Generalized FFDIAG algorithm for non-Hermitian joint matrix diagonalization

Nacerredine Lassami, Ammar Mesloub, Abdeldjalil Aissa El Bey, Karim Abed-Meraim, Adel Belouchrani

Generalized FFDIAG Algorithm for Non-Hermitian Joint Matrix Diagonalization

Nacerredine Lassami[a], Ammar Mesloub[a], Abdeldjalil Aïssa-El-Bey[b], Karim Abed-Meraim[c], Adel Belouchrani[d]

[a] *Laboratoire traitement du signal, Ecole Militaire Polytechnique, Bordj El Bahri, Algiers, Algeria*
[b] *IMT Atlantique, Lab-STICC, UMR CNRS 6285, F-29238, France.*
[c] *University of Orleans, 12 Rue de Blois, 45067 Orleans, France.*
[d] *Ecole Nationale Polytechnique, LDCCP, Algiers, Algeria.*

Abstract

This paper deals with the joint diagonalization problem of a set of non-Hermitian matrices. It generalizes the first-order approximation used in the Fast Frobenius Diagonalization algorithm for Hermitian matrices to handle the non-Hermitian case. In the case of non-unitary diagonalization, the proposed Generalized (G)FFDiag algorithm outperforms the state-of-the-art algorithms and ensures fast convergence while reducing computational complexity. In the case of unitary diagonalization, we propose the Unitary (U)GFFDiag algorithm that follows the same framework as the GFFDiag algorithm and includes a final step that ensures the unitary constraint. Two variants are proposed depending on whether the unitary constraint is exact or approximated. The efficacy of GFFDiag and UGFFDiag is demonstrated by numerical experiments, including validation on real-world data, which indicate that these algorithms outperform existing algorithms in terms of computational efficiency and accuracy, particularly when dealing with a small number of large size matrices. The flexibility of these algorithms is highlighted by their applicability to both unitary and non-unitary diagonalization, as well as to rectangular and square matrices, thereby

Email addresses: lassami.nacerredine@gmail.com (Nacerredine Lassami), mesloub.a@gmail.com (Ammar Mesloub), abdeldjalil.aissaelbey@imt-atlantique.fr (Abdeldjalil Aïssa-El-Bey), karim.abed-meraim@univ-orleans.fr (Karim Abed-Meraim), adel.belouchrani@g.enp.edu.dz (Adel Belouchrani)

expanding their relevance across a wide range of applications.

Keywords: Joint matrix diagonalization, Blind source separation, Canonical Polyadic Decomposition, non-Hermitian, Higher-order singular value decomposition.

---

1. Introduction

Joint matrix Diagonalization (JD) and its generalization into tensor decomposition are well-established data analysis and signal processing techniques. For instance, those techniques are used to solve problems such as Independent Component Analysis [1][2], Blind Source Separation (BSS) [3][4][5][6][7], image denoising [8], radar tracking [9][10] and channel estimation in Multiple-Input and Multiple-Output (MIMO) telecommunications systems [11][12][13].

Joint Diagonalization (JD) usually refers generally to one of two distinct approaches: joint diagonalization by congruence (wherein the left diagonalization matrix is the transpose of the right one) and joint diagonalization by similarity (wherein the left diagonalization matrix is the inverse of the right one). The latter is also known as joint eigenvalue decomposition. This paper addresses a more general JD problem, which aims to identify a common pair of diagonalization matrices able to achieve joint diagonalization of a given set of matrices. It should be noted that the left and right diagonalization matrices are not necessarily related (for example, through an inverse, pseudo-inverse, or transpose), and they may be rectangular. Therefore, JD by congruence and JD by similarity represent special cases of the considered generalized problem.

