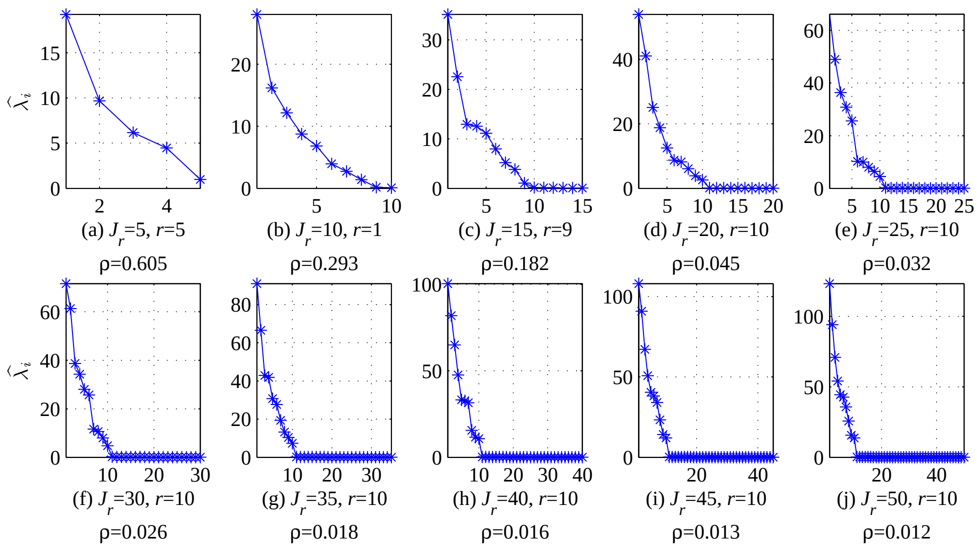


Fig. 4. Illustration of adaptive low-rank approximation and rank detection by random sampling-based SORTE. The rank r is the rank estimated from the J_r sampled columns by using SORTE, and ρ is defined by (22)

Algorithm 1: Fiber sampling-based SORTE for dimensionality reduction and rank detection (FSSORTE)

```
Input: $\mathbf{Y}_{(n)} \in \mathbb{R}^{I_n \times \breve{I}_n}$, $(n = 1, 2, \ldots, N)$.
Output: $J_n = r_n$ and $\tilde{\mathbf{Y}}_{(n)} \in \mathbb{R}^{I_n \times J_n}$ (or $\tilde{\mathbf{Y}}_{(n)} \in \mathbb{R}^{I_n \times t}$ for NMF).
for $t = 1, 2, \cdots$, do
    $\widetilde{\mathbf{Y}}_{(n)}(:, t) = \mathbf{y}^{(t)}_{(n)}$, where $\mathbf{y}^{(t)}_{(n)}$ is a randomly sampled mode-$n$ fiber and $\|\mathbf{y}^{(t)}_{(n)}\| > \varepsilon$;
    if mode$(t, p_0) = 0$ then
        $\widehat{\lambda}_t \leftarrow$ eigenvalues of $\widetilde{\mathbf{Y}}^T_{(n)} \widetilde{\mathbf{Y}}_{(n)}$;
        $r_n = \text{SORTE}(\widehat{\lambda}_t)$;
        if freq$(r_n) > \gamma$ and $t > p \times r_n$ then
            Output $\widetilde{\mathbf{Y}}_{(n)}$ if nonnegativity is required; otherwise output $\widetilde{\mathbf{Y}}_{(n)} = \widetilde{\mathbf{Y}}_{(n)} \mathbf{V}_{r_n}$, where $\mathbf{V}_{r_n}$ is the $r_n$ eigenvectors corresponding to the $r_n$ largest eigenvalues of $\widetilde{\mathbf{Y}}^T_{(n)} \widetilde{\mathbf{Y}}_{(n)}$.
            break;
        end if
    end if
end for
```

In the FSTD method the entries with larger absolute values dominate the sampling procedure. In our case random uniform sampling is utilized instead but with the restriction that $\|\mathbf{y}^{(t)}_{(n)}\| > \epsilon$. In the case where the number of components is known, we can simply terminate FSSORTE if $J_{r_n} > pJ_k$, and $p \geq 5$ is an empirically-based choice [5]. FSSORTE outputs the optimal number of components $r_n = J_k$ and the significantly reduced matrix $\widetilde{\mathbf{Y}}_{(n)}$. Compared with standard PCA, Algorithm 1 performs eigenvalue decomposition of a series of $J_{r_n} \times J_{r_n}$ matrices. Considering that the number of latent sources of the high dimensional data is relatively small with $J_n \leq J_{r_n} \ll I_n$, Algorithm 1 is more efficient than standard PCA. Note that the projection $\widetilde{\mathbf{Y}}_{(n)} \mathbf{V}_{r_n}$ actually gives a random approximation of PCA and it can be used for ordinary BSS algorithms. However, if the original tensor is nonnegative and we want to run NMF to estimate the component matrices, we generally use the sampled fibers $\widetilde{\mathbf{Y}}_{(n)}$ directly without projection.

It is worth noticing that in the above analysis the noises have been assumed to be Gaussian. Under this consumption, some methods to estimate the number of components has been proposed for three way tensors, see, e.g., [43, 44]. On the other hand, some alternative methods have been proposed to estimate the latent components in the matrix case [45]. Apparently, the proposed approach enables us to employ all these methods freely and avoid extensively repeating these work which has been done for matrices.

If the noise is not Gaussian, we need to apply another suitable low-rank approximation methods. For example, if the additive noise is very sparse, we can use the robust PCA (RPCA) proposed in [39] to perform dimensionality reduction and filter out noise. From this point of view, MBSS is actually quite flexible because any well-established matrix factorization methods can be easily incorporated for specific purposes and analysis, instead of re-designing all kinds of new methods for tensors.

