Received: 20 February 2019 | Revised: 6 January 2020 | Accepted: 21 January 2020
DOI: 10.1002/nla.2291

RESEARCH ARTICLE

WILEY

A unitary joint diagonalization algorithm for nonsymmetric higher-order tensors based on Givens-like rotations

Jifei Miao[1] | **Guanghui Cheng**[1] | **Wenrui Li**[2] | **Eric Moreau**[3]

[1]School of Mathematical Sciences, University of Electronic Science and Technology of China, Chengdu, P.R. China

[2]School of Information and Software Engineering, University of Electronic Science and Technology of China, Chengdu, P.R. China

[3]Aix Marseille Université, CNRS, ENSAM, 13397 Marseille et Université de Toulon La Garde, 83957, France

Correspondence
Guanghui Cheng, School of Mathematical Sciences, University of Electronic Science and Technology of China, Chengdu, P.R. China.
Email: ghcheng@uestc.edu.cn

Summary

Based on Givens-like rotations, we present a unitary joint diagonalization algorithm for a set of nonsymmetric higher-order tensors. Each unitary rotation matrix only depends on one unknown parameter which can be analytically obtained in an independent way following a reasonable assumption and a complex derivative technique. It can serve for the canonical polyadic decomposition of a higher-order tensor with orthogonal factors. Furthermore, based on cross-high-order cumulants of observed signals, we show that the proposed algorithm can be applied to solve the joint blind source separation problem. The simulation results reveal that the proposed algorithm has a competitive performance compared with those of several existing related methods.

KEYWORDS

canonical polyadic decomposition, higher-order tensor, joint blind source separation, joint diagonalization

1 | INTRODUCTION

In the past few decades, there have been considerable works on tensor decompositions,[1–3] etc. Moreover, the joint diagonalization (JD) of a set of matrices or tensors has been recognized to be instrumental in signal processing, such as the blind source separation (BSS), image denoising, the independent component analysis (ICA), the multiple-input and multiple-output (MIMO) telecommunication systems, and so on.

The JD of a set of matrices has a long history, and there are a huge number of related literatures, such as References 4–7, and the JD of a set of tensors can naturally be seen as its generalization. Tensor diagonalization and its link with the ICA were first introduced in Reference 8. A BSS algorithm by the simultaneous third-order tensor diagonalization was proposed in Reference 9, which shows that for third-order tensors, the computation of an elementary Jacobi-rotation is equivalent to the best rank-1 approximation. A generalization to the statistics of any order greater than two was done in Reference 10 but mainly considered the real number field. In addition, Maurandi and Moreau proposed one of the first coordinate algorithms for the nonorthogonal JD of any order complex tensors in Reference 11, which relies on a particular decomposition of diagonalizing matrices. In all the above works, factor matrices are assumed to be identical for all modes, which are only suitable for the problem of the single-set BSS. However, with the increasing availability of multiset and multimodal signals, the conventional BSS approaches encounter huge challenges, since they are inherently developed to process single-set data and, hence, without sufficiently considering this interset dependence. Therefore, joint BSS (JBSS) algorithms have attracted great interests, which can simultaneously recover the underlying multiple variables from multiple datasets.[7] In addition, the JBSS can keep the extracted components aligned across different datasets,[12] which is an important feature that is not provided by the single-set BSS methods.

Some methods have been proposed to generalize the BSS to the JBSS. The multiset canonical correlation analysis (MCCA), for example, has been widely applied to discover associations across multiple datasets.[13] The Group ICA model that concatenates multiple datasets in the vertical dimension followed by a principal component analysis step.[14] Another way is to directly make use of the independent vector analysis (IVA) which generalizes the ICA to multiple datasets by exploring statistical dependences across datasets.[15,16] Moreover, the recent method of the JD of many cross-cumulate matrices also has been proposed in Reference 17, which is an extension of the canonical correlation analysis model. In this paper, we aim to solve the JBSS problem by generalizing the JD of traditional matrices to higher-order tensors and show its optimality by the simulations.

In addition, the proposed algorithm can serve for the canonical polyadic decomposition (CPD) of a higher-order tensor with orthogonal factors and we demonstrate this in the next section. In many applications, such as blind receiver design for shot for direct sequence-code division multiple access systems, linear image coding for a collection of images, etc., some of the factor matrices are known to have orthogonality structure, and this information can be exploited to improve the accuracy of latent factors recovery. We know that under rather mild conditions, the CPD is unique up to a trivial scalar and permutation ambiguity,[18] which has underlain its importance in signal processing.[9–11,19] Pioneering work[20] established the fact that the uniqueness of tensor CPD incorporating orthogonal factors is guaranteed under an even milder condition than the case without orthogonal factors. Mikael et al[20] proposed an alternating least-squares (ALS) algorithm for tensor CPD with orthogonal factors (ALS1-CPO) which extended the conventional methods to account for the orthogonality structure. For brevity, we call it the OALS algorithm in this paper, and we can see that the performance of the proposed algorithm is much better than that of the OALS in certain cases. Some orthogonal diagonalization algorithms of single tensor, such as that proposed by Comon and Sorensen,[21] are the special cases of our proposed algorithm when only one tensor is considered in the JD model. Even in this case (only one tensor is considered), the proposed algorithm still has competitive performance.

In summary, the main contributions of this paper are displayed as follows:

- Based on Givens-like rotations, we propose a unitary JD algorithm for a set of higher-order tensors in the general case that all the factor matrices are distinct, that is, the nonsymmetric JD algorithm. Different from some nonorthogonal algorithms, such as References 11 and 22, the matrix factors are constrained to be unitary, in the proposed algorithm, based on a crucial parameter structure of the unitary rotation matrix.
- We separate mixed source signals, based on the proposed algorithm, through jointly diagonalizing a set of time-delay cross-high-order cumulants established by observed signals (pre-whited) from multiple datasets, directly. To our best knowledge, this approach of the JBSS has not been explicitly discussed in the open literature. The simulation results demonstrate that the JBSS problem can be effectively solved by the proposed algorithm.

1.1 | Basic notations and operations

The notations used in this paper are very similar to References 18 and 23. Higher-order tensors can be interpreted as multiway arrays, which are denoted by calligraphic letters, for example, $\mathcal{X}$. The order of a tensor is the number of dimensions, also known as ways or modes. Matrices (second-order tensors) are denoted by boldface capital letters, for example, $\mathbf{A}$; vectors (first-order tensors) are denoted by boldface lowercase letters, for example, $\mathbf{b}$; and scalars are denoted by lowercase letters, for example, c. The $(i_1, \dots, i_L)$th element of a higher-order tensor $\mathcal{X} \in \mathbb{C}^{I_1 \times I_2 \times \dots \times I_L}$ is denoted by $x_{i_1 \dots i_L}$ or $\mathcal{X}(i_1, \dots, i_L)$, and the same is true for matrices. The kth component in a sequence is denoted by a superscript in a square bracket, for example, $\mathcal{X}^{[k]}$ denotes the kth tensor in a sequence. The complex conjugate is denoted by $(\cdot)^*$. $\mathbf{I}$ denotes the identity matrix with appropriate size. $|c|$, $\circ$, $\otimes$, $(\cdot)^\dagger$ and $(\cdot)^H$ denote the complex modulus of c, the outer product operation, the Kronecker product operator, the Moore-Penrose inverse operator and the conjugate transpose operator, respectively. $\|\mathcal{X}\|$ denotes the Frobenius norm of a tensor $\mathcal{X}$, that is, $\|\mathcal{X}\| = \sqrt{\sum_{i_1=1}^{I_1} \sum_{i_2=1}^{I_2} \cdots \sum_{i_L=1}^{I_L} |x_{i_1 \dots i_L}|^2}$. Besides, we let set $[\![K]\!] := \{1, \dots, K\}$ for all $K \in \mathbb{N}$. The mode-l product of a tensor $\mathcal{X} \in \mathbb{C}^{I_1 \times I_2 \times \dots \times I_L}$ with a matrix $\mathbf{B} \in \mathbb{C}^{J \times I_l}$ is denoted as $\mathcal{Y} = \mathcal{X} \times_l \mathbf{B}$. It is of the size $I_1 \times \dots \times I_{l-1} \times J \times I_{l+1} \times \dots \times I_L$, and $y_{i_1 \dots i_{l-1} j i_{l+1} \dots i_L} = \sum_{i_l=1}^{I_l} x_{i_1 i_2 \dots i_L} b_{j i_l}$.

The remainder of this paper is organized as follows. In Section 2, the problem of the JD for nonsymmetric higher-order tensors is formulated. In Section 3, we present the details of the analytical derivation for the updating matrices. In Section 4, numerical simulations are given to illustrate the performance of the proposed algorithm. Finally, some conclusions are drawn in Section 5.

2 | PROBLEM FORMULATION

We consider a set of K ($K \geq 1$) L-order N-dimensional tensors $\mathcal{X}^{[k]} \in \mathbb{C}^{N_1 \times N_2 \times \ldots \times N_N}$, $k \in [\![K]\!]$, with restriction that $N_1 = N_2 = \ldots = N_N = N$. For other cases (if $N_1, N_2, \ldots, N_N$ are different in some applications), the Tucker compression can be applied prior to the JD,[24] we would not consider such cases in this paper. The considered tensors all share the following latent common decomposition:

$$\mathcal{X}^{[k]} = \mathcal{D}^{[k]} \times_1 \mathbf{A}^{[1]} \times_2 \mathbf{A}^{[2]} \ldots \times_L \mathbf{A}^{[L]}, \quad k \in [[K]], \tag{1}$$

where $\mathcal{D}^{[k]} \in \mathbb{C}^{N \times N \times \ldots \times N}$ are K diagonal tensors, that is, $d^{[k]}_{i_1 \ldots i_L} \neq 0$ only if $i_1 = \ldots = i_L$, and $\mathbf{A}^{[l]}$ for all $l \in [\![L]\!]$ are unitary matrices , that is, $(\mathbf{A}^{[l]})^H \mathbf{A}^{[l]} = \mathbf{A}^{[l]} (\mathbf{A}^{[l]})^H = \mathbf{I}$, $l \in [\![L]\!]$ called the factor matrices or the mixing matrices in the context of the JBSS.

Remark 1. We stack the L-order target tensors $\mathcal{X}^{[k]}, k \in [\![K]\!]$, in a $(L+1)$-order tensor $\mathcal{F}$ as follows: $\mathcal{F}(i_1, i_2, \ldots, i_L, k) = \mathcal{X}^{[k]}(i_1, i_2, \ldots, i_L)$. If we define matrix $\mathbf{A}^{[L+1]} \in \mathbb{C}^{K \times N}$ with the elements $\mathbf{A}^{[L+1]}(k, n) = \mathcal{D}^{[k]}(n, n, \ldots, n), k \in [\![K]\!], n \in [\![N]\!]$, then* one can easily see that

$$\mathcal{F}(i_1, i_2, \ldots, i_L, k) = \sum_{n=1}^{N} \mathbf{A}^{[1]}(i_1, n) \mathbf{A}^{[2]}(i_2, n) \ldots \mathbf{A}^{[L]}(i_L, n) \mathbf{A}^{[L+1]}(k, n),$$
$$\Downarrow$$
$$\mathcal{F} = \sum_{n=1}^{N} \mathbf{A}^{[1]}(:, n) \circ \mathbf{A}^{[2]}(:, n) \circ \ldots \circ \mathbf{A}^{[L]}(:, n) \circ \mathbf{A}^{[L+1]}(:, n), \tag{2}$$

where $\mathbf{A}^{[l]}(:, n)$ denotes the nth column of $\mathbf{A}^{[l]}, l \in [\![L+1]\!]$. Model (2) clearly is the CPD of the $(L+1)$-order tensor $\mathcal{F}$ with L orthogonal factors $\mathbf{A}^{[1]}, \ldots, \mathbf{A}^{[L]}$.