Based on the JD of fourth-order and second-order moments, respectively, JADE [1] and SOBI [14] are among the prominent JD algorithms addressing the BSS problem. Other JD algorithms, such as in [3][15][6][16][17], were later proposed. Two classes of algorithms can be distinguished: orthogonal JD algorithms [1][14][5] and non-orthogonal JD algorithms [3][15][6][16][17][18]. The first class constrains the diagonalizing matrix to be orthogonal and typically requires pre-whitening the data to perform BSS. In contrast, non-orthogonal

JD algorithms are more general, requiring only that the diagonalizing matrix to be non-singular.

Non-Hermitian (or Non-Symmetrical in the case of real data) Joint matrix Diagonalization (NHJD) has attracted many researchers' attention lately [19][20][21] [22][23][24][25]. The NHJD aims to find a diagonalization pair of matrices, equivalent to left and right Singular Value Decomposition (SVD) matrices (not necessarily orthogonal), that can diagonalize a set of matrices.

The NHJD formulation is more general and interesting compared to the Hermitian case. For instance, in conventional instantaneous BSS methods, the target matrices are typically intra-set statistics of a dataset, such as time-shifted covariance matrices with a symmetric or Hermitian inherent structure. However, the growing availability of multi-set signals has presented new challenges that address the inter-set statistics. For instance, in the convolutive BSS context, the intra-set independence is considered for each frequency bin, and the inter-set dependence is related to the cross-frequency dependencies [26].

Exploiting these inter-dependencies between datasets increases the separation efficiency in comparison to the separation achievable with each dataset separately. In addition, it reduces the permutation ambiguity to a unique common permutation matrix, i.e. it aligns the separated sources across datasets[5]. Those assumptions are also widely used in multi-set biomedical data fusion (fMRI, EEG, MEG) [27].

Authors in [20] proposed an approach to the orthogonal NHJD problem that alternates between two principal eigenvector problems. These are solved using either the Jacobi iterations technique or the power method combined with Lödwin symmetric orthonormalization. The same alternating technique was adopted in [21] in the non-orthogonal case and applied to operational modal analysis. Subsequently, Jacobi-like algorithms were proposed in [22] and [23] for both the orthogonal and non-orthogonal cases. Both algorithms minimize the off-diagonal criterion with a reduced computational complexity. In their study [24], the authors proposed a solution based on the minimization of the least-squares criterion, which proved to be more robust to noise than previous

methods.

The NHJD problem can be expressed as a third-order tensor decomposition known as Canonical Polyadic Decomposition (CPD)[28]. This can be seen naturally as the generalization of matrix JD while dealing with multidimensional data that can be structured in tensors of higher order. The connection between Tensor diagonalization and Independent Component Analysis (ICA) was initially studied in [29] where the solution proposed was based on a Jacobi-like iteration algorithm. Later, in [30] and [31], the link between tensor CPD and simultaneous JD of matrices was studied and this formulation was applied for BSS. The formulation of Joint Eigenvalue Decomposition (JD by similarity), has also been linked to the CPD, as discussed in [32][33], using a semi-algebraic framework. This framework was extended in [34] and [35], for non-orthogonal simultaneous diagonalization of any order complex tensors and applied to digital telecommunications source separation.

Authors in [25] considered the unitary joint diagonalization case for a set of nonsymmetric higher-order tensors (three or higher). This is also known in the literature as the Higher-Order version of Singular Value Decomposition (HOSVD) for third-order tensors.

Paper contributions

This paper proposes a fast algorithm based on the off-diagonal criterion for joint diagonalization of a set of non-Hermitian matrices. First, the non-unitary case is considered and a first-order approximation as the one proposed in the FFDiag algorithm [15] for the symmetrical case is developed. The resulting solution requires inversion and multiplication of large matrices. Furthermore, we expand the calculation and we propose a fast version algorithm GFFDiag where the overall complexity is reduced. The proposed algorithm GFFDiag, compared to the state-of-the-art, reaches approximately the same performance while having a reduced complexity. The key advantage in terms of performance, can be seen in the case of fewer matrices to diagonalize compared to the matrices' dimensions. Initially developed for the square case, the proposed GFFDiag

algorithm can also be extended to handle the rectangular case, enhancing its versatility. This flexibility allows it to serve as a general Canonical Polyadic Decomposition (CPD) algorithm for third-order tensors, extending its applicability to a wider range of problems. In the case of unitary diagonalization, we propose the UGFFDiag algorithm, which has the same framework as the GFF-Diag algorithm, followed by a final step that ensures the unitary constraint. Two variants are proposed depending on whether the unitary constraint is exact or approximated. Finally, the proposed algorithms were validated through numerical simulations using both synthetic data and real-world examples.