5. Simulations

In this section the validity and efficiency of the MBSS is investigated by simulations of synthetic and real data. We use two performance indices (PI) to evaluate the performance. The first one is the signal-to-interference ratio (SIR) which is defined by

$$\text{SIR}(\mathbf{a}, \widehat{\mathbf{a}}) = 10 \log_{10} \frac{\sum_t a_t^2}{\sum_t (a_t - \widehat{a}_t)^2}, \tag{23}$$

where $\mathbf{a}$, $\widehat{\mathbf{a}}$ are normalized with zero mean and unit variance, and $\widehat{\mathbf{a}}$ is an estimate of $\mathbf{a}$. The value of SIR reflects how well the estimated component (source) matches the true original one. The second PI measures the fit of the estimated tensor to the original raw data tensor and is defined as

$$\text{fit}(\underline{\mathbf{Y}}, \widehat{\underline{\mathbf{Y}}}) = 1 - \frac{\|\underline{\mathbf{Y}} - \widehat{\underline{\mathbf{Y}}}\|_F}{\|\underline{\mathbf{Y}}\|_F}, \tag{24}$$

where $\widehat{\underline{\mathbf{Y}}}$ is an estimate of $\underline{\mathbf{Y}}$. For synthetic data $\underline{\mathbf{Y}}$ denotes the noiseless tensor whereas it denotes the observation data for real data because in this case the latent noiseless data are unknown. Obviously, $\text{fit}(\underline{\mathbf{Y}}, \widehat{\underline{\mathbf{Y}}}) = 1$ if and only if $\widehat{\underline{\mathbf{Y}}} = \underline{\mathbf{Y}}$. All the simulations were done in MATLAB 2008a on a computer with Intel 7i 3.33GHz CPU and 24GB memory running Windows 7.

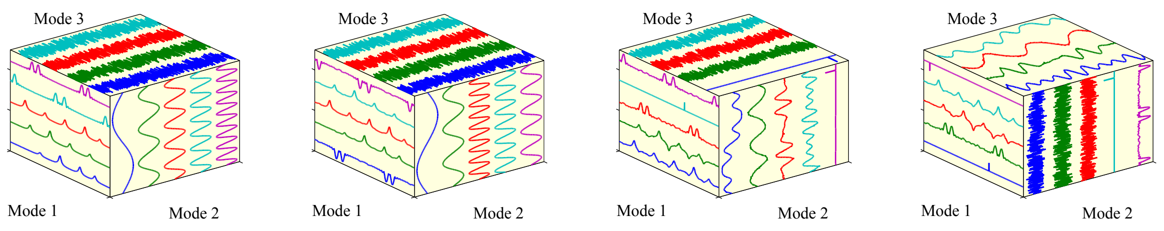
Simulation 1. In this simulation different constraints were imposed on the components in different modes, that is, temporal (second-order statistics) decorrelation in mode-2 and mode-3 (they were chosen as acsin10d, which is included in the benchmark of ICALAB [46]), and the higher-order statistical independence in mode-1. The elements of $\mathbf{A}^{(1)} \in \mathbb{R}^{500\times4}$ and the core tensor $\underline{\mathbf{G}} \in \mathbb{R}^{4\times5\times5}$ were drawn from independent uniform distributions. Finally, very heavy Gaussian noise with SNR=0dB was added. We used EFICA [47] to extract the first component matrix and the SOBI [48] to extract the two others, respectively. The MBSS approach was compared with the higher-order orthogonal iteration (HOOI) algorithm presented in [32] and the standard ALS, i.e., TuckerALS implemented in the N-way toolbox [49]. Because the ordinary Tucker decompositions are always non-unique, we used EFICA and SOBI to refine the component matrices obtained by HOOI and TuckerALS, which can be viewed as two-step implementations of TuckerBSS, and were denoted as HOOI+BSS and TuckerALS+BSS, respectively. The maximum iteration number was set 100 for HOOI and TuckerALS. The results are shown in Table 1. It can be seen that if all the fibers were employed, MBSS achieved the same separation accuracy as the other two methods. However, MBSS was about two times faster. When $500J_n$ fibers were sampled (denoted by 500x), MBSS was about five times faster than the other methods and achieved satisfactory separation accuracy. This feature makes MBSS very competitive for large-scale problems.

Table 1
Comparison of performances of the MBSS, HOOI+BSS and TuckerALS+BSS for the decomposition of a large tensor with mixed constraints. Very heavy Gaussian noise with SNR=0dB was added. MBSS [500x] used $500J_n$ randomly sampled fibers whereas MBSS [100%] used all the fibers

Algorithm	mSIR$_1$	mSIR$_2$	mSIR$_3$	Runtime(s)	Fit
MBSS [500x]	41	17	25	1.9	0.96
MBSS [100%]	41	18	25	4.1	0.99
HOOI	41	18	24	9.3	0.99
TuckerALS	47	18	25	11	0.99

In some applications the noise may not be Gaussian. For example, in electroencephalography (EEG) signal processing, eye blink artifacts typically are sparse yet have very large amplitude. To simulate this kind of situations, we added sparse noise to the observation tensor $\underline{\mathbf{Y}}$ as follows. We randomly selected 1000 entries from $\underline{\mathbf{Y}}$ and added huge bipolar sparse noise (outliers drawn from Gaussian distribution) with the SNR = -20 dB.

For this corrupted by spiky noise tensor, both HOOI+BSS and TuckerALS+BSS failed to retrieve the latent components, see Fig. 5c and d. In fact they also achieved very low fit to the original tensor, as shown in Table 2. In MBSS, we randomly sampled only $100J_n$ fibers first. Then, we applied the Robust PCA (RPCA) proposed in [39] on the sampled fibers to filter out the sparse outliers noise. Finally, we apply ordinary PCA, SOBI and FastICA algorithms to retrieve hidden factors. The recovered waveforms were plotted in Fig. 5b. It can be seen that the recovered signals match the true sources very well. The mean SIRs, runtime, and fit are detailed in Table 2. In fact, only MBSS was able to extract the desired components with a high accuracy in the presence of outliers. From the simulation results, MBSS is more flexible and versatile as any off-the-shelf matrix factorization methods can be easily incorporated[11]. Figure 6 plots the mean SIRs of $\mathbf{A}^{(1)}$ obtained by MBSS with different configurations over 50 independent runs. The blue crosses mark the results obtained by MBSS [500x] when 0 dB Gaussian noise was involved and the red asterisks denote the MBSS incorporating RPCA when -20 dB sparse noise was involved, which shows that



(a) The sources (b) MBSS (c) HOOI+BSS (d) TuckerALS+BSS

Fig. 5. Comparison of performances of the MBSS, HOOI+BSS and TuckerALS+BSS in the decomposition of a tensor with mixed constraints. Very sparse noise with SNR=-20dB was added to the observations tensor. In MBSS we used $100J_n$ randomly sampled fibers and then we preprocessed them by employing RPCA

[11]In this case the improved performance was achieved via suitable preprocessing which was relatively easy to incorporate into the MBSS approach. The implementation of such preprocessing, however, is not so straightforward for other existing methods.

the separation accuracy was quite stable although MBSS only used randomly sampled fibers. By fibers sampling, MBSS can make a good trade-off between the separation accuracy and efficiency. This feature is quite promising for large-scale data analysis. To justify that the favorable performance achieved by the MBSS with the RPCA was not due to the sampled fibers (which occasionally noise-free, since the noises were very sparse), the MBSS procedure was repeated but without the RPCA preprocessing procedure and results are presented in the same figure with the cyan circles marks). From the figure, it is evident that the RPCA played a key role to improve the performance of the MBSS for data with outliers. In summary, compared with ordinary Tucker decomposition, carefully designed MBSS is more flexible and robust and it is able to tackle some very challenging problems in practical applications.