In practice, the given data tensors $\mathcal{X}^{[k]}, k \in [\![K]\!]$ (typically by some sample statistics obtained through a finite number of data), one would like to identify L unitary matrices and K diagonal tensors in such a way that they are closed to the assumed model in (1). For that task, it is classical to introduce the following quadratic cost function:

$$\mathcal{J}(\{\tilde{\mathbf{A}}\}, \{\tilde{\mathcal{D}}\}) = \sum_{k=1}^{K} \| \mathcal{X}^{[k]} - \tilde{\mathcal{D}}^{[k]} \times_1 \tilde{\mathbf{A}}^{[1]} \times_2 \tilde{\mathbf{A}}^{[2]} \ldots \times_L \tilde{\mathbf{A}}^{[L]} \|^2. \tag{3}$$

For simplicity, in the above definition (3), $\{\tilde{\mathbf{A}}\} = \{\tilde{\mathbf{A}}^{[1]}, \tilde{\mathbf{A}}^{[2]}, \ldots, \tilde{\mathbf{A}}^{[L]}\}$ and $\{\tilde{\mathcal{D}}\} = \{\tilde{\mathcal{D}}^{[1]}, \tilde{\mathcal{D}}^{[2]}, \ldots, \tilde{\mathcal{D}}^{[K]}\}$ denote the set of the unitary matrix and diagonal tensor, respectively. Hence we would like to solve the following minimization problem:

$$(\{\hat{\mathbf{A}}\}, \{\hat{\mathcal{D}}\}) = \arg \min_{\{\tilde{\mathbf{A}}\}, \{\tilde{\mathcal{D}}\}} \mathcal{J}(\{\tilde{\mathbf{A}}\}, \{\tilde{\mathcal{D}}\}). \tag{4}$$

For that, $\mathcal{J}(\{\tilde{\mathbf{A}}\}, \{\tilde{\mathcal{D}}\})$ is first minimized with respect to the diagonal tensors $\tilde{\mathcal{D}}^{[k]}, k \in [\![K]\!]$ assuming fixed the unitary matrices $\tilde{\mathbf{A}}^{[l]}, l \in [\![L]\!]$. For a simple derivation, we need the two following results whose proofs are reported in the Appendix A1. Given a tensor $\mathcal{X}$, if $\tilde{\mathbf{A}}^{[l]}$ for all $l \in [\![L]\!]$ are unitary matrices then the two hereafter relations hold:

$$\| \mathcal{X} \times_1 \tilde{\mathbf{A}}^{[1]} \times_2 \tilde{\mathbf{A}}^{[2]} \ldots \times_L \tilde{\mathbf{A}}^{[L]} \|^2 = \| \mathcal{X} \|^2, \tag{5}$$

$$(\mathcal{X} \times_1 \tilde{\mathbf{A}}^{[1]} \times_2 \tilde{\mathbf{A}}^{[2]} \ldots \times_L \tilde{\mathbf{A}}^{[L]}) \times_1 (\tilde{\mathbf{A}}^{[1]})^H \times_2 (\tilde{\mathbf{A}}^{[2]})^H \ldots \times_L (\tilde{\mathbf{A}}^{[L]})^H = \mathcal{X}. \tag{6}$$

Recalling that if matrix $\tilde{\mathbf{A}}^{[l]}$ is unitary then $(\tilde{\mathbf{A}}^{[l]})^H$ is unitary too, using (5) and (6), $\mathcal{J}(\{\tilde{\mathbf{A}}\}, \{\tilde{\mathcal{D}}\})$ can be rewritten as

$$\mathcal{J}(\{\tilde{\mathbf{A}}\}, \{\tilde{\mathcal{D}}\}) = \sum_{k=1}^{K} \| \mathcal{X}^{[k]} \times_1 (\tilde{\mathbf{A}}^{[1]})^H \times_2 (\tilde{\mathbf{A}}^{[2]})^H \ldots \times_L (\tilde{\mathbf{A}}^{[L]})^H - \tilde{\mathcal{D}}^{[k]} \|^2. \tag{7}$$

*For a sake of simplicity and readability, we only considered in the real domain.

Let us remark that the minimization of $\mathcal{J}(\{\tilde{\mathbf{A}}\},\{\tilde{\mathcal{D}}\})$ with respect to the diagonal tensors $\tilde{\mathcal{D}}^{[k]}, k \in [\![K]\!]$ is equivalent to the minimization of each individual terms of the sum. Then for a given value of k and for the fixed matrices $\tilde{\mathbf{A}}^{[l]}, l \in [\![L]\!]$, the optimal value of $\tilde{\mathcal{D}}^{[k]}$ denoted $\hat{\mathcal{D}}^{[k]}$ minimizing $\mathcal{J}(\{\tilde{\mathbf{A}}\},\{\tilde{\mathcal{D}}\})$ in (7) can now be directly derived as

$$\hat{\mathcal{D}}^{[k]} = \mathsf{Tdiag}\{\mathcal{X}^{[k]}\times_1(\tilde{\mathbf{A}}^{[1]})^H\times_2(\tilde{\mathbf{A}}^{[2]})^H \dots \times_L(\tilde{\mathbf{A}}^{[L]})^H\}, \tag{8}$$

where $\mathsf{Tdiag}\{\cdot\}$ is the operator that sets all the off-diagonal elements of a tensor to zero, that is, $\mathsf{Tdiag}\{\mathcal{X}\}(i_1,\dots,i_L) = x_{i_1\dots i_L}\delta_{i_1\dots i_L}$ where $\delta_{i_1\dots i_L} = 1$ if $i_1 = \dots = i_L$ and 0 otherwise. Now, substituting $\tilde{\mathcal{D}}^{[k]}$ by $\hat{\mathcal{D}}^{[k]}$ in (7) leads to

$$\mathcal{J}(\{\tilde{\mathbf{A}}\},\{\hat{\mathcal{D}}\}) = \sum_{k=1}^{K} \|\mathcal{X}^{[k]}\times_1(\tilde{\mathbf{A}}^{[1]})^H\times_2(\tilde{\mathbf{A}}^{[2]})^H \dots \times_L(\tilde{\mathbf{A}}^{[L]})^H - \hat{\mathcal{D}}^{[k]}\|^2. \tag{9}$$

And we have

$$\begin{aligned}\mathcal{J}(\{\tilde{\mathbf{A}}\},\{\hat{\mathcal{D}}\}) &= \sum_{k=1}^{K} \|\mathcal{X}^{[k]}\times_1(\tilde{\mathbf{A}}^{[1]})^H\times_2(\tilde{\mathbf{A}}^{[2]})^H \dots \times_L(\tilde{\mathbf{A}}^{[L]})^H\|^2 - \sum_{k=1}^{K}\|\hat{\mathcal{D}}^{[k]}\|^2.\\ &= \sum_{k=1}^{K} \|\mathsf{ZTdiag}\{\mathcal{X}^{[k]}\times_1(\tilde{\mathbf{A}}^{[1]})^H\times_2(\tilde{\mathbf{A}}^{[2]})^H \dots \times_L(\tilde{\mathbf{A}}^{[L]})^H\}\|^2\\ &\triangleq \mathcal{J}((\tilde{\mathbf{A}}^{[1]})^H,(\tilde{\mathbf{A}}^{[2]})^H,\dots,(\tilde{\mathbf{A}}^{[L]})^H),\end{aligned}$$

where $\mathsf{ZTdiag}\{\cdot\}$ is the operator that sets all the diagonal elements of a tensor to zero, that is, $\mathsf{ZTdiag}\{\mathcal{X}\}(i_1,\dots,i_L) = x_{i_1\dots i_L}(1-\delta_{i_1\dots i_L})$ where $\delta_{i_1\dots i_L} = 1$ if $i_1 = \dots = i_L$ and 0 otherwise. Thus, the minimization of $\mathcal{J}(\{\tilde{\mathbf{A}}\},\{\hat{\mathcal{D}}\})$ is equivalent to the minimization of $\mathcal{J}((\tilde{\mathbf{A}}^{[1]})^H,(\tilde{\mathbf{A}}^{[2]})^H,\dots,(\tilde{\mathbf{A}}^{[L]})^H)$. To more clearly, the $(\tilde{\mathbf{A}}^{[l]})^H$ are labeled as $\mathbf{B}^{[l]}$ for all $l \in [\![L]\!]$ and

$$\begin{aligned}\hat{\mathcal{X}}^{[k]} &\triangleq \mathcal{X}^{[k]}\times_1(\tilde{\mathbf{A}}^{[1]})^H\times_2(\tilde{\mathbf{A}}^{[2]})^H \dots \times_L(\tilde{\mathbf{A}}^{[L]})^H\\ &= \mathcal{X}^{[k]}\times_1\mathbf{B}^{[1]}\times_2\mathbf{B}^{[2]}\dots\times_L\mathbf{B}^{[L]}.\end{aligned}$$

Then, we have

$$\begin{aligned}\mathcal{J}((\tilde{\mathbf{A}}^{[1]})^H,(\tilde{\mathbf{A}}^{[2]})^H,\dots,(\tilde{\mathbf{A}}^{[L]})^H) &= \sum_{k=1}^{K}\|\mathsf{ZTdiag}\{\hat{\mathcal{X}}^{[k]}\}\|^2\\ &\triangleq \mathcal{J}(\mathbf{B}^{[1]},\dots,\mathbf{B}^{[L]}).\end{aligned}$$

Hence, the goal is to find optimal matrices $\tilde{\mathbf{B}}^{[l]}$, for all $l \in [\![L]\!]$ in such a way that

$$(\tilde{\mathbf{B}}^{[1]},\dots,\tilde{\mathbf{B}}^{[L]}) = \arg\min_{(\mathbf{B}^{[1]},\dots,\mathbf{B}^{[L]})\in\mathfrak{B}} \mathcal{J}(\mathbf{B}^{[1]},\dots,\mathbf{B}^{[L]}),$$

where $\mathfrak{B}$ is the set of unitary matrices. In other words, from only data tensors $\mathcal{X}^{[k]}, k \in [\![K]\!]$, we just need to estimate L unitary matrices $\tilde{\mathbf{B}}^{[l]}$ called the diagonalizing matrices or the demixing matrices in the context of the JBSS, for all $l \in [\![L]\!]$ such that all the transformed tensors

$$\hat{\mathcal{X}}^{[k]} = \mathcal{X}^{[k]}\times_1\tilde{\mathbf{B}}^{[1]}\times_2\tilde{\mathbf{B}}^{[2]}\dots\times_L\tilde{\mathbf{B}}^{[L]}, \quad k \in [\![K]\!], \tag{10}$$

are jointly as diagonal as possible (in a given sense). Hence, ideally, $\tilde{\mathbf{B}}^{[l]}$ in (10) equals the inverse of $\mathbf{A}^{[l]}$ in (1) up to a permutation matrix for all $l \in [\![L]\!]$. Note that the factor matrices $\mathbf{A}^{[l]}$ in (1) (or the diagonalizing matrices $\tilde{\mathbf{B}}^{[l]}$ in (10)) for all $l \in [\![L]\!]$ are not necessarily identical, and we even assume that they are different from each other.