Paper notations

The notations x, $\mathbf{x}$, $\mathbf{X}$, and $\mathcal{X}$ denote a scalar, a vector, a matrix, and a tensor, respectively. For a matrix $\mathbf{A} \in \mathbb{C}^{d\times p}$, $\mathbf{A}_{ij}$ denotes its ij-th element. The notations $(.)^*$, $(.)^H$ and $(.)^T$ are the complex conjugate operator, the complex conjugate transpose, and the transpose operator, respectively. $\mathbf{I_p}$ is the $p \times p$ identity matrix. We define the Frobenius norm of $\mathbf{A}$ by $\|\mathbf{A}\|_F \triangleq \sqrt{\sum_{i=1}^{d}\sum_{j=1}^{p}|\mathbf{A}_{ij}|^2} = \sqrt{\mathrm{Tr}(\mathbf{A}\mathbf{A}^{\mathrm{H}})}$ where Tr(.) is the trace function. Assuming $p < d$, the Moore-Penrose pseudo inverse is designated by $\mathbf{A}^\dagger = (\mathbf{A}^H\mathbf{A})^{-1}\mathbf{A}^H$ and $(.)^{-1}$ is inverse operator of a square matrix. The multiplication $\otimes_k$ denotes the tensor-matrix multiplication along the dimension k, known also as the k-mode product. The function off(.) is the off-diagonal operator that sets all diagonal elements of the argument to zero. The operator $\mathbb{E}[.]$ denotes the statistical expectation.

2. Data model and problem formulation

Let us consider a set of of $N_1 \times N_2$ complex non-Hermitian matrices $\mathbf{C}_k$ that admit the same common decomposition:

$$\mathbf{C}_k = \mathbf{A}_1\mathbf{D}_k\mathbf{A}_2^H + \mathbf{E}_k \quad \text{for} \quad k = 1, ..., K \tag{1}$$

where $\mathbf{D}_k \in \mathbb{C}^{N\times N}$ are diagonal matrices for all $k = 1, ..., K$. $\mathbf{A}_1 \in \mathbb{C}^{N_1\times N}$ and $\mathbf{A}_2 \in \mathbb{C}^{N_2\times N}$ are two full column rank mixing matrices. The noise matrices $\mathbf{E}_k \in$

$\mathbb{C}^{N\times N}$ for $k = 1, ..., K$ express different types of modelling and/or estimation errors. We consider, without loss of generality, the square case where $N_1 = N_2 = N$. The rectangular case will be discussed at the end of 3.1 subsection.

This model is used in many signal processing applications. For instance, the Joint BSS [27] problem exploits the dependence between two data sets by considering the time-delay cross-covariance matrices which would follow the model in (1) under mild assumptions. Furthermore, if we consider $\mathbf{A}_1 = \mathbf{A}_2$, the model in (1) is reduced to the hermitian case which was studied intensively, especially for second-order or fourth-order statistics.

The aim of NHJD is to find the complex $N \times N$ full column rank matrices $\mathbf{B}_1$ and $\mathbf{B}_2$, called separation (diagoalization) matrices such that the matrices $\hat{\mathbf{D}}_k = \mathbf{B}_1\mathbf{C}_k\mathbf{B}_2^H$ are as diagonal as possible for all $k = 1, ..., K$.