Table 2
Comparison of performances of the MBSS, HOOI+BSS and TuckerALS+BSS in the decomposition of a tensor with mixed constraints. Very large sparse noise with SNR = -20 dB was added. The MBSS used the RPCA to preprocess the sampled fibers

Algorithm	mSIR$_1$	mSIR$_2$	mSIR$_3$	Runtime (s)	Fit
MBSS	**47**	**18**	**21**	**2.4**	**1.0**
HOOI	2.1	2.4	1.0	63	-0.4
TuckerALS	5.1	7.7	11	58	-0.0

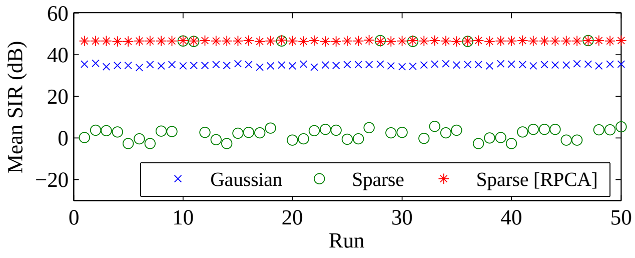


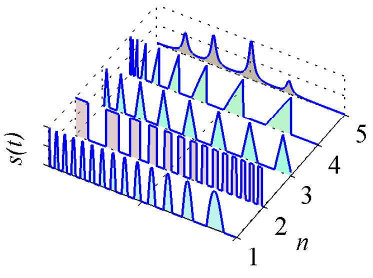
Fig. 6. Mean SIRs of $\mathbf{A}^{(1)}$ obtained by the MBSS with different configurations over 50 runs. The results are stable although randomly sampled fibers were used. The MBSS without the RPCA performed poorly, which justifies to use the RPCA

Simulation 2. In this simulation MBSS incorporating NMF methods is applied to perform nonnegative Tucker decomposition (NTD) of large-scale data. Three sets of nonnegative signals were chosen from the benchmarks of NMFLAB, see Fig. 7a-c for their waveforms. Each set consisted of five signals and the sample numbers were all 1,000. The entries of the core tensor $\underline{\mathbf{G}} \in \mathbb{R}^{5\times5\times5}$ were drawn from a uniform distribution, and 60% of them were randomly set to zero. By using such components a large data tensor $\underline{\mathbf{Y}} \in \mathbb{R}^{1000\times1000\times1000}$ with sparse components was constructed using the Tucker model. Finally, 20 dB noise drawn from independent uniform distributions was added to $\underline{\mathbf{Y}}$. For this large data tensor, most of existing NTD algorithms such as the HALSNTD [50] and HONMF algorithm [16] ran *out of memory* or their convergence was extremely slow in our computer and thus failed to perform nonnegative Tucker decompositions. In MBSS, this time we applied the DNNMF algorithm [51] to extract nonnegative components in each mode and using only $60J_n$ (i.e., 300) fibers $n = 1, 2, 3$. Figure 7d-f shows the recovered waveforms by the MBSS and (g) shows the corresponding SIRs of each component. From the figure, it is evident that all components were nicely recovered by the MBSS. The averaged SIRs of the estimated components were 21 dB, 29 dB, and 26 dB. Moreover, MBSS consumed only 32.3 seconds and achieved a fit of 0.9192 in a typical run. (We have used the multiplicative update rule proposed in [52] to obtain the nonnegative core tensor). For a comparison, TuckerALS with nonnegativity constraints consumed 2344.9 seconds to achieve a fit of 0.8132, using only one iteration. By setting the maximum iteration number only 10, TuckerALS consumed 23440 seconds and achieved a fit of 0.92. However, the averaged SIRs of the components estimated by TuckerALS were only 3.9, 2.4, and 2.4 dB. From these results we can conclude that the MBSS is quite competitive with existing algorithms for large-scale nonnegative Tucker decomposition.

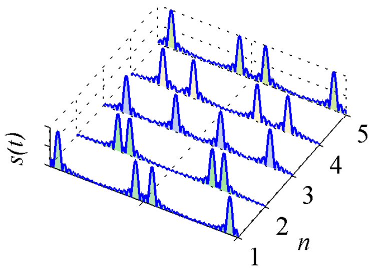
Simulation 3. In this simulation we applied MBSS to the COIL-20 images clustering [53]. The database COIL-20 consists of 1440 gray images of 20 objects (72 images per object). Each image has been preprocessed and was with the size of 128 × 128, which was captured from different orientations of a object. We stacked these images together and formed a tensor $\underline{\mathbf{Y}}$ with the size of 128 × 128 × 1440. In the first step the tensor $\underline{\mathbf{Y}}$ was decomposed by using MBSS with $J_1 = J_2 = 5,\ J_3 = 25$. In MBSS, we used LRA-NMF described in detail in [52] to extract each factor. Then we obtained two sets of features: $\mathbf{F}_\mathrm{M}=\widehat{\mathbf{A}}^{(3)}$ and

$$\mathbf{F}_\mathrm{T} = \underline{\mathbf{Y}} \times_1 \widehat{\mathbf{A}}^{(1)\dagger} \times_2 \widehat{\mathbf{A}}^{(2)\dagger}, \tag{25}$$

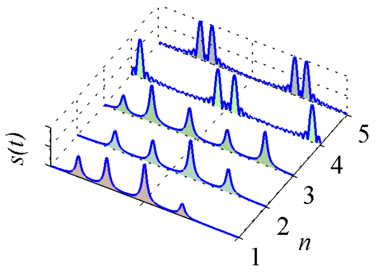
where the features in $\mathbf{F}_\mathrm{M}$ are a sort of features which can actually be extracted by running matrix factorization, e.g., NMF, on the unfolding version $\mathbf{Y}_{(3)}$ directly, whereas the features in $\mathbf{F}_\mathrm{T}$ are tensor features as they have exploited the Tucker structure of tensors. Then we used their two t-SNE [54] components to visualize and cluster the images by using the K-means method. It is well known that K-means often gives slightly different clustering results in different runs. To reduce the uncertainty, in each run of K-means, K-means was replicated 5 times, each with a new of initial cluster centroid positions (see the document for K-means included in Matlab Statistics Toolbox). Then we ran K-means 100 times and their averaged values of clustering accuracy were recorded and compared (see [55] for the definition of clustering accuracy). From our experimental results, if $\mathbf{F}_\mathrm{M}$ was used, the averaged clustering accuracy was 72.57%. If $\mathbf{F}_\mathrm{T}$ was adopted, the averaged accuracy was increased to 79.58%, which shows that the clustering procedure indeed benefitted from the Tucker structure. This also shows that, although the MBSS is based on independent matrix factorizations, it is different from ordinary matrix factorizations because its factors actually share a special multilinear structure. This also partially explain why tensor decompositions cannot be simply replaced by matrix factorizations. In fact, its Tucker structure allows