3 | PROPOSED ALGORITHM

We consider the following Jacobi-like procedure that consists of decomposing the overall $N \times N$ matrices $\mathbf{B}^{[l]}$, for all $l \in [\![L]\!]$ as the product of $\frac{N(N-1)}{2}$ elementary updating matrices $\mathbf{B}_{ij}^{[l]}$ $(1 \le i < j \le N)$ at each "sweep" and does the following update of $\mathbf{B}^{[l]}$ for all $l \in [\![L]\!]$:

$$\mathbf{B}^{[l]} \leftarrow \mathbf{B}^{[l]} \prod_{i=1}^{N-1} \prod_{j=i+1}^{N} \mathbf{B}_{ij}^{[l]}, \quad l \in [\![L]\!]. \tag{11}$$

One updating process of (11) for all the (i,j) pair $(1 \le i < j \le N)$ is called one sweep. Several sweeps are generally necessary to reach convergence. Now, for a given (i,j) position, the key is to find optimal elementary updating matrices $\mathbf{B}_{ij}^{[l]}$, for all $l \in [\![L]\!]$ minimizing $\mathcal{J}(\mathbf{B}_{ij}^{[1]}, \dots, \mathbf{B}_{ij}^{[L]})$.

For a fixed (i,j) $(1 \le i < j \le N)$ position, we consider the following parameter structure of $\mathbf{B}_{ij}^{[l]}$ for all $l \in [\![L]\!]$, which equal to the identity matrix but for the following four elements:

$$\begin{cases} \mathbf{B}_{ij}^{[l]}(i,j) = \lambda^{[l]}(\vartheta^{[l]})^*, \\ \mathbf{B}_{ij}^{[l]}(j,i) = -\lambda^{[l]}\vartheta^{[l]}, \\ \mathbf{B}_{ij}^{[l]}(i,i) = \mathbf{B}_{ij}^{[l]}(j,j) = \lambda^{[l]}, \end{cases} \tag{12}$$

where $\vartheta^{[l]}$ is an unknown parameter and $\lambda^{[l]} = (1 + |\vartheta^{[l]}|^2)^{-1/2}$ for all $l \in [\![L]\!]$. It is easy to see that $\mathbf{B}_{ij}^{[l]}$ for all $l \in [\![L]\!]$ are unitary matrices. The set of target tensors, for a fixed (i,j) $(1 \le i < j \le N)$ position, is updated as

$$\hat{\mathcal{X}}^{[k]} = \mathcal{X}^{[k]} \times_1 \mathbf{B}_{ij}^{[1]} \times_2 \mathbf{B}_{ij}^{[2]} \dots \times_L \mathbf{B}_{ij}^{[L]}, \quad k \in [\![K]\!]. \tag{13}$$

We denote $\mathcal{J}(\mathbf{B}_{ij}^{[1]}, \dots, \mathbf{B}_{ij}^{[L]}) = \sum_{k=1}^{K} ||\mathrm{ZTdiag}\{\hat{\mathcal{X}}^{[k]}\}||^2$. Although $\mathcal{J}(\mathbf{B}_{ij}^{[1]}, \dots, \mathbf{B}_{ij}^{[L]})$ is not convex itself, it is convex with respect to each single updating matrix. Hence, a common practice is to estimate each $\mathbf{B}_{ij}^{[l]}$ independently. This means that we estimate only one updating matrix each time while keeping the remaining ones fixed.

Without loss of generality, we first derive the optimal solution for $\mathbf{B}_{ij}^{[1]}$ setting $\mathbf{B}_{ij}^{[2]}, \dots, \mathbf{B}_{ij}^{[L]}$ to the identity matrix. Thus, (13) becomes

$$\hat{\mathcal{X}}_1^{[k]} = \mathcal{X}^{[k]} \times_1 \mathbf{B}_{ij}^{[1]}, \quad k \in [\![K]\!]. \tag{14}$$

Let the set $\mathcal{H} = [\![N]\!] \backslash \{i,j\}$, after straightforward derivations, we can obtain the elements of $\mathcal{T}_1^{[k]}$ as follows:

$$\begin{cases} \hat{\mathcal{X}}_1^{[k]}(i, n_1, \dots, n_{L-1}) = \lambda^{[1]} x_{in_1 \dots n_{L-1}}^{[k]} + \lambda^{[1]}(\vartheta^{[1]})^* x_{jn_1 \dots n_{L-1}}^{[k]}, \\ \hat{\mathcal{X}}_1^{[k]}(j, n_1, \dots, n_{L-1}) = \lambda^{[1]} x_{jn_1 \dots n_{L-1}}^{[k]} - \lambda^{[1]}\vartheta^{[1]} x_{in_1 \dots n_{L-1}}^{[k]}, \\ \hat{\mathcal{X}}_1^{[k]}(h, n_1, \dots, n_{L-1}) = x_{hn_1 \dots n_{L-1}}^{[k]}, \end{cases} \tag{15}$$

with $(n_1, \dots, n_{L-1}) \in [\![N]\!]^{L-1}$ and $h \in \mathcal{H}$. Then, $\mathcal{J}(\mathbf{B}_{ij}^{[1]}, \dots, \mathbf{B}_{ij}^{[L]})$ can be rewritten as

$$\begin{aligned} \mathcal{J}(\mathbf{B}_{ij}^{[1]}) = \sum_{k=1}^{K} \Bigg(& \sum_{(n_1,\dots,n_{L-1}) \in \mathcal{O}} |\hat{\mathcal{X}}_1^{[k]}(i, n_1, \dots, n_{L-1})|^2 \\ & + \sum_{(n_1,\dots,n_{L-1}) \in \mathcal{P}} |\hat{\mathcal{X}}_1^{[k]}(j, n_1, \dots, n_{L-1})|^2 \Bigg) + c_0, \end{aligned} \tag{16}$$

where the sets $\mathcal{O} = \{(n_1, \dots, n_{L-1}) | (n_1, \dots, n_{L-1}) \in \left([\![N]\!]^{L-1} \backslash (i, \dots, i)\right)\}$ and $\mathcal{P} = \{(n_1, \dots, n_{L-1}) | (n_1, \dots, n_{L-1}) \in \left([\![N]\!]^{L-1} \backslash (j, \dots, j)\right)\}$, c_0 is a positive constant that does not depend on $\vartheta^{[1]}$. Using (15) in (16), one can find that it is difficult to obtain an analytical solution, if we directly optimize $\mathcal{J}(\mathbf{B}_{ij}^{[1]})$ with respect to the $\vartheta^{[1]}$ in (16). Hence, we consider a reasonable approximation as in References 25 and 26, etc. Suppose we are closing enough to a diagonalizing solution in the sense that all off-diagonal elements of $\mathcal{X}^{[k]}(k \in [\![K]\!])$ have a very small magnitude, that is, we have

$$\left|x^{[k]}_{n_1,\ldots,n_L}\right| \ll 1, \quad k \in [\![K]\!], \tag{17}$$

where all $(n_1, \ldots, n_L) \in \{[\![N]\!]^L \backslash \{(n, \ldots, n)|n \in [\![N]\!]\}\}$. One should note that during the iterating procedure it is expected that the elementary updating matrices $\mathbf{B}^{[l]}_{ij}$ for all $l \in [\![L]\!]$ get closer and closer to the identity matrix. The similar assumption was adopted in References 11 and 25. It is then rather easy to show that

$$|\vartheta^{[l]}| \ll 1, \quad l \in [\![L]\!]. \tag{18}$$

Therefore, based on (17) and (18), we ignore both the high-order (≥ 2-order) terms of $x^{[k]}_{n_1,\ldots,n_L}(k \in [\![K]\!])$ for all $(n_1, \ldots, n_L) \in \{[\![N]\!]^L \backslash \{(n, \ldots, n)|n \in [\![N]\!]\}\}$ and $\vartheta^{[1]}$ in the elements of the tensor $\hat{\mathcal{X}}^{[k]}_1$ (simplified as $\tilde{\mathcal{X}}^{[k]}_1$) for all $k \in [\![K]\!]$. As a result, we can obtain

$$\begin{cases} \tilde{\mathcal{X}}^{[k]}_1(i, n_1, \ldots, n_{L-1}) = x^{[k]}_{ij\ldots j} + (\vartheta^{[1]})^* x^{[k]}_{jj\ldots j}, \\ \tilde{\mathcal{X}}^{[k]}_1(j, n_1, \ldots, n_{L-1}) = x^{[k]}_{ji\ldots i} - \vartheta^{[1]} x^{[k]}_{ii\ldots i}, \\ \tilde{\mathcal{X}}^{[k]}_1(h, n_1, \ldots, n_{L-1}) = x^{[k]}_{hn_1\ldots n_{L-1}}, \end{cases} \tag{19}$$

with $(n_1, \ldots, n_{L-1}) \in [\![N]\!]^{L-1}$ and $h \in \mathcal{H}$. We denote

$$\mathcal{J}_a(\mathbf{B}^{[1]}_{ij}) = \sum_{k=1}^{K} \|\mathrm{ZTdiag}\{\tilde{\mathcal{X}}^{[k]}_1\}\|^2,$$

Equation (16) can be simplified as

$$\mathcal{J}_a(\mathbf{B}^{[1]}_{ij}) = \sum_{k=1}^{K} \left(|x^{[k]}_{ij\ldots j} + (\vartheta^{[1]})^* x^{[k]}_{jj\ldots j}|^2 + |x^{[k]}_{ji\ldots i} - \vartheta^{[1]} x^{[k]}_{ii\ldots i}|^2 \right) + c_0. \tag{20}$$

Consequently, the goal is to find an optimal $\vartheta^{[1]}$ minimizing $\mathcal{J}_a(\mathbf{B}^{[1]}_{ij})$ in (20). For this we use complex derivative techniques, see, for example References 25 and 27. We first have

$$\begin{aligned} \frac{\partial \mathcal{J}_a(\mathbf{B}^{[1]}_{ij})}{\partial (\vartheta^{[1]})^*} &= \left(\frac{\partial \mathcal{J}_a(\mathbf{B}^{[1]}_{ij})}{\partial \vartheta^{[1]}} \right)^* \\ &= \vartheta^{[1]} \left(\sum_{k=1}^{K} \left(|x^{[k]}_{ii\ldots i}|^2 + |x^{[k]}_{jj\ldots j}|^2 \right) \right) - \sum_{k=1}^{K} \left(x^{[k]}_{ji\ldots i} (x^{[k]}_{ii\ldots i})^* - (x^{[k]}_{ij\ldots j})^* x^{[k]}_{jj\ldots j} \right), \end{aligned} \tag{21}$$

and

$$\frac{\partial^2 \mathcal{J}_a(\mathbf{B}^{[1]}_{ij})}{\partial \vartheta^{[1]} \partial (\vartheta^{[1]})^*} = \sum_{k=1}^{K} \left(|x^{[k]}_{ii\ldots i}|^2 + |x^{[k]}_{jj\ldots j}|^2 \right) \geq 0. \tag{22}$$