In the noiseless case, the considered problem is also equivalent to the Canonical Polyadic Decomposition (CPD) of the third-order tensor

$$\mathcal{C} = \mathcal{D} \otimes_1 \mathbf{A}_1 \otimes_2 \mathbf{A}_2^* \tag{2}$$

where $\mathcal{D} \in \mathbb{C}^{N\times N\times K}$ is the third-order tensor made from stacking the diagonal matrices $\mathbf{D}_k$ for $k = 1, ..., K$ and $\otimes_i$ refers to the i^{th} mode tensor product. Hence, our proposed algorithm can be used to solve the problem of the CPD in the case of third-order tensors.

To resolve the above problem, many objective criteria were extended from those first proposed to address the Hermitian case [6]. In this work, we have considered minimizing the off-diagonal elements of the reconstructed diagonal matrices $\hat{\mathbf{D}}_k$:

$$\min_{\mathbf{B}_1,\mathbf{B}_2} \sum_{k=1}^{K} \|\text{off}(\hat{\mathbf{D}}_k)\|_F^2 = \min_{\mathbf{B}_1,\mathbf{B}_2} \sum_{k=1}^{K} \|\text{off}(\mathbf{B}_1\mathbf{C}_k\mathbf{B}_2^H)\|_F^2 \tag{3}$$

The solution to this problem is unique, under some mild assumptions, up to a scaling and permutation indeterminacy. Actually, if $\{\hat{\mathbf{B}}_1, \hat{\mathbf{B}}_2, \hat{\mathbf{D}}_k \quad \text{for} \quad k = 1, ..., K\}$ is a solution, then for any permutation matrix $\mathbf{P} \in \mathbb{R}^{N\times N}$ and any scaling diagonal matrices $\mathbf{\Delta}_1$ and $\mathbf{\Delta}_2 \in \mathbb{C}^{N\times N}$, the set $\{\hat{\mathbf{B}}_1\mathbf{\Delta}_1\mathbf{P}, \hat{\mathbf{B}}_2\mathbf{\Delta}_2\mathbf{P}, \mathbf{P}^T\mathbf{\Delta}_1^{-1}\hat{\mathbf{D}}_k\mathbf{\Delta}_2^{-H}\mathbf{P} \quad \text{for} \quad k = 1, ..., K\}$ is also a solution.

In some applications such as data analysis and dimension reduction, the joint diagonalization problem is restricted by the unitary constraint. In that case, the separations matrices $\mathbf{B}_1$ and $\mathbf{B}_2$ must be unitary. This is a generalization of the SVD from matrices into third-order tensors. Hence, the optimization problem is given by

$$\min_{\mathbf{B}_1,\mathbf{B}_2} \sum_{k=1}^{K} \|\text{off}(\mathbf{B}_1\mathbf{C}_k\mathbf{B}_2^H)\|_F^2 \quad \text{subject to} \quad \begin{cases} \mathbf{B}_1^H\mathbf{B}_1 = \mathbf{I}_N \\ \mathbf{B}_2^H\mathbf{B}_2 = \mathbf{I}_N \end{cases} \tag{4}$$

FFDiag algorithm

The FFDiag algorithm introduced in [3], and later in [15] solves the JD problem of symmetrical (Hermitian) matrices. It has one of the least computational complexity compared to other algorithms while providing a comparable performance in terms of estimation errors [15] [36]. The considered criterion is formulated as in (3) with $\mathbf{B}_1 = \mathbf{B_2} = \mathbf{B}$, and an iterative minimization procedure is developed based on a multiplicative update given at the i^{th} iteration by

$$\mathbf{B}^{i+1} = (\mathbf{I}_N + \mathbf{Z}^i)\mathbf{B}^i \tag{5}$$

where the $N \times N$ complex matrix $\mathbf{Z}^i$ has a null diagonal. The full column rank constraint on the diagonalization matrix $\mathbf{B}$ is ensured at each iteration based on the Levi-Desplanques theorem, i.e. $\mathbf{B}$ is forced to be strictly diagonally dominant [15]. The computational efficiency of the algorithm is due to the first-order approximation of the cost function which results in the closed-form solution. A detailed discussion of the generalization of this method for the non-hermitian case will be in the next section.