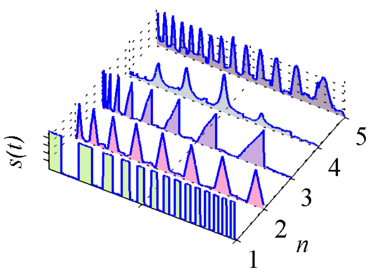
(a) The sources in mode 1



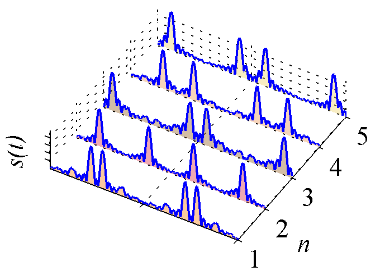
(b) The sources in mode 2



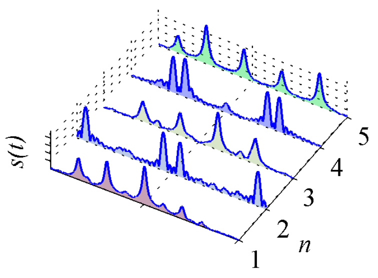
(c) The sources in mode 3



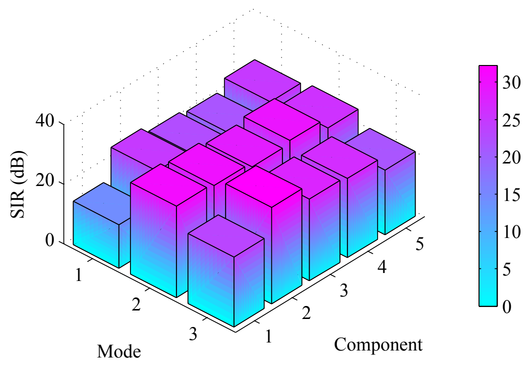
(d) The estimated sources in mode 1



(e) The estimated sources in mode 2



(f) The estimated sources in mode 3



(g) SIRs of individual signals achieved by using the MBSS

Fig. 7. Performance of the MBSS for Nonnegative Tucker Decomposition (NTD) of a large-scale tensor with the size of $1000 \times 1000 \times 1000$, where the DNNMF algorithm was used to extract nonnegative components. Most source signals were nicely recovered and the runtime was only 32 seconds. For this large-scale tensor NTD methods such as HALSNTD and HONMF ran *out of memory* on our computer standard PC

us to perform more complicated data analysis tasks. The above mentioned implementation of the MBSS was also compared with the HOOI, the TuckerALS, and the HALSNTD. The maximum iteration number of these ALS algorithms was set 500. After the decomposition, $\mathbf{F}_\mathrm{T}$ defined by (25) were computed and their two t-SNE components were clustered by using K-means. In clustering stage all the algorithms used the same configuration as MBSS, which has been detailed above. The results are shown in Table 3. Note that the HOOI did not impose any nonnegative constraints, Tucker-ALS only imposed the nonnegative constraints on the loading matrices (it currently cannot impose nonnegativity on the core tensor), whereas the HALSNTD and the MBSS impose the full nonnegative constraints (all the loading matrices and the core tensor are nonnegative). So the fits of the HALSNTD and MBSS were slightly lower than those of the HOOI and TuckerALS. MBSS did not use ALS iterations, however, it obtained almost the same fit as the HALSNTD, but much faster than the HALSNTD and the TuckerALS (MBSS was slower than the HOOI because the HOOI did not need to impose nonnegative constraints.) The final fit and clustering accuracy reveals that, although without ALS iterations, the MBSS is able to provide better or similar results by consuming significantly reduced time in comparison to ALS methods.

Table 3
Comparison of performance of 4 algorithms for the real-world application – COIL 20 images clustering

Algorithm	HOOI	TuckerALS	HALSNTD	MBSS
Runtime(s)	13	5644	3603	45
Fit	0.69	0.69	0.67	0.66
Accuracy (%)	73	71	78	80

Simulation 4. In this simulation the MBSS is applied to the steady-state visual evoked potential (SSVEP) data analysis using real EEG data. SSVEP is a periodic electrical response over the occipital scalp areas of the brain, elicited by the repetitive presentation of a flickering visual stimulus. SSVEP has the same frequency (plus higher harmonics) as the stimulus, and can be recorded from the scalp using electroencephalography (EEG) [56]. Based on this mechanism a SSVEP brain-computer interface (BCI) can be designed, which typically depends on the external visual stimuli which are in the form of an array of light sources with different and distinct frequencies [57]. SSVEP BCI can translate the frequency modulation of EEG signals into computer commands, by recognizing the frequency components of the EEG signals recorded during different stimulus presentations [58]. How to extract and recognize SSVEP components accurately is one of the crucial issues for SSVEP BCI. Although SSVEP is evoked by a repetitive stimulus with relatively stationary intensity, the spontaneous EEG signal or noise with the same frequency as the stimulus and its harmonics, but having time-varying intensity, may contaminate the SSVEP measured from the scalp and make it an unstable signal [59]. Effective extraction of the true SSVEP components from the EEG signals will help in enhancing the recognition accuracy of stimulus frequency components, thereby improving the performance of SSVEP BCI. Here, we validate the proposed MBSS algorithm with real EEG data recorded in a SSVEP experiment in comparison to the TuckerALS, the HONMF and the HALSNTD [50] algorithms.

The EEG signals were recorded at a 250 Hz sampling rate from eight channels P7, P3, Pz, P4, P8, O1, Oz and O2 (arranged according to the international 10–20 standard system) by a Biosemi Active Two amplifier, with a bandpass filtering 5–50 Hz. The EEG data were then collected from 12 stimulus trials with 4 s duration for each trial. The first six trials correspond to 6.5 Hz flickering frequency and the remaining to 10.5 Hz. A complex Morlet wavelet transform was first applied to obtain the time-frequency information from the EEG data with the minimum frequency resolution of 0.5 Hz. Finally, an EEG spectrum tensor with dimension of 91 frequency components $\times$ 1000 sample points $\times$ 8 channels $\times$ 12 trials was constructed.