Hence, letting Equation (21) equal to zero, that is, $\frac{\partial \mathcal{J}_a(\mathbf{B}^{[1]}_{ij})}{\partial (\vartheta^{[1]})^*} = 0$, we can obtain the following optimal solution of $\vartheta^{[1]}$ analytically:

$$\vartheta^{[1]} = \frac{\sum_{k=1}^{K} \left(x^{[k]}_{ji\ldots i} \left(x^{[k]}_{ii\ldots i} \right)^* - \left(x^{[k]}_{ij\ldots j} \right)^* x^{[k]}_{jj\ldots j} \right)}{\sum_{k=1}^{K} \left(|x^{[k]}_{ii\ldots i}|^2 + |x^{[k]}_{jj\ldots j}|^2 \right)}. \tag{23}$$

TABLE 1 The nonsymmetric higher-order tensors algorithm

Require: $\mathcal{X}^{[k]} \in \mathbb{C}^{N\times N\times\cdots\times N},\ k \in [\![K]\!]$ (target tensors to be jointly diagonalized)
1: **Initial** $\mathbf{B}^{[1]}, \ldots, \mathbf{B}^{[L]} \in \mathbb{C}^{N\times N}$ are unitary matrices.
2: $\mathcal{X}^{[k]} \longleftarrow \mathcal{X}^{[k]} \times_1 \mathbf{B}^{[1]} \times_2 \mathbf{B}^{[2]} \ldots \times_L \mathbf{B}^{[L]}$ for all $k \in [\![K]\!]$.
3: **Repeat**
4: **for** $i = 1, \cdots, N-1$ **do**
5: **for** $j = i+1, \cdots, N$ **do**
6: Compute parameters $\vartheta^{[1]}, \ldots, \vartheta^{[L]}$ based on (23) and (24), then construct unitary matrices $\mathbf{B}_{ij}^{[1]}, \ldots, \mathbf{B}_{ij}^{[L]}$ based on (12).
7: $\mathbf{B}^{[l]} \longleftarrow \mathbf{B}_{ij}^{[l]}\mathbf{B}^{[l]}$, for all $l \in [\![L]\!]$.
8: $\mathcal{X}^{[k]} \longleftarrow \mathcal{X}^{[k]} \times_1 \mathbf{B}_{ij}^{[1]} \times_2 \mathbf{B}_{ij}^{[2]} \ldots \times_L \mathbf{B}_{ij}^{[L]}$ for all $k \in [\![K]\!]$.
9: **end for**
10: **end for**
11: **Until convergence**
Ensure: $\mathbf{B}^{[1]}, \ldots, \mathbf{B}^{[L]}$ and $\mathcal{X}^{[k]}$ for all $k \in [\![K]\!]$.

Then, the optimal $\mathbf{B}_{ij}^{[1]}$ is obtained based on (23). The derivation processes of the $\vartheta^{[2]}, \ldots, \vartheta^{[L]}$ are the same as that of $\vartheta^{[1]}$. They all can be simply deduced by index permutations of $\vartheta^{[1]}$ in (23), that is,

$$\begin{cases} \vartheta^{[2]} = \dfrac{\sum_{k=1}^{K}\left(x_{ij\ldots i}^{[k]}\left(x_{ii\ldots i}^{[k]}\right)^{*} - \left(x_{ji\ldots j}^{[k]}\right)^{*} x_{jj\ldots j}^{[k]}\right)}{\sum_{k=1}^{K}\left(\left|x_{ii\ldots i}^{[k]}\right|^{2} + \left|x_{jj\ldots j}^{[k]}\right|^{2}\right)}, \\ \quad\vdots \\ \vartheta^{[L]} = \dfrac{\sum_{k=1}^{K}\left(x_{ii\ldots j}^{[k]}(x_{ii\ldots i}^{[k]})^{*} - (x_{jj\ldots i}^{[k]})^{*} x_{jj\ldots j}^{[k]}\right)}{\sum_{k=1}^{K}\left(\left|x_{ii\ldots i}^{[k]}\right|^{2} + \left|x_{jj\ldots j}^{[k]}\right|^{2}\right)}. \end{cases} \tag{24}$$

Then, $\mathbf{B}_{ij}^{[2]}, \ldots, \mathbf{B}_{ij}^{[L]}$ can be obtained in a sequential or parallel way. As a result, we have established a unitary JD algorithm for a set of **no**n-symmetric **h**igher-order **t**ensors. The algorithm is called the NOHTJD, and shown in Table 1.

In the line 1 of Table 1, $\mathbf{B}^{[1]}, \ldots, \mathbf{B}^{[L]}$ are initialized as good guesses† for them or the identity matrix. Performing line 4 for all $j > i$ once is called a sweep or an iteration. The stopping criterion that we adopted is the difference (a given threshold like $1e^{-6}$ which we set in the simulation) between the values of the cost function $\mathcal{J}(\mathbf{B}^{[1]}, \ldots, \mathbf{B}^{[L]})$ in two consecutive sweeps.

We briefly analyze the computational complexity (flops) within one sweep for the NOHTJD algorithm shown in Table 1. When computing parameters $\vartheta^{[1]}, \ldots, \vartheta^{[L]}$ based on (23) and (24), the computational cost is about $\mathcal{O}(KLN^2)$. Then the computational cost of updating $\mathbf{B}^{[1]}, \ldots, \mathbf{B}^{[L]}$ in line 7 is about $\mathcal{O}(LN^3)$. Finally, the computational cost of updating K target tensors $\mathcal{X}^{[k]}, k \in [\![K]\!]$ in line 8 is about $\mathcal{O}(KLN^{(L+1)})$. Hence, the total computational cost of the NOHTJD algorithm at each sweep is about $\mathcal{O}(KLN^{(L+1)})$.

4 | SIMULATION RESULTS

In this section, the performance of the proposed algorithm is evaluated based on the results of simulations. We adopt the following widely used performance index PI,[11,28,29] *etc.*:

$$PI\left(G^{[1]}, \ldots, G^{[L]}\right) = \frac{1}{L}\{pi(G^{[1]})+, \ldots, +pi(G^{[L]})\},$$

†Actually, the proposed algorithm does not necessary to rely on any better initialization, $\mathbf{B}^{[1]}, \ldots, \mathbf{B}^{[L]}$ can be initialized as any random unitary matrices or generally by using identity matrix.

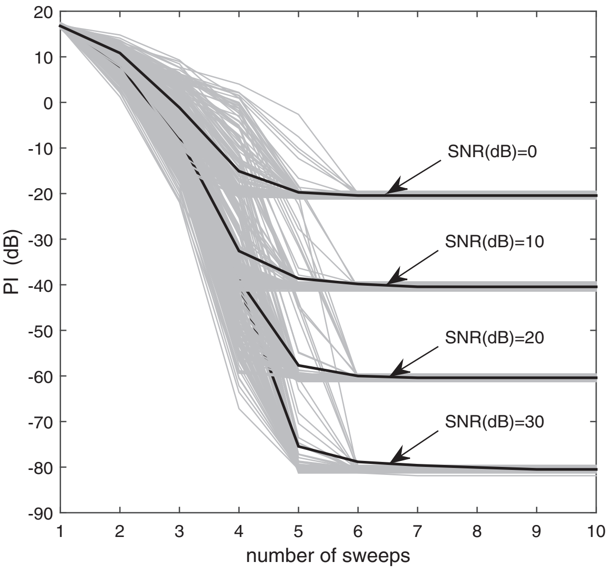


FIGURE 1 Performance index values of the nonsymmetric higher-order tensors algorithm with different noise level cases when $K = N = 10$. In gray, 100 individual realizations are plotted and in black the mean over these realizations

where

$$pi(\mathbf{G}^{[l]}) = \frac{1}{N(N-1)} \sum_{i=1}^{N} \left(\sum_{j=1}^{N} \frac{|g_{ij}^{[l]}|^2}{\max_s |g_{is}^{[l]}|^2} - 1 \right) + \frac{1}{N(N-1)} \sum_{j=1}^{N} \left(\sum_{i=1}^{N} \frac{|g_{ij}^{[l]}|^2}{\max_s |g_{sj}^{[l]}|^2} - 1 \right),$$

and $\mathbf{G}^{[l]} = (g_{ij})$ is an $N \times N$ matrix, called the global matrix. It is obvious that $PI \geq 0$, and the equality only holds when the matrix $\mathbf{G}$ is equal to the product of a diagonal matrix and a permutation matrix.

Simulation 1. We intend to illustrate the robustness and the convergence[‡] of the proposed algorithm in different scenarios. The target tensors are generated by using the model (1). Without loss of generality, we consider a set of fourth-order tensors, that is, $L = 4$. Four mixing matrices $\mathbf{A}^{[l]}, l = 1, \dots, 4$, are randomly generated unitary ones in $\mathbb{C}^{N\times N}$, respectively, obtained through a QR-decomposition of random matrices whose elements (including their real and imaginary parts) are i.i.d. and follow the standard normal distribution $\mathcal{N}(0, 1)$. The diagonal elements of $\mathcal{D}^{[k]}, k \in [\![K]\!]$ are also i.i.d. and follow the standard normal distribution $\mathcal{N}(0, 1)$ for both their real parts and imaginary parts. Finally, the noisy observation tensors are generated as $\check{\mathcal{X}}^{[k]} = \mathcal{X}^{[k]} + \delta^{[k]}\mathcal{E}^{[k]}, k \in [\![K]\!]$, where the elements of error tensors $\mathcal{E}^{[k]}, k \in [\![K]\!]$ are drawn from the standard normal distribution $\mathcal{N}(0, 1)$ for both their real parts and imaginary parts. The coefficients $\delta^{[k]} \geq 0$ using to control the noise level, are measured by the following signal to noise ratio (SNR) expressed in dB:[7,30]

$$SNR(dB) = 10log_{10} \left(\frac{\|\mathcal{X}^{[k]}\|}{\|\delta^{[k]}\mathcal{E}^{[k]}\|} \right).$$

Let $\mathbf{G}^{[l]} = \mathbf{B}^{[l]}\mathbf{A}^{[l]}, l = 1, \dots, 4$, and let the initial diagonalizing matrices $\mathbf{B}^{[l]}, l = 1, \dots, 4$, be the identity one.

Figures 1 and 2 show the PI values of the NOHTJD algorithm with different noise level cases. The results indicate that the proposed algorithm has good robustness and accuracy even in the high noise level. Figures 3 and 4 show the values of the cost function $\mathcal{J}(\mathbf{B}^{[1]}, \dots, \mathbf{B}^{[L]})$ with different noise level cases, which also demonstrate that the NOHTJD algorithm has good robustness, and the cost function will converge to a meaningful value within several sweeps (iterations) (less than six sweeps in most cases). Remarkably, the initial condition is far from a diagonalizing solution, which means that the proposed algorithm is really robust, even if the assumption (17) does not hold in the earlier stage of iterations, that is, it is not necessary to rely on any better initialization.