3. Proposed method

The proposed method is a generalization of the FFDiag [15] algorithm from the Hermitian case where $\mathbf{A}_1 = \mathbf{A}_2$ to the non-Hermitian case where $\mathbf{A}_1 \neq \mathbf{A}_2$.

3.1. Non-unitary case

The main proposed algorithm is developed for non-unitary joint diagonalization of non-Hermitian matrices. We propose to solve Eq. (3) by updating the matrices $\mathbf{B}_1$ and $\mathbf{B}_2$ iteratively in same way as in (5) such that:

$$\mathbf{B}_m^{i+1} = (\mathbf{I}_N + \mathbf{Z}_m^i)\mathbf{B}_m^i \quad \text{for} \quad m = 1, 2 \tag{6}$$

where both matrices $\mathbf{Z}_1^i \in \mathbb{C}^{N\times N}$ and $\mathbf{Z}_2^i \in \mathbb{C}^{N\times N}$ have a null diagonal. The superscript i is used as an indicator for the i^{th} iteration number.

The stability of such a method is related to the invertibility of the matrices $\mathbf{B}_1$ and $\mathbf{B}_2$ at each iteration i. The invertibility of $\mathbf{B}_1$ and $\mathbf{B}_2$ translates the invertibility of the mixing matrices $\mathbf{A}_1$ and $\mathbf{A}_2$ (i.e. ideally $\mathbf{B}_1 = \mathbf{A}_1^{-1}$ and $\mathbf{B}_2 = \mathbf{A}_2^{-1}$). When these matrices are singular or ill-conditioned, this inverse problem becomes difficult, e.g. [37][38], and the JD algorithms may suffer from poor convergence or instability.

Considering that $\mathbf{B}_1^i$ and $\mathbf{B}_2^i$ are full-rank matrices, then, at the $(i+1)^{\text{th}}$ iteration, one has to ensure that $\mathbf{I}_N + \mathbf{Z}_1^i$ and $\mathbf{I}_N + \mathbf{Z}_2^i$ are also invertible. The latter can be carried out by forcing these matrices to be strictly diagonally dominant according to the Levi-Desplanques theorem. In our case, this is equivalent to

$$\sum_{j\neq l} |(\mathbf{Z}_m^i)_{lj}| < 1 \qquad \text{for each row} \quad 1 \leq l \leq N \quad \text{for} \quad m = 1, 2 \tag{7}$$

This will be handled using a normalization step such that:

$$\text{If} \quad ||\mathbf{Z}_m^i||_F > 1 \quad \text{then} \quad \mathbf{Z}_m^i = \mathbf{Z}_m^i / ||\mathbf{Z}_m^i||_F \quad \text{for} \quad m = 1, 2 \tag{8}$$

One can use different scalar parameters instead of $||\mathbf{Z}_m^i||_F$, for instance, choosing optimal values that minimize the criterion in Eq. (3) under the constraint that the strictly diagonally dominant condition remains verified. Such a choice can reduce the number of iterations required until convergence. However, it will increase the computational complexity of the algorithm.

Using the normalization described in Eq. (8), the updating method will reduce the optimization problem at the i^{th} iteration into

$$\min_{\mathbf{Z}_1^i, \mathbf{Z}_2^i} \sum_{k=1}^{K} \|\text{off}\big((\mathbf{I}_N + \mathbf{Z}_1^i)\mathbf{B}_1^i \mathbf{C}_k \mathbf{B}_2^{iH} (\mathbf{I}_N + \mathbf{Z}_2^i)^H\big)\|_F^2 \tag{9}$$