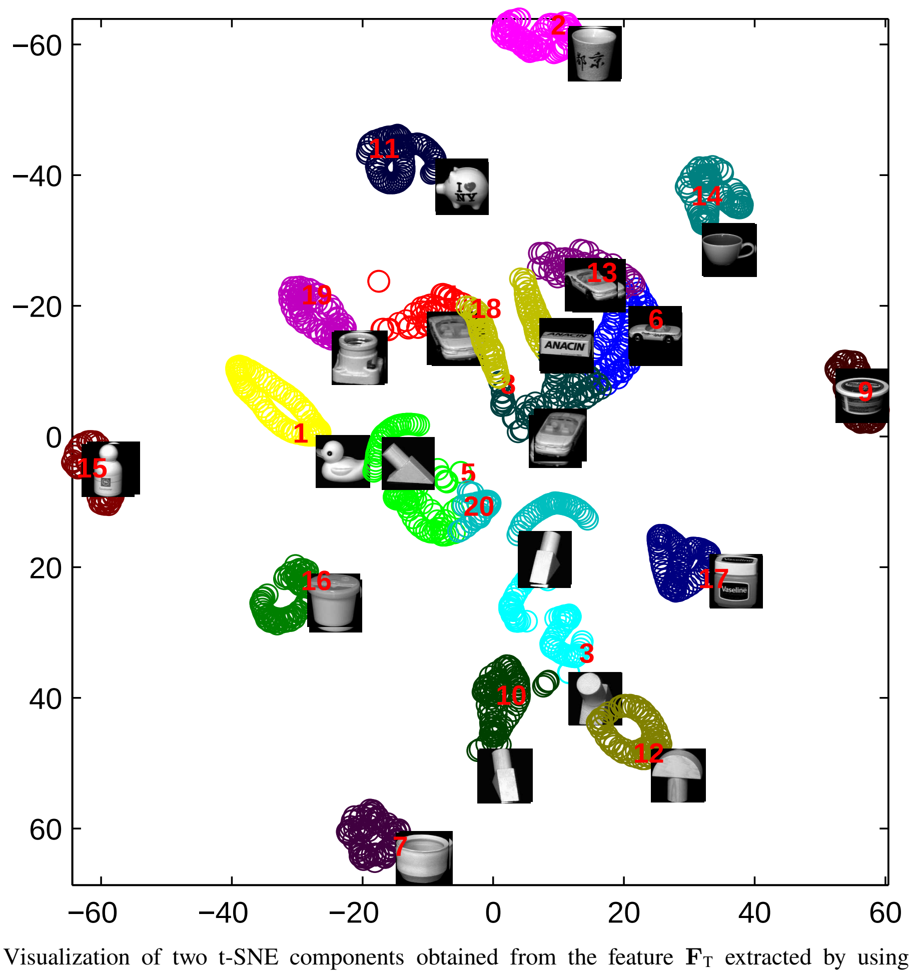


Fig. 8. Example 3: Visualization of two t-SNE components obtained from the feature $\mathbf{F}_{\mathrm{T}}$ extracted by using MBSS with LRA-NMF. The averaged clustering accuracy was 79.58% over 100 runs

In the MBSS, only $40J_n$ fibers were uniformly randomly sampled and then the DNNMF algorithm was employed to extract the latent non-negative components. Figure 9 illustrates the four-way EEG spectrum tensor factorization results upon applying the MBSS. Two components in factor $\mathbf{A}^{(1)}$ explicitly represent the SSVEP frequency components consisting of the fundamental frequency and higher harmonics, corresponding to the stimulus frequencies of 6.5 Hz and 10.5 Hz. The components of $\mathbf{A}^{(2)}$ and $\mathbf{A}^{(3)}$ reflect the temporal and spatial information about the SSVEP spectrum, respectively. The components in factor $\mathbf{A}^{(4)}$ provide considerably discriminative class information, which show trials 1–6 have larger contributions on the stimulus frequency of 6.5 Hz, whereas trials 7–12 are more related to the stimulus frequency of 10.5 Hz. We further compare the TuckerALS with nonnegativity constraints, MBSS, HONMF and HALS. The maximum iteration number for each algorithm was 100. Figure 10 shows the class information obtained by each algorithm. From the figure, it is seen that the MBSS yields much more discriminative class features than the other existing algorithms.

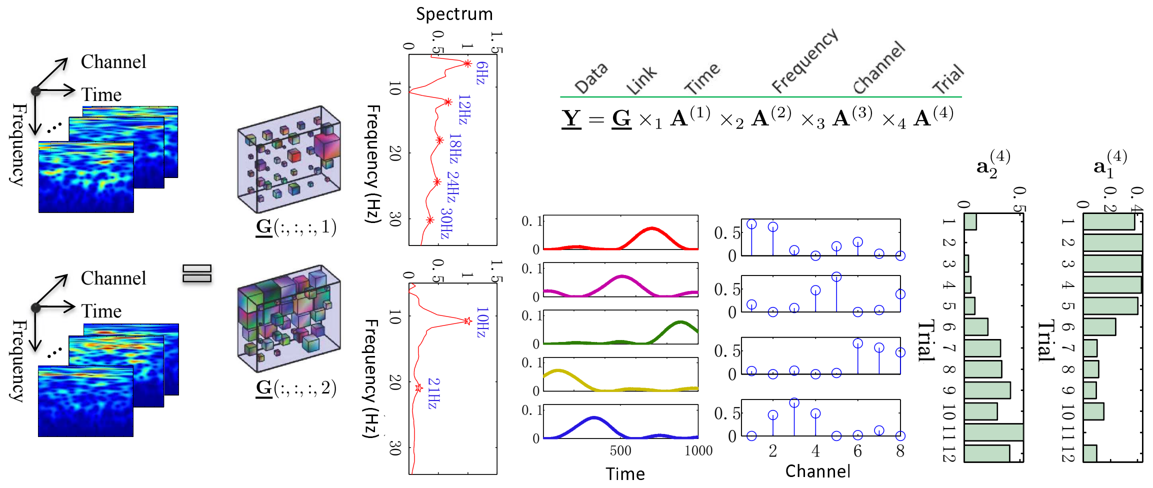

Channel
Time
Frequency
$\underline{\mathbf{G}}(:,:,:,1)$
$\underline{\mathbf{G}}(:,:,:,2)$
Spectrum
Frequency (Hz)
6Hz
12Hz
18Hz
24Hz
30Hz
10Hz
21Hz
Data
Link
Time
Frequency
Channel
Trial
$\underline{\mathbf{Y}} = \underline{\mathbf{G}} \times_1 \mathbf{A}^{(1)} \times_2 \mathbf{A}^{(2)} \times_3 \mathbf{A}^{(3)} \times_4 \mathbf{A}^{(4)}$
$\mathbf{a}_2^{(4)}$
$\mathbf{a}_1^{(4)}$
Time
Channel
Trial