Simulation 2. In this simulation, we compare the performance of the proposed algorithm with the OALS algorithm (labeled as OALS).[20] The target tensors are generated by using the same way mentioned in Simulation 1. We still, without loss of generality, consider a set of fourth-order tensors. For the OALS algorithm we stack the target tensors in a fifth-order

‡While the convergence of the proposed algorithm is quite difficult to prove theoretically, the simulation results could show its good convergence.

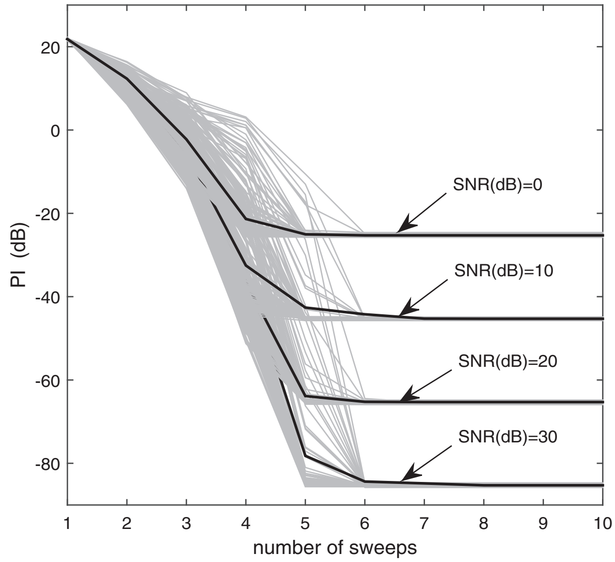


FIGURE 2 Performance index values of the nonsymmetric higher-order tensors algorithm with different noise level cases when $K = N = 20$. In gray, 100 individual realizations are plotted and in black the mean over these realizations

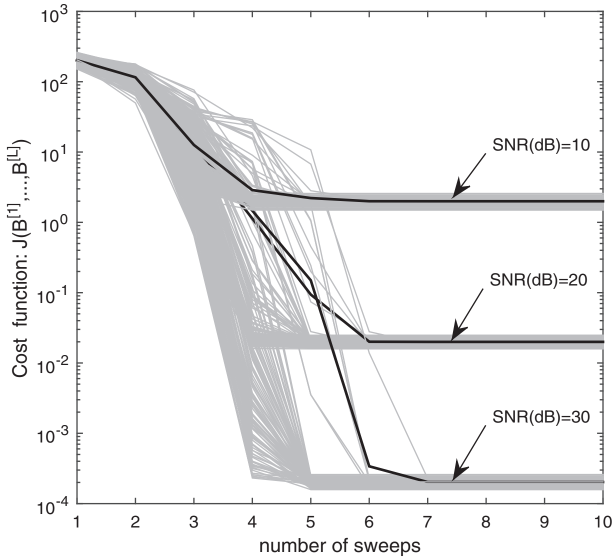


FIGURE 3 Values of the cost function $\mathcal{J}(\mathbf{B}^{[1]}, \ldots, \mathbf{B}^{[L]})$ with different noise level cases when $K = N = 10$. In gray, 100 individual realizations are plotted and in black the mean over these realizations

tensor $\mathcal{Y}$, and first identify the four estimated matrices $\tilde{\mathbf{A}}^{[1]}, \ldots, \tilde{\mathbf{A}}^{[4]}$ corresponding to each searched matrices $\mathbf{A}^{[1]}, \ldots, \mathbf{A}^{[4]}$, and we consider $\mathbf{G}^{[l]} = (\tilde{\mathbf{A}}^{[l]})^{\dagger}\mathbf{A}^{[l]}, l = 1, \ldots, 4$. The good initial matrices of $\tilde{\mathbf{A}}^{[l]}$ (labeled as $\breve{\mathbf{A}}^{[l]}$), $l = 1, \ldots, 4$ are built by the four left-hand singular vectors of $\mathbf{Y}_{(l)}, l = 1, \ldots, 4$, respectively, where $\mathbf{Y}_{(l)}$ denotes mode-l matricization of tensor $\mathcal{Y}$. A good initialization is crucial for the validity of the OALS algorithm. While the NOHTJD algorithm does not necessary to rely on any better initialization, in order to compare with the OALS algorithm, we let the initial diagonalizing matrices $\mathbf{B}^{[l]} = (\breve{\mathbf{A}}^{[l]})^{\dagger}, l = 1, \ldots, 4$.

Figures 5 and 6 show the average (100 individual realizations) PI values of the NOHTJD and the OALS algorithms with different noise level cases. The results indicate that the proposed algorithm has a much better accuracy than that of the OALS. The number of iterations of the NOHTJD seems to be also a little less than that of the OALS algorithm. The computational complexity (flops per iteration) of the proposed algorithm is comparable to that of the OALS algorithm ($\mathcal{O}(KLN^{(L+1)})$). Hence, we can say that the convergence speed[§] of the proposed algorithm has a slight edge over that of the OALS algorithm. Figures 7 and 8 further demonstrate that the NOHTJD algorithm has a better accuracy than that of

[§]We did not compare the costed CPU time which is highly correlated with the degree of optimization of the code. It may make more sense to combine the computational complexity per iteration and the number of iterations.

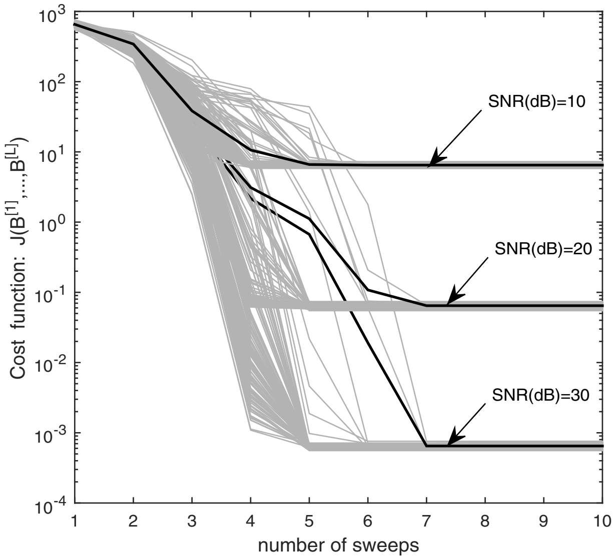


FIGURE 4 Values of the cost function $\mathcal{J}(\mathbf{B}^{[1]}, \dots, \mathbf{B}^{[L]})$ with different noise level cases when $K = N = 20$. In gray, 100 individual realizations are plotted and in black the mean over these realizations

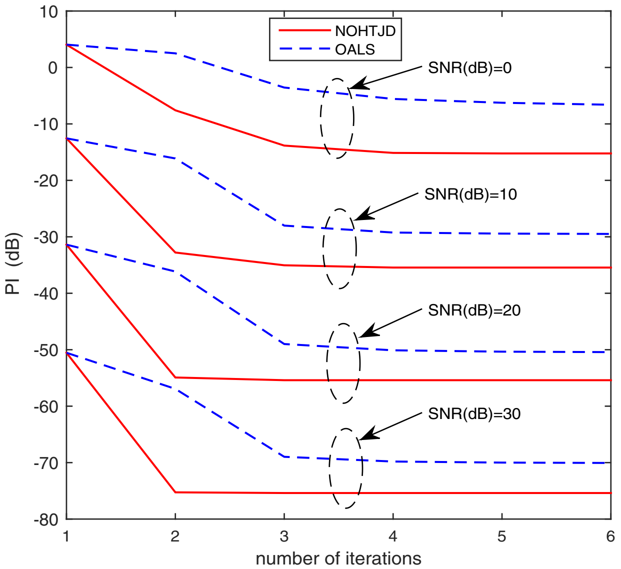


FIGURE 5 The average (100 individual realizations) Performance index values of the nonsymmetric higher-order tensors and the OALS algorithms with different noise level cases when $K = N = 5$

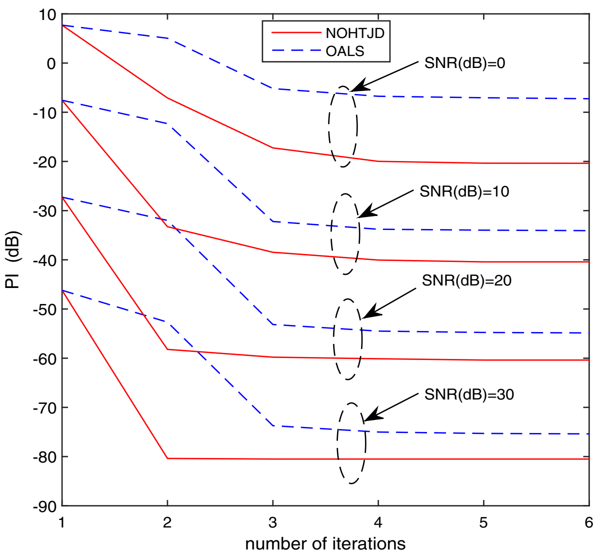


FIGURE 6 The average (100 individual realizations) performance index values of the nonsymmetric higher-order tensors and the OALS algorithms with different noise level cases when $K = N = 10$

FIGURE 7 The average (100 individual realizations) performance index values of the nonsymmetric higher-order tensors and the OALS algorithms when $K = N = 5$, SNR(dB)=0, 5, … , 40

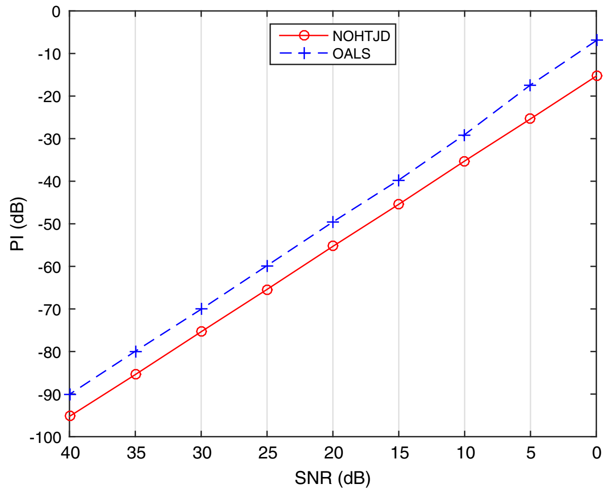
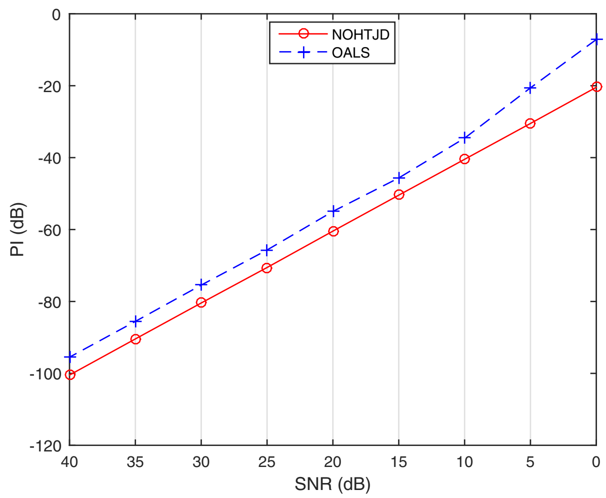


FIGURE 8 The average (100 individual realizations) performance index values of the nonsymmetric higher-order tensors and the OALS algorithms when $K = N = 10$, SNR(dB)=0, 5, … , 40



the OALS. And in the extremely higher noise level (SNR(dB)= 0), the advantage of the proposed algorithm seems to be more obvious.