We define $\mathbf{F}_k^{i+1}$ as

$$\mathbf{F}_k^{i+1} = (\mathbf{I}_N + \mathbf{Z}_1^i)\mathbf{B}_1^i \mathbf{C}_k \mathbf{B}_2^{iH} (\mathbf{I}_N + \mathbf{Z}_2^i)^H \tag{10}$$

such that our objective function of Eq. (9) can be expressed as:

$$\min_{\mathbf{Z}_1, \mathbf{Z}_2} \sum_{k=1}^{K} \|\text{off}\big(\mathbf{F}_k^{i+1}\big)\|_F^2 \tag{11}$$

Hence, one has

$$\mathbf{F}_k^i = \mathbf{B}_1^i \mathbf{C}_k \mathbf{B}_2^{iH} \tag{12}$$

and we can write that

$$\mathbf{F}_k^{i+1} = (\mathbf{I}_N + \mathbf{Z}_1^i)\mathbf{F}_k^i (\mathbf{I}_N + \mathbf{Z}_2^i)^H \tag{13}$$

we decompose the $\mathbf{F}_k^i$ in such way that

$$\mathbf{F}_k^i = \mathbf{D}_k^i + \mathbf{E}_k^i \tag{14}$$

where $\mathbf{D}_k^i$ is a $N \times N$ complex diagonal matrix and $\mathbf{E}_k^i$ is the $N \times N$ complex matrix with a null diagonal. Hence, we expand Eq. (13), into

$$\begin{aligned} \mathbf{F}_k^{i+1} =& (\mathbf{I} + \mathbf{Z}_1^i)(\mathbf{D}_k^i + \mathbf{E}_k^i)(\mathbf{I} + \mathbf{Z}_2^i)^H & (15) \\ =& \mathbf{D}_k^i + \mathbf{Z}_1^i \mathbf{D}_k^i + \mathbf{D}_k^i {\mathbf{Z}_2}^{iH} + \mathbf{E}_k^i + \mathbf{Z}_1^i \mathbf{E}_k^i \\ &+ \mathbf{E}_k^i {\mathbf{Z}_2}^{iH} + \mathbf{Z}_1^i \mathbf{D}_k^i {\mathbf{Z}_2}^{iH} + \mathbf{Z}_1^i \mathbf{E}_k^i {\mathbf{Z}_2}^{iH} & (16) \end{aligned}$$

Let us suppose that we are close to converging into a diagonalization solution, that means, the matrices $\mathbf{E}_k^i, \mathbf{Z}_1^i$ and $\mathbf{Z}_2^i$ have small values. This will allow us to use a first-order approximation of the above equation by neglecting the terms: $\mathbf{Z}_1^i \mathbf{E}_k^i$, $\mathbf{E}_k^i {\mathbf{Z}_2}^{iH}$, $\mathbf{Z}_1^i \mathbf{D}_k^i {\mathbf{Z}_2}^{iH}$ and $\mathbf{Z}_1^i \mathbf{E}_k^i {\mathbf{Z}_2}^{iH}$.

$$\mathbf{F}_k^{i+1} \approx \mathbf{D}_k^i + {\mathbf{Z}^i}_1 \mathbf{D}_k^i + \mathbf{D}_k^i {\mathbf{Z}_2}^{iH} + \mathbf{E}_k^i \tag{17}$$

Thus, this leaves us with an approximated optimization problem

$$\min_{\mathbf{Z}_1^i, \mathbf{Z}_2^i} \sum_{k=1}^{K} \|\mathrm{off}\big(\mathbf{D}_k^i + \mathbf{Z}_1^i \mathbf{D}_k^i + \mathbf{D}_k^i {\mathbf{Z}_2}^{iH} + \mathbf{E}_k^i\big)\|_F^2 \tag{18}$$

Since all the matrices $\mathbf{D}_k^i$ are diagonal, we can rewrite the optimization problem as:

$$\min_{\mathbf{Z}_1^i, \mathbf{Z}_2^i} \sum_{k=1}^{K} \sum_{l \neq j} |(\mathbf{Z}_1^i \mathbf{D}_k^i + \mathbf{D}_k^i {\mathbf{Z}_2}^{iH} + \mathbf{E}_k^i)_{lj}|^2 \tag{19}$$