Fig. 9. Example 4: Four-way EEG spectrum tensor (frequency × time × channel × trial) factorization by MBSS. The example includes 12 trials recorded from eight channels P7, P3, Pz, P4, P8, O1, Oz and O2 during 6.5 Hz and 10.5 Hz flickering visual stimulus (6 trials each). Frequency components between 5 Hz and 50 Hz with 0.5 Hz resolution (i.e., 91 frequency bins) were analyzed and the time window length was 4s (i.e., 1000 sample points). Each trial is represented by a 3-way tensor with dimension of 91 × 1000 × 8

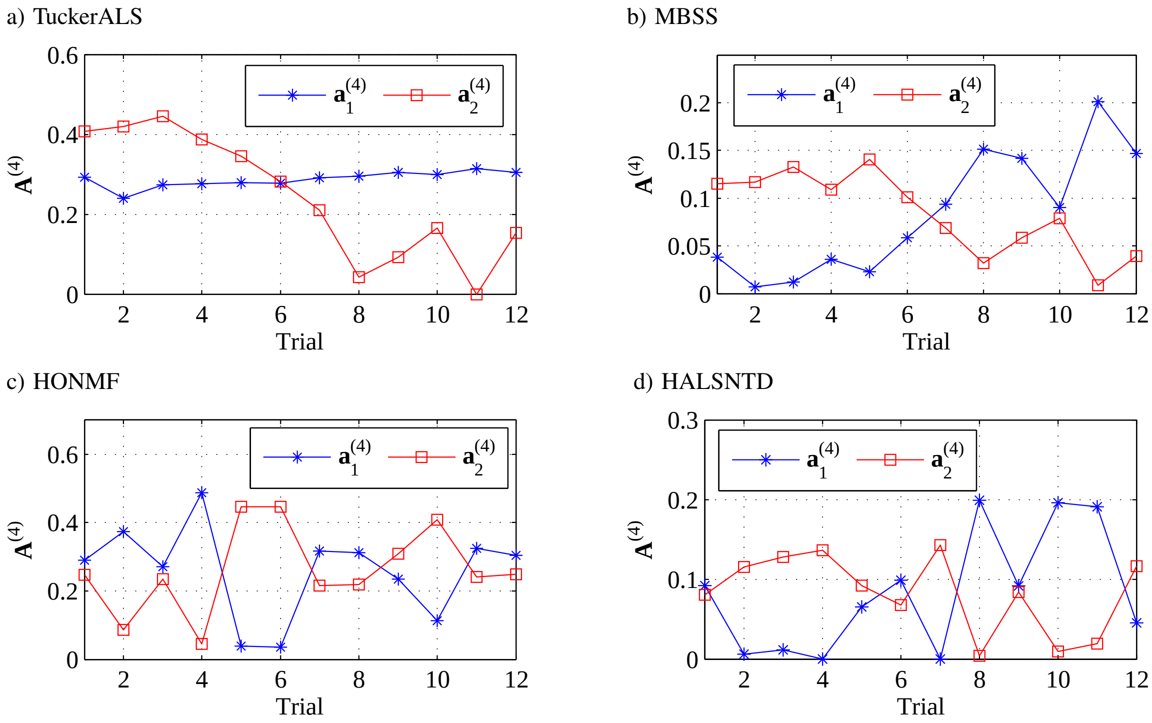


Fig. 10. Plots of vector $\mathbf{a}_1^{(4)}$ and $\mathbf{a}_2^{(4)}$ of the factor matrix $\mathbf{A}^{(4)} = [\mathbf{a}_1^{(4)}, \mathbf{a}_2^{(4)}]$ representing class information of the SSVEP for the TuckerALS, MBSS, HONMF and HALS algorithms. The first 6 trials correspond to the class 1 SSVEP with frequency 6.5 Hz and the next 6 trials correspond to the class 2 SSVEP with frequency 10.5 Hz. From these plots it is clear that the MBSS method provides, in this case, the best discriminative features

Their computation time and fit are shown in Table 4. It should be noted that the MBSS needs much less computation time in comparison to the others, and achieves comparable fit. The TuckerALS and the MBSS implemented here did not impose nonnegativity constraints on the core tensor. From the experimental results, by using the existing the DNN-MF method, the decomposition results obtained by MBSS are more interpretable and the such decompositions are also more efficient than NTD methods employing ALS approach. Here, our aim was not to show that the proposed MBSS can yield better performance than that of other commonly used methods for SSVEP analysis, but rather to show that the MBSS is quite promising in the real-world data analysis. Worth mentioning, this may also provide a novel way to analyze EEG data. Instead of usual two-way analysis (e.g., temporal-spatial analysis) for the EEG data, the MBSS approach attempts to extract the SSVEP components by multiway-array factorization from the four-way EEG spectrum tensor which integrates time-frequency features, temporal-spatial patterns and trial-to-trial variability. The MBSS can impose various constraints on the data in different ways according to the corresponding characteristics of the electrophysiological signals, and may provide more intuitive interpretations for the physical meaning of the signals, thereby assisting the follow-up signal analysis, such as the classification or detection in BCIs.

Table 4
Comparison of performances of 4 algorithms for the steady-state visual evoked potential (SSVEP) data analysis in terms of run time and fit

Algorithm	TuckerALS	MBSS	HALSNTD	HONMF
Runtime(s)	596	9	236	238
Fit	0.54	0.53	0.53	0.52

6. Conclusions

The existing methods for the constrained Tucker decompositions are often slow, stuck in local minima and do not provide unique desired decompositions and therefore, the results are difficult to interpret. In this paper, a simple approach based on the generalized multiway blind source separation (MBSS) is investigated. Using the MBSS approach described in detail, the component matrices, which are the subject of our main interest, are estimated directly by applying existing efficient BSS methods to each unfolding matrix of the data tensor. This approach leads to essentially unique (neglecting unavoidable scaling and permutation ambiguities of the components) Tucker decompositions with usually meaningful and physically interpretable results. In other words, we have demonstrated that by employing the MBSS approach, constrained or penalized Tucker decompositions provide essentially unique decompositions and thus the extracted components have specific statistical or deterministic properties (e.g., statistical independence, smoothness, sparseness and nonnegativity)[12]. The MBSS approach provides an attractive and efficient alternative to existing algorithms for unique tensor analysis. Note that in the MBSS approach the specific component matrices of interest can be extracted independently of other modes, directly from a reduced matricized data without the need to perform a full decomposition of the whole data tensor, e.g.,without the need to employ ALS algorithms which may fail to perform such decompositions, especially for ill-conditioned, highly collinear or noisy data. Simulations show the validity and efficiency of the proposed method, especially for large-scale problems.