Simulation 3. As we mentioned in the introduction, when only one tensor is considered (i.e., $K = 1$) in our JD model, the proposed algorithm does the same thing (orthogonal diagonalization of the single tensor) as those algorithms provided in Reference 21. Therefore, in this simulation, we aim to compare the performance of the proposed algorithm (in the special case that $K = 1$) with the ALS1[¶] algorithm provided in Reference 21. The target tensor is generated by using the similar way mentioned in the Simulation 1 but $K = 1$, and we consider third-order tensor in real number field as Reference 21 does. Let $\mathbf{G}^{[l]} = \mathbf{B}^{[l]}\mathbf{A}^{[l]}, l = 1, \dots , 3$, and let the initial diagonalizing matrices $\mathbf{B}^{[l]}, l = 1, \dots , 3$, be the identity one.

Figure 9 displays PI values with respect to the sweeps of the two algorithms for 100 independent realizations. From the results, one can see that the robustness and convergence of the NOHTJD are better than those of the ALS1. Specifically, compared with the ALS1, the NOHTJD seems to less prone to stick into local minima. Figure 10 shows the average $||\mathrm{ZTdiag}\{\mathcal{X}\}||^2$ values of 100 individual realizations with respect to the different levels of noises. The results indicate that in the high noise atmosphere (SNR(dB)≤ 10), their accuracies (diagonalization) are very close, but the accuracy of the NOHTJD is much higher than that of the ALS1 when SNR(dB)>10 which is probably more common in reality.

Simulation 4. This simulation aims to demonstrate the effectiveness of the proposed algorithm in solving the JBSS problem. In order to ensure the dependence between the sources of different datasets, we can synthesize the given sources

[¶]The several algorithms proposed in Reference 21 seem to have similar performance, so we just choose one of them to compare with our algorithm.

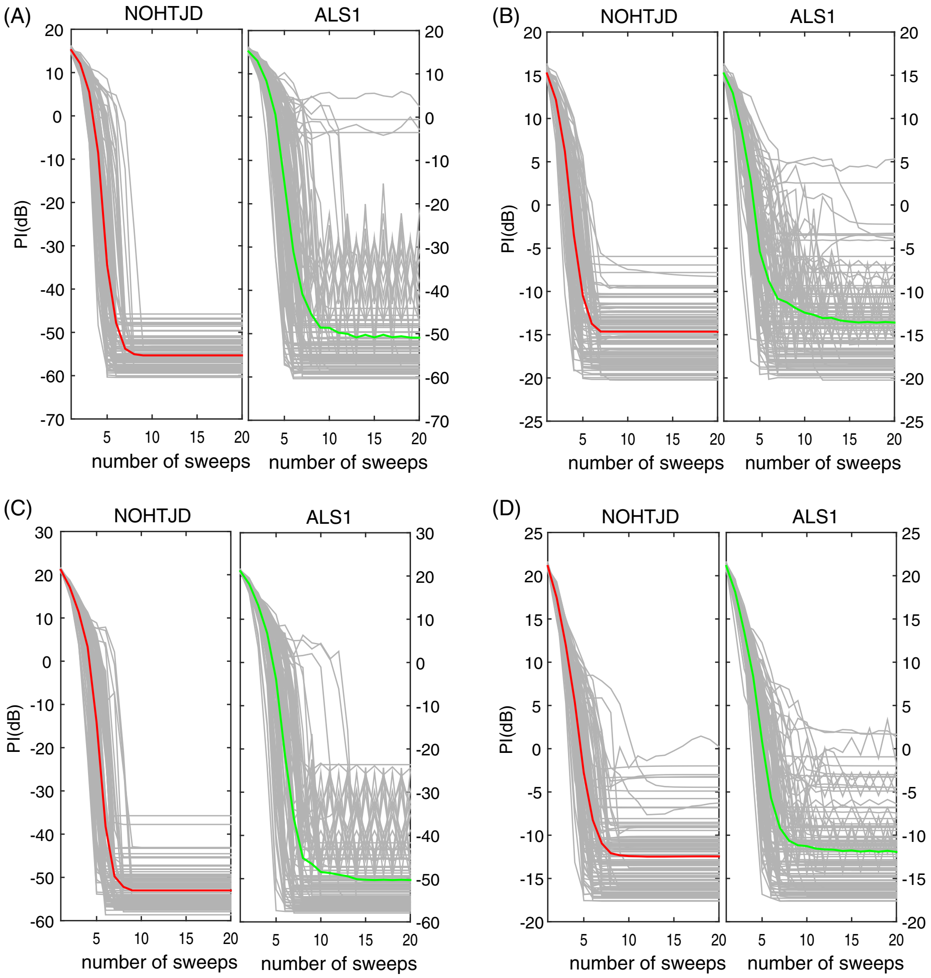


FIGURE 9 (a) $\mathcal{X} \in \mathbb{R}^{10\times10\times10}$, SNR(dB)=30. (b) $\mathcal{X} \in \mathbb{R}^{10\times10\times10}$, SNR(dB)=10. (c) $\mathcal{X} \in \mathbb{R}^{20\times20\times20}$, SNR(dB)=30. (d) $\mathcal{X} \in \mathbb{R}^{20\times20\times20}$, SNR(dB)=10. In gray, 100 individual realizations are plotted and in red/green the mean over these realizations.

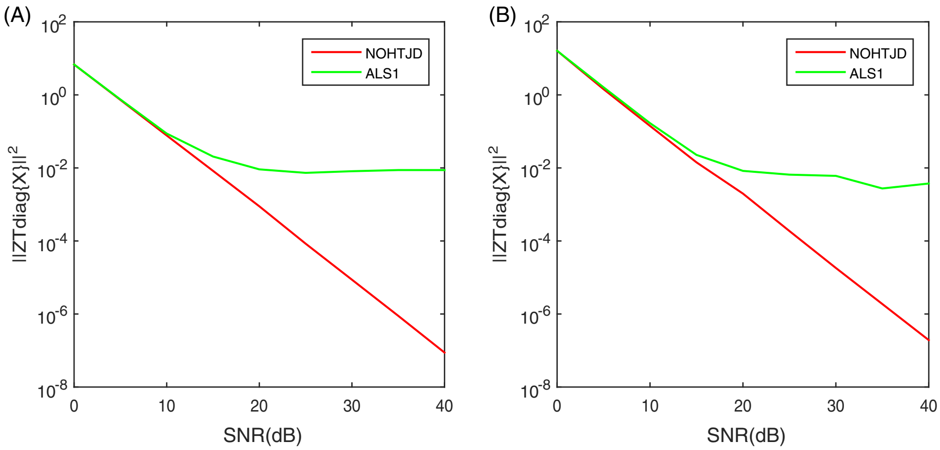


FIGURE 10 Average $\|\text{ZTdiag}\{\mathcal{X}\}\|^2$ values under scenarios (a) $\mathcal{X} \in \mathbb{R}^{10\times10\times10}$, SNR(dB)=0,10,…, 40, and (b) $\mathcal{X} \in \mathbb{R}^{20\times20\times20}$, SNR(dB)=0,10,…, 40.

as follows:[12]

$$
\begin{aligned}
\mathbf{s}^{[1]} &= \left[s_1^{[1]}, s_2^{[1]}, \ldots, s_L^{[1]}\right]^T, \\
\mathbf{s}^{[2]} &= \left[s_1^{[2]}, s_2^{[2]}, \ldots, s_L^{[2]}\right]^T, \\
&\ldots \\
\mathbf{s}^{[M]} &= \left[s_1^{[M]}, s_2^{[M]}, \ldots, s_L^{[M]}\right]^T \\
&= \mathbf{s}^{[1]}.*(unifrnd(0, 1, \mathbf{s}^{[1]})),
\end{aligned} \tag{25}
$$

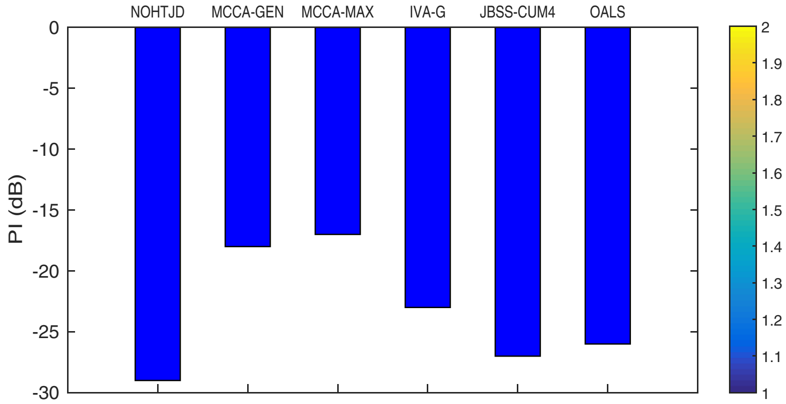


FIGURE 11 Average performance index values of 100 individual realizations, in simulation 4