In summary, the MBSS approach can be viewed as a very flexible and general technique for constrained or penalized tensor decompositions, which is an efficient alternative to many existing algorithms for tensor decompositions, especially to a wide class of the ALS algorithms. The extensive computer simulations confirmed that by using the MBSS approach, we can reduce the computing time at least by one or two orders of magnitude by comparison with ALS and HOOI algorithms under assumptions that some *a priori* knowledge about the hidden components is available. Moreover, for experimental, real world data, the MBSS often gives more meaningful components of hidden latent variables, with proper physical or physiological interpretations. However, if such kind of *a priori* knowledge is not available, the standard Tucker or CP (PARAFAC) decomposition algorithms should still be the first choices.

Acknowledgements. This work was in part supported by the National Natural Science Foundation of China (Grant 61103122, 61104053), and the China Postdoctoral Science Foundation funded project (Grant 201104360).

REFERENCES

[1] F. Hitchcock, "Multiple invariants and generalized rank of a p-way matrix or tensor", *J. Mathematics and Physics* 7, 39–79 (1927).

[2] L. Tucker, "Some mathematical notes on three-mode factor analysis", *Psychometrika* 31 (3), 279–311 (1966).

[3] L. De Lathauwer, B. De Moor, and J. Vandewalle, "A multilinear singular value decomposition", *SIAM J. on Matrix Analysis and Applications* 21, 1253–1278 (2000).

[4] T.G. Kolda and B.W. Bader, "Tensor decompositions and applications", *SIAM Review* 51 (3), 455–500 (2009).

[5] A. Cichocki, R. Zdunek, A.-H. Phan, and S. Amari, *Nonnegative Matrix and Tensor Factorizations: Applications to Exploratory Multi-way Data Analysis and Blind Source Separation*, Wiley, Chichester, 2009.

[6] A. Cichocki, "Tensors decompositions: new concepts for brain data analysis?", *J. Control, Measurement, and System Integration (SICE)* 7, 507–517 (2011).

[7] L. De Lathauwer, B. De Moor, and J. Vandewalle, "On the best rank-1 and rank-$(R_1, R_2, ..., R_n)$ approximation of higher-order tensors", *SIAM J. on Matrix Analysis and Applications* 21 (4), 1324–1342 (2000).

[8] F. Miwakeichi, E. Martnez-Montes, P. Valds-Sosa, N. Nishiyama, H. Mizuhara, and Y. Yamaguchi, "Decomposing EEG data into space-time-frequency components using parallel factor analysis", *NeuroImage* 22 (3), 1035–1045 (2004).

[12]Of course, some BSS algorithms can also stuck in local minima, but usually the MBSS approach alleviate this problem due to the reduction degree of freedom.

[9] Z. He, A. Cichocki, S. Xie, and K. Choi, "Detecting the number of clusters in n-way probabilistic clustering", *IEEE Trans. on Pattern Analysis and Machine Intelligence* 32 (11), 2006–2021 (2010).

[10] H.A. Phan and A. Cichocki, "Tensor decompositions for feature extraction and classification of high dimensional datasets", *Nonlinear Theory and Its Applications, IEICE* 1 (1), 37–68 (2010).

[11] R. Bro, "Multi-way analysis in the food industry: Models, algorithms & applications", *Food Technology* 309, http://curis.ku.dk/ws/files/13035961/Rasmus_Bro.pdf, (1998).

[12] A.H. Andersen and W.S. Rayens, "Structure-seeking multilinear methods for the analysis of fMRI data", *NeuroImage* 22 (2), 728–739 (2004).

[13] E. Martínez-Montes, P.A. Valdés-Sosa, F. Miwakeichi, R. Goldman, and M. Cohen, "Concurrent EEG/fMRI analysis by multiway partial least squares", *NeuroImage* 22 (3), 1023–1034 (2004).

[14] E. Acar, C. Aykut-Bingol, H. Bingol, R. Bro, and B. Yener, "Multiway analysis of epilepsy tensors", *Bioinformatics* 23 (13), i10–i18 (2007).

[15] M. De Vos, A. Vergult, and L. De Lathauwer, "Canonical decomposition of ictal scalp EEG reliably detects the seizure onset zone", *NeuroImage* 37 (3), 844–854 (2007).

[16] M. Mørup, L.K. Hansen, and S.M. Arnfred, "Algorithms for sparse nonnegative Tucker decompositions", *Neural Computation* 20 (8), 2112–2131 (2008).

[17] L. De Lathauwer, "A survey of tensor methods", *IEEE Int. Symposiumon Circuits and Systems (ISCAS)* 1, 2773–2776 (2009).

[18] V. De Silva and L.-H. Lim, "Tensor rank and the ill-posedness of the best low-rank approximation problem", *SIAM J. on Matrix Analysis and Applications* 30 (3), 1084–1127 (2008).

[19] S. Weiland and F. Van Belzen, "Singular value decompositions and low rank approximations of tensors", *IEEE Trans. on Signal Processing* 58 (3), 1171–1182 (2010).

[20] L. De Lathauwer, "A link between the canonical decomposition in multilinear algebra and simultaneous matrix diagonalization", *SIAM J. on Matrix Analysis and Applications* 28 (3), 642–666 (2006).

[21] C.F. Beckmann and S.M. Smith, "Tensorial extensions of independent component analysis for multisubject fMRI analysis", *NeuroImage* 25 (1), 294–311 (2005).

[22] M. De Vos, L. De Lathauwer, and S. Van Huffel, "Algorithm for imposing SOBI-type constraints on the CP model", *IEEE Int. Symp. Circuits and Systems (ISCAS).*, 2008, 1344–1347.

[23] M. Vasilescu and D. Terzopoulos, "Multilinear independent components analysis", *IEEE Computer Society Conf. on Computer Vision and Pattern Recognition* 1, 547–553 (2005).

[24] S. Unkel, A. Hannachi, N. Trendafilov, and I. Jolliffe, "Independent component analysis for three-way data with an application from atmospheric science", *J. Agricultural, Biological, and Environmental Statistics* 16, 319–338 (2011).

[25] A. Bell and T. Sejnowski, "An information-maximization approach to blind separation and blind deconvolution", *Neural Computation* 7, 1129–1159 (1995).

[26] A. Hyvarinen, J. Karhunen, and E. Oja, *Independent Component Analysis*, Wiley, New York, 2001.

[27] A. Cichocki and S. Amari, *Adaptive Blind Signal and Image Processing: Learning Algorithms and Applications*, John Wiley & Sons, London, 2002.

[28] A. Karfoul, L. Albera, and L. De Lathauwer, "Iterative methods for the canonical decomposition of multi-way arrays: Application to blind underdetermined mixture identification", *Signal Processing* 91 (8), 1789–1802 (2011).

[29] M. Zibulevsky and B. A. Pearlmutter, "Blind source separation by sparse decomposition in a signal dictionary", *Neural Computation* 13 (4), 863–882 (2001).

[30] P. Georgiev, F. Theis, and A. Cichocki, "Sparse component analysis and blind source separation of underdetermined mixtures", *IEEE Trans. on Neural Networks* 16 (4), 992–996 (2005).

[31] D.D. Lee and H.S. Seung, "Learning the parts of objects by non-negative matrix factorization", *Nature* 401 (6755), 788–791 (1999).

[32] B.W. Bader and T.G. Kolda, "MATLAB tensor toolbox version 2.5", http://csmr.ca.sandia.gov/ tgkolda/TensorToolbox/, (2012).

[33] G. Zhou and A. Cichocki, "Canonical polyadic decomposition based on a single mode blind source separation", *IEEE Signal Processing Letters* 19 (8), 523–526 (2012).

[34] X. Feng and Z. Zhang, "The rank of a random matrix", *Applied Mathematics and Computation* 185 (1), 689–694 (2007).

[35] F.M. Fisher, *The Identification Problem in Econometrics*, McGraw-Hill, New York, 1966.

[36] M. De Vos, D. Nion, S. Van Huffel, and L. De Lathauwer, "A combination of parallel factor and independent component analysis", *Tech. Rep.*, ftp://ftp.esat.kuleuven.ac.be/pub/sista/mdevos/reports/Ica_cp08.pdf (2008).

[37] D. Langers, "Unbiased group-level statistical assessment of independent component maps by means of automated retrospective matching", *Human Brain Mapping* 31, 727–742 (2010).

[38] A. Cichocki, "Generalized component analysis and blind source separation methods for analyzing mulitchannel brain signals", *Statistical and Process Models for Cognitive Neuroscience and Aging* 1, 201–272 (2007).

[39] Z. Lin, M. Chen, and Y. Ma, "The augmented Lagrange multiplier method for exact recovery of corrupted low-rank matrices", *ArXiv e-prints*, 1009.5055 (2010).

[40] P. Drineas, R. Kannan, and M.W. Mahoney, "Fast Monte Carlo algorithms for matrices II: computing a low-rank approximation to a matrix", *SIAM J. on Computing* 36 (1), 158–183 (2006).

[41] M.W. Mahoney and P. Drineas, "CUR matrix decompositions for improved data analysis", *Proc. Nat. Ac. Sciences* 106 (3), 697–702 (2009).

[42] C.F. Caiafa and A. Cichocki, "Generalizing the column-row matrix decomposition to multi-way arrays", *Linear Algebra and Its Applications* 433 (3), 557–573 (2010).

[43] E. Ceulemans and H.A.L. Kiers, "Selecting among three-mode principal component models of different types and complexities: a numerical convex hull based method", *British J. Mathematical and Statistical Psychology* 59 (1), 133–150 (2006).

[44] H.A.L. Kiers and A. der Kinderen, "A fast method for choosing the numbers of components in tucker3 analysis", *British J. Mathematical and Statistical Psychology* 56 (1), 119–125 (2003).

[45] M.O. Ulfarsson and V. Solo, "Dimension estimation in noisy pca with sure and random matrix theory", *IEEE Trans. on Signal Processing* 56 (12), 5804–5816 (2008).

[46] A. Cichocki, S. Amari, and K. Siwek, "ICALAB toolbox", http://www.bsp.brain.riken.jp/ICALAB (2007).

[47] Z. Koldovsky, P. Tichavsky, and E. Oja, "Efficient variant of

algorithm FastICA for independent component analysis attaining the Cramer-Rao lower bound", *IEEE Trans. on Neural Networks* 17 (5), 1265–1277 (2006).

[48] A. Belouchrani, K. AbedMeraim, and J.F. Cardoso, "A blind source separation technique using second-order statistics", *IEEE Trans. on Signal Processing* 45 (2), 434–444 (1997).

[49] C.A. Andersson and R. Bro, "The N-way toolbox for MATLAB", http://www.models.life.ku.dk/source/nwaytoolbox/ (2000).

[50] A.H. Phan and A. Cichocki, "Extended HALS algorithm for nonnegative Tucker decomposition and its applications for multiway analysis and classification", *Neurocomputing* 74 (11), 1956–1969 (2011).

[51] R. Zdunek, H.A. Phan, and A. Cichocki, "Damped Newton iterations for nonnegative matrix factorization", *Australian J. Intelligent Information Processing Systems* 12 (1), 16–22 (2010).

[52] G. Zhou, A. Cichocki, and S. Xie, "Fast nonnegative matrix/tensor factorization based on low-rank approximation", *IEEE Trans. on Signal Processing* 60 (6), 2928–2940 (2012).

[53] S.A. Nene, S.K. Nayar, and H. Murase, "Columbia object image library (COIL-20)", http://www.cs.columbia. edu/ CAVE/software/softlib/coil-20.php (1996).

[54] L. Van Der Maaten and C. Detection, "Visualizing data using t-SNE", *J. Machine Learning Research* 9, 2579–2605 (2008).

[55] Y. Yang, D. Xu, and F. Nie, "Image clustering using local discriminant models and global integration", *IEEE Trans. on Image Processing* 19 (10), 2761–2773 (2010).

[56] B.Z.Allison, D.J. McFarland, G. Schalk, S.D. Zheng, M. Jackson, and J.R. Wolpaw, "Towards an independent brain-computer interface using steady state visual evoked potentials", *Clinical Neurophysiology* 119 (2), 399–408 (2008).

[57] Y. Zhang, G. Zhou, Q. Zhao, A. Onishi, J. Jin, X. Wang, and A. Cichocki, "Multiway canonical correlation analysis for frequency components recognition in SSVEP-based BCIs", *Neural Information Processing*, Springer, Berlin, 2011.

[58] X. Gao, D. Xu, M. Cheng, and S. Gao, "A BCI-based environmental controller for the motion-disabled", *IEEE Trans. on Neural Systems and Rehabilitation Engineering* 11 (2), 137–140 (2003).

[59] Z. Wu and D. Yao, "Frequency detection with stability coefficient for steady-state visual evoked potential (SSVEP)-based BCIs", *J. Neural Engineering* 5 (1), 36 (2008).