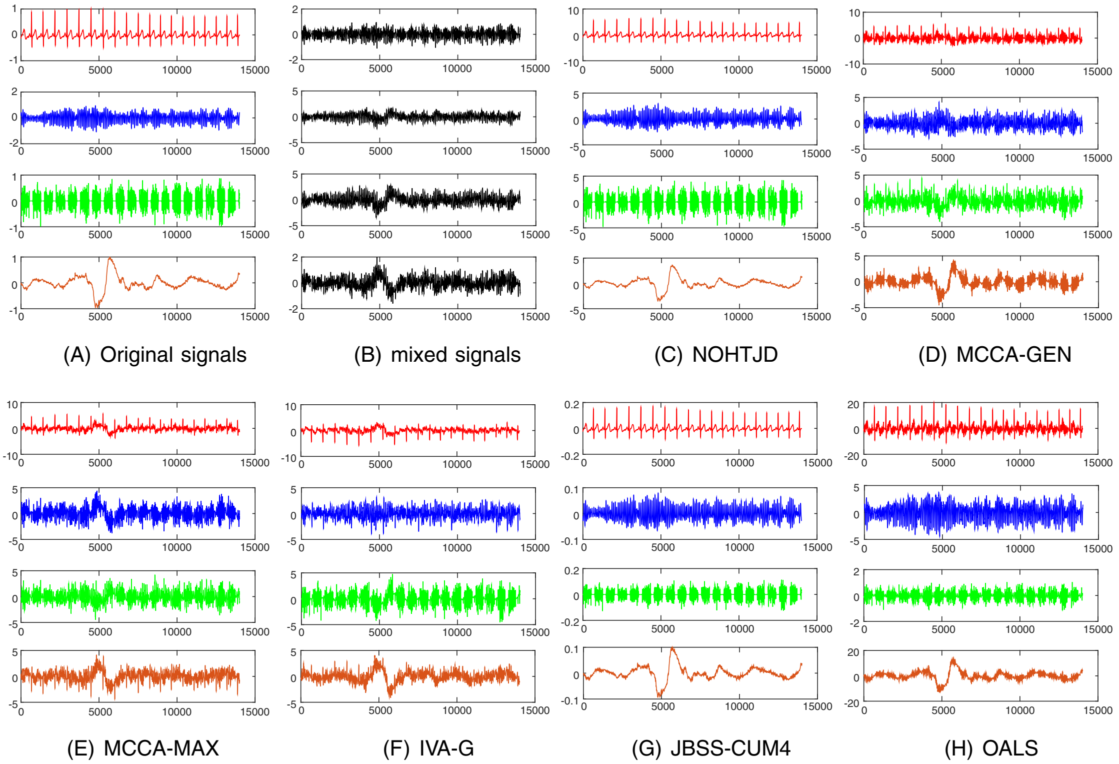


FIGURE 12 Original signals (a) and recovered signals (c) to (h) from the first dataset. Similar results are observed for the other datasets

where $\mathbf{X}. * \mathbf{Y}$ denotes the element-by-element multiplication of two vectors or matrices $\mathbf{X}$ and $\mathbf{Y}$ which must have the same dimensions, $unifrnd(0, 1, \mathbf{s}^{[1]})$ generates a vector with the same size of $\mathbf{s}^{[1]}$ and each element of the vector is randomly drawn from the continuous uniform distribution on the interval $(0, 1)$, and $s_l^{[1]}, \ldots, s_l^{[M]}$ denote the lth element of datasets $\mathbf{s}^{[1]}, \ldots, \mathbf{s}^{[M]}$, respectively. The average correlation between the corresponding sources in two different datasets is about 0.75, which can be regarded as highly correlated. Four datasets of sources are used in this simulation, which are generated following (25). Each dataset includes four physiological signals (electrocardiogram [ECG], electroencephalogram [EEG], electromyography [EMG], and electrooculography [EOG]) from a publicly available database.[31] Mixing matrices are generated randomly with elements drawn from the standard normal distribution $\mathcal{N}(0, 1)$ (the mixed signals have been pre-whited). The relation between the JBSS and the JD of higher-order tensors are shown in B1, where we show how the target tensors are generated. Specifically, we consider 10 ($K = 10$) target tensors that are generated with time-delay cross fourth-order cumulants ($L = 4$) of pre-whited observation signals (in real number field). A set of time delays, $\tau_1^{[k]} = \tau_2^{[k]} = \tau_3^{[k]} = k$, $(k = 1, 2, \ldots, 10)$, are empirically considered in our simulation.

TABLE 2 The correlation coefficients between original signals and recovered signals in simulation 4 (using the bold face type for the best correlation)

	Algorithms	$s1$	$s2$	$s3$	$s4$
Dataset 1	NOHTJD	**0.9998**	0.9966	**0.9995**	**0.9989**
	MCCA-GEN	0.8787	0.7646	0.7272	0.4602
	MCCA-MAX	0.7293	0.6319	0.7794	0.8888
	IVA-G	0.6694	0.8603	0.7984	0.7628
	JBSS-CUM4	0.9965	0.9962	0.9974	0.9986
	OALS	0.8961	**0.9967**	0.9053	0.7994
Dataset 2	NOHTJD	**0.9998**	**0.9958**	**0.9992**	**0.9989**
	MCCA-GEN	0.6526	0.8284	0.6785	0.8273
	MCCA-MAX	0.8878	0.7486	0.8088	0.6809
	IVA-G	0.7405	0.8360	0.8611	0.7331
	JBSS-CUM4	0.9996	0.9904	0.9888	0.8983
	OALS	0.9033	0.8221	0.8991	0.9506
Dataset 3	NOHTJD	**0.9997**	**0.9976**	**0.9997**	**0.9992**
	MCCA-GEN	0.7373	0.8461	0.8038	0.6913
	MCCA-MAX	0.8230	0.6649	0.8910	0.7324
	IVA-G	0.9105	0.8063	0.7765	0.7785
	JBSS-CUM4	0.9994	0.9884	0.9995	0.9992
	OALS	0.8324	0.8993	0.9538	0.8369
Dataset 4	NOHTJD	0.9989	0.9971	**0.9998**	**0.9998**
	MCCA-GEN	0.9538	0.8343	0.7919	0.8879
	MCCA-MAX	0.7990	0.8088	0.8673	0.7477
	IVA-G	0.8536	0.8122	0.6988	0.7641
	JBSS-CUM4	**0.9993**	0.9992	0.9995	0.8709
	OALS	0.9008	**0.9996**	0.8807	0.9162

We compare the NOHTJD with the MCCA (including "GENVAR" and " MAXVAR" objective functions, called the MCCA-GEN and the MCCA-MAX, respectively),[13] the IVA (using the multivariate Gaussian distribution prior, called the IVA-G),[32] the JBSS-CUM4[17] and the OALS.[20] Figure 11 shows average PI values of 100 individual realizations of each considered method. We can find that the NOHTJD provides the highest level of accuracy. Figure 12a shows the original signals of the first dataset (from top to bottom, ECG, EEG, EMG, and EOG). Figure 12b is the mixed signals. Figure 12c–h exhibit the recovered signals by the NOHTJD, the MCCA-GEN, the MCCA-MAX, the IVA-G, the JBSS-CUM4 and the OALS from the first dataset, respectively. Similar results are observed for the other datasets. Table 2 shows the correlation coefficients between original signals and recovered signals measured by the Pearson product-moment correlation coefficient (PPMCC)[33] which is defined as

$$PPMCC = \frac{\mathbb{C}ov\{s_n, \hat{s}_n\}}{\sigma_{s_n}\sigma_{\hat{s}_n}},$$

where $\hat{s}_n$ denotes the estimate of the source s_n, $\mathbb{C}ov$ denotes the covariance operator and σ denotes the SD. From the simulation results, we can see that the overall JBSS performance of the proposed algorithm has a great competitive edge (the results of Figure 12 and Table 2 are obtained by choosing the first realization from the 100 individual realizations).

5 | CONCLUSIONS

We have proposed a unitary JD algorithm for a set of nonsymmetric higher-order tensors based on Givens-like rotations. The analytical solution is obtained for each unitary rotation matrix, leading to a simple implementation. Corresponding to our model, the proposed algorithm can serve for the CPD with orthogonal factors, and it can outperform traditional methods in certain scenarios. The simulation results demonstrate that the proposed algorithm has good convergence and is not sensitive to the initial conditions, although the convergence of it is quite difficult to prove theoretically. Moreover, when the proposed algorithm is applied to deal with the JBSS problem, that is, by jointly diagonalizing a set of time-delay cross-high-order cumulants established by observed signals (pre-whited) from multiple datasets, it can adequately utilize the rich structure hidden in high-order data. Simulation results illustrate the overall good behavior of the proposed approach.

ACKNOWLEDGEMENTS
This work was supported by NSFC (11671023) and the Fundamental Research Funds for the Central Universities.

ORCID
Guanghui Cheng https://orcid.org/0000-0002-7325-8892

REFERENCES

1. Kilmer ME, Martin CD. Factorization strategies for third-order tensors. Linear Algebra Appl. 2011;435(3):641–658.
2. Batselier K, Wong N. Symmetric tensor decomposition by an iterative eigendecomposition algorithm. J Comput Appl Math. 2016;308:69–82.
3. Tichavský P, Phan AH, Cichocki A. Numerical CP decomposition of some difficult tensors. J Comput Appl Math. 2017;317:362–370.
4. Iferroudjene R, Abed-Meraim K, Belouchrani A. A new Jacobi-like method for joint diagonalization of arbitrary non-defective matrices. Appl Math Comput. 2009;211(2):363–373.
5. Shi DC, Cai YF, Xu SF. Some perturbation results for a normalized non-orthogonal joint diagonalization problem. Linear Algebra Appl. 2015;484:457–476.
6. Glashoff K, Bronstein MM. Matrix commutators: Their asymptotic metric properties and relation to approximate joint diagonalization. Linear Algebra Appl. 2013;439(8):2503–2513.
7. Miao J, Cheng G, Li W, Zhang G. Non-orthogonal approximate joint diagonalization of non-Hermitian matrices in the least-squares sense. Neurocomputing. 2019;364:63–76.
8. Comon P. Independent component analysis: A new concept? Signal Process. 1994;36(3):287–314.
9. Lathauwer LD, Moor BD, Vandewalle J. Independent component analysis and (simultaneous) third-order tensor diagonalization. IEEE Trans Signal Process. 2001;49(10):2262–2271.
10. Moreau E. A generalization of joint-diagonalization criteria for source separation. IEEE Trans Signal Process. 2001;49(3):530–541.
11. Maurandi V, Moreau E. Non-orthogonal simultaneous diagonalization of K-order complex tensors for source separation. IEEE Signal Process Lett. 2017;24(11):1621–1625.
12. Zou L, Chen X, Wang ZJ. Underdetermined joint blind source separation for two datasets based on tensor decomposition. IEEE Signal Process Lett. 2016;23(5):673–677.
13. Li Y, Adali T, Wang W, Calhoun VD. Joint blind source separation by multiset canonical correlation analysis. IEEE Trans Signal Process. 2009;57(10):3918–3929.
14. Calhoun VD, Adali T. Multisubject independent component analysis of fMRI: A decade of intrinsic networks, default mode, and neurodiagnostic discovery. IEEE Rev Biomed Eng. 2012;5:60–73.
15. Michael AM, Anderson M, Miller RL, Adali TT, Calhoun VD. Preserving subject variability in group fMRI analysis: Performance evaluation of GICA vs. IVA. Front Syst Neurosci. 2014;8(106):106.
16. Adali T, Levin-Schwartz Y, Calhoun VD. Multimodal data fusion using source separation: Two effective models based on ICA and IVA and their properties. Proc IEEE. 2015;103(9):1478–1493.
17. Li X, Adali T, Anderson M. Joint blind source separation by generalized joint diagonalization of Cumulant matrices. Signal process. 2011;91(10):2314–2322.
18. Kolda TG, Bader BW. Tensor decompositions and applications. SIAM Rev. 2009;51(3):455–500.
19. Cheng L, Wu Y, Poor HV. Probabilistic tensor canonical polyadic decomposition with orthogonal factors. IEEE Trans Signal Process. 2017;65(3):663–676.
20. Sørensen M, Lathauwer LD, Comon P, Icart S, Deneire L. Canonical polyadic decomposition with a columnwise orthonormal factor matrix. SIAM J Matrix Anal Appl. 2012;33(4):1190–1213.
21. Comon P, Sorensen M. Tensor diagonalization by orthogonal transforms. Technical report, Rapport de recherche ISRN I3S/RR-2007-06-FR, Sophia Antipolis: Universite Nice Sophia Antipolis; 2007.

22. Maurandi V, Moreau E. Fast Jacobi algorithm for non-orthogonal joint diagonalization of non-symmetric third-order tensors. Poster session presented at: 23rd European Signal Processing Conference, EUSIPCO 2015; August 31 - September 4, 2015:1311-1315; Nice, France.
23. Ozdemir A, Zare A, Iwen MA, Aviyente S. Multiscale analysis for higher-order tensors. Paper presented at: Proc. SPIE 11138, Wavelets and Sparsity XVIII; 2019:1113808. https://doi.org/10.1117/12.2525530.
24. Tichavský P, Phan AH, Cichocki A. Non-orthogonal tensor diagonalization. Signal process. 2017;138:313–320.
25. Cheng G, Li S, Moreau E. New Jacobi-like algorithms for non-orthogonal joint diagonalization of Hermitian matrices. Signal process. 2016;128:440–448.
26. Maurandi V, Moreau E, Luigi CD. Jacobi like algorithm for non-orthogonal joint diagonalization of Hermitian matrices. Poster session presented at: IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2014; May 4-9, 2014:6196-6200; Florence, Italy.
27. Trainini T, Moreau E. A coordinate descent algorithm for complex joint diagonalization under Hermitian and transpose congruences. IEEE Trans Signal Process. 2014;62(19):4974–4983.
28. Macchi O, Moreau E. Self-adaptive source separation by direct and recursive networks. Poster session presented at: International Conference on Digital Signal Processing (DSP93); 1993:1154-1159.
29. Miao J, Cheng G, Cai Y, Xia J. Approximate joint singular value decomposition algorithm based on Givens-like rotation. IEEE Signal Process Lett. 2018;25(5):620–624.
30. Chabriel G, Barrère J. A direct algorithm for non-orthogonal approximate joint diagonalization. IEEE Trans Signal Process. 2012;60(1):39–47. https://doi.org/10.1109/TSP.2011.2171682.
31. Goldberger AL, Amaral LAN, Glass L, et al. Physiobank, physiotoolkit, and physionet components of a new research resource for complex physiologic signals. Circulation. 2000;101(23):E215.
32. Anderson M, Li X, Adali T. Complex-valued independent vector analysis: Application to multivariate Gaussian model. Signal process. 2012;92(8):1821–1831.
33. Benesty J, Chen J, Huang Y, Cohen I. Pearson correlation coefficient. Berlin, Germany / Heidelberg: Springer, 2009.
34. Lathauwer LD, Moor BD, Vandewalle J. A multilinear singular value decomposition. SIAM J Matrix Anal Appl. 2000;21(4):1253–1278.
35. Haykin S. Independent component analysis. Dordrecht, Netherlands: Kluwer Academic Publishers, 1998.

How to cite this article: Miao J, Cheng G, Li W, Moreau E. A unitary joint diagonalization algorithm for nonsymmetric higher-order tensors based on Givens-like rotations. *Numer Linear Algebra Appl.* 2020;e2291. https://doi.org/10.1002/nla.2291

APPENDIX A. PROOF OF RELATIONS (5) AND (6)

We first give a proof of the relation in (5). Given a tensor $\mathcal{X}$, we recall that $\|\mathcal{X}\|^2 = \|\mathbf{X}_{(l)}\|^2$ for any mode-l unfolding of $\mathcal{X}$, and that the Frobenius norm for matrices is unitarily invariant. We have

$$\begin{aligned}\|\mathcal{X}\times_1\tilde{\mathbf{A}}^{[1]}\times_2\tilde{\mathbf{A}}^{[2]}\dots\times_L\tilde{\mathbf{A}}^{[L]}\|^2 &= \|\tilde{\mathbf{A}}^{[l]}\mathbf{X}_{(l)}(\tilde{\mathbf{A}}^{[L]}\otimes\dots\otimes\tilde{\mathbf{A}}^{[l+1]}\otimes\tilde{\mathbf{A}}^{[l-1]}\otimes\dots\otimes\tilde{\mathbf{A}}^{[1]})^H\|^2\\ &= \|\mathbf{X}_{(l)}(\tilde{\mathbf{A}}^{[L]}\otimes\dots\otimes\tilde{\mathbf{A}}^{[l+1]}\otimes\tilde{\mathbf{A}}^{[l-1]}\otimes\dots\otimes\tilde{\mathbf{A}}^{[1]})^H\|^2\\ &= \|\mathbf{X}_{(l)}\mathbf{W}^H\|^2\\ &= \|\mathbf{X}_{(l)}\|^2\\ &= \|\mathcal{X}\|^2,\end{aligned}\tag{A1}$$

where $\mathbf{W} = \tilde{\mathbf{A}}^{[L]}\otimes\dots\otimes\tilde{\mathbf{A}}^{[l+1]}\otimes\tilde{\mathbf{A}}^{[l-1]}\otimes\dots\otimes\tilde{\mathbf{A}}^{[1]}$, and, we see that

$$\begin{aligned}\mathbf{W}\mathbf{W}^H &= \mathbf{W}^H\mathbf{W}\\ &= (\tilde{\mathbf{A}}^{[L]}\otimes\dots\otimes\tilde{\mathbf{A}}^{[l+1]}\otimes\tilde{\mathbf{A}}^{[l-1]}\otimes\dots\otimes\tilde{\mathbf{A}}^{[1]})^H(\tilde{\mathbf{A}}^{[L]}\otimes\dots\otimes\tilde{\mathbf{A}}^{[l+1]}\otimes\tilde{\mathbf{A}}^{[l-1]}\otimes\dots\otimes\tilde{\mathbf{A}}^{[1]})\\ &= ((\tilde{\mathbf{A}}^{[L]})^H\tilde{\mathbf{A}}^{[L]})\otimes\dots\otimes((\tilde{\mathbf{A}}^{[l+1]})^H\tilde{\mathbf{A}}^{[l+1]})\otimes((\tilde{\mathbf{A}}^{[l-1]})^H\tilde{\mathbf{A}}^{[l-1]})\otimes\dots\otimes((\tilde{\mathbf{A}}^{[1]})^H\tilde{\mathbf{A}}^{[1]})\\ &= \mathbf{I},\end{aligned}$$

That is, $\mathbf{W}$ is a unitary matrix. Hence, clearly, (A1) is hold.

We now give a proof of the second relation in (6). We, firstly, recall that the n-mode product satisfies the following two properties, see De Lathauwer et al.[34] and Tamara G. Kolda and Brett W. Bader.[18]

Property 1. Given the tensor $\mathcal{X}$ and the matrices $\mathbf{F}$ and $\mathbf{G}$, for distinct modes in a series of multiplications, the order of the multiplication is irrelevant, that is,

$$\mathcal{X}\times_m \mathbf{F}\times_n \mathbf{G} = \mathcal{X}\times_n \mathbf{G}\times_m \mathbf{F} \quad (m \neq n).$$

Property 2. Given the tensor $\mathcal{X}$ and the matrices $\mathbf{F}$ and $\mathbf{G}$, for the same modes in a series of multiplications, one has

$$\mathcal{X}\times_m \mathbf{F}\times_m \mathbf{G} = \mathcal{X}\times_m (\mathbf{GF}).$$

Then, based on the Property 1 and Property 2, we easily have

$$\begin{aligned}
&(\mathcal{X}\times_1 \tilde{\mathbf{A}}^{[1]}\times_2 \tilde{\mathbf{A}}^{[2]} \ldots \times_L \tilde{\mathbf{A}}^{[L]})\times_1 (\tilde{\mathbf{A}}^{[1]})^H \times_2 (\tilde{\mathbf{A}}^{[2]})^H \ldots \times_L (\tilde{\mathbf{A}}^{[L]})^H \\
&= \mathcal{X}\times_1 (\tilde{\mathbf{A}}^{[1]}(\tilde{\mathbf{A}}^{[1]})^H)\times_2 (\tilde{\mathbf{A}}^{[2]}(\tilde{\mathbf{A}}^{[2]})^H) \ldots \times_L (\tilde{\mathbf{A}}^{[L]}(\tilde{\mathbf{A}}^{[L]})^H) \\
&= \mathcal{X}\times_1 \mathbf{I}\times_2 \mathbf{I} \ldots \times_L \mathbf{I} \\
&= \mathcal{X}.
\end{aligned}$$

APPENDIX B. THE RELATION BETWEEN THE JBSS AND THE JD OF HIGHER-ORDER TENSORS

Let us consider a classical linear instantaneous mixing model described by $\mathbf{x}^{[l]}(t) = \mathbf{A}^{[l]}\mathbf{s}^{[l]}(t), l \in [\![L]\!]$, where t is the time index, and $\mathbf{x}^{[l]}(t) \in \mathbb{C}^N$, $\mathbf{s}^{[l]}(t) \in \mathbb{C}^N$, and $\mathbf{A}^{[l]} \in \mathbb{C}^{N\times N}$ denote the observation vector, the source vector and the mixing matrix in the lth dataset, respectively. Then, the time-delay cross L-order cumulants of observation vectors (pre-whited[#]) with the assumptions that the intra-set independence and interset dependence at the source level can be given by

$$\begin{aligned}
\mathcal{T}^{[k]} &= Cum\left[\mathbf{x}^{[1]}(t), (\mathbf{x}^{[2]})^*(t-\tau_1^{[k]}), \ldots, (\mathbf{x}^{[L]})^{b_L}(t-\tau_{(L-1)}^{[k]})\right] \\
&= Cum\left[\mathbf{s}^{[1]}(t), (\mathbf{s}^{[2]})^*(t-\tau_1^{[k]}), \ldots, (\mathbf{s}^{[L]})^{b_L}(t-\tau_{(L-1)}^{[k]})\right] \\
&\times_1 \mathbf{A}^{[1]}\times_2 (\mathbf{A}^{[2]})^* \ldots \times_L (\mathbf{A}^{[L]})^{b_L} \\
&= \mathcal{D}^{[k]}\times_1 \mathbf{A}^{[1]}\times_2 (\mathbf{A}^{[2]})^* \ldots \times_L (\mathbf{A}^{[L]})^{b_L}, \; k \in [\![K]\!],
\end{aligned} \tag{B1}$$

where $Cum[\cdot]$ denotes the cumulant operator, $\tau_1^{[k]}, \ldots, \tau_{L-1}^{[k]}$ for all $k \in [\![K]\!]$ represent the time-delay, and $b_l = 1$ for odd index l (no complex conjugate) and $b_l = *$ for even index l (complex conjugate). $\mathcal{T}^{[k]} \in \mathbb{C}^{N\times N\times \ldots \times N}$ and $\mathcal{D}^{[k]} \in \mathbb{C}^{N\times N\times \ldots \times N}$ for all $k \in [\![K]\!]$ are L-order target tensors and diagonal tensors, respectively. Hence, the problem of the JBSS for L datasets can be transformed into the JD problem of L-order nonsymmetric tensors.

[#] For simplicity, we still use $\mathbf{A}^{[l]}$ to represent $\mathbf{W}^{[l]}\mathbf{A}^{[l]}$, $l \in [\![L]\!]$, where $\mathbf{W}^{[l]}$ denotes the whitening matrix.[35]