We define vectors $\mathbf{z}$ and $\mathbf{e}$ by stacking the elements of the matrices $\mathbf{Z}_1^i$, $\mathbf{Z}_2^i$ and $\mathbf{E}_k^i$ such that

$$\begin{aligned} \mathbf{z}^i = ((\mathbf{Z}_1^i)_{12}, (\mathbf{Z}_2^i)_{21}^*, (\mathbf{Z}_1^i)_{21}, (\mathbf{Z}_2^i)_{12}^*, \cdots, \\ (\mathbf{Z}_1^i)_{lj}, (\mathbf{Z}_2^i)_{jl}^*, (\mathbf{Z}_1^i)_{jl}, (\mathbf{Z}_2^i)_{lj}^*, \cdots)^T \end{aligned} \tag{20}$$

$$\begin{aligned} \mathbf{e}^i = ((\mathbf{E}_1^i)_{12}, (\mathbf{E}_1^i)_{21}, \cdots, (\mathbf{E}_1^i)_{lj}, (\mathbf{E}_1^i)_{jl}, \\ \cdots, (\mathbf{E}_k^i)_{lj}, (\mathbf{E}_k^i)_{jl}, \cdots)^T \end{aligned} \tag{21}$$

This vectorization step allows us to reformulate the problem in Eq. (19) into the linear least-squares problem

$$\min_{\mathbf{z}^i} \quad (\mathbf{J}^i \mathbf{z}^i + \mathbf{e}^i)^H (\mathbf{J}^i \mathbf{z}^i + \mathbf{e}^i) \tag{22}$$

where the $NK(N-1) \times 2N(N-1)$ complex matrix $\mathbf{J}^i$ is given by

$$\mathbf{J}^i = \begin{pmatrix} \mathbf{J}_1^i \\ \vdots \\ \mathbf{J}_K^i \end{pmatrix} \tag{23}$$

$$\mathbf{J}_k^i = \begin{pmatrix} [\mathcal{D}_k^i]_{12} & & & & \mathbf{0} \\ & \ddots & & & \\ & & [\mathcal{D}_k^i]_{lj} & & \\ & & & \ddots & \\ \mathbf{0} & & & & [\mathcal{D}_k^i]_{NN-1} \end{pmatrix} \tag{24}$$

where $\mathbf{J}_k^i$ are block-diagonal matrices, composed of the 4×2 block-diagonal complex matrices $[\mathcal{D}_k^i]_{lj}$ which are designed such that

$$[\mathcal{D}_k^i]_{lj} = \begin{pmatrix} (\mathbf{D}_k^i)_{jj} & (\mathbf{D}_k^i)_{ll} & 0 & 0 \\ 0 & 0 & (\mathbf{D}_k^i)_{ll} & (\mathbf{D}_k^i)_{jj} \end{pmatrix} \tag{25}$$

for $k = 1, ..., K$ and for each $1 \leq l < j \leq N$. This linear least-square problem has a well-known solution given by

$$\mathbf{z}^i = -\mathbf{J}^{i\dagger}\mathbf{e}^i = -(\mathbf{J}^{iH}\mathbf{J}^i)^{-1}\mathbf{J}^{iH}\mathbf{e}^i \tag{26}$$

An algorithm based on applying Eq. (26) iteratively, will be impractical due to the storage required and the computational power needed to handle the matrix $\mathbf{J}^i$ and calculate its pseudo-inverse. However, the structure of the matrix $\mathbf{J}^i$ will allow us to drastically reduce the required resources. Let us start by expanding the product $\mathbf{J}^{iH}\mathbf{J}^i$ into a block-diagonal matrix given by

$$\mathbf{J}^{iH}\mathbf{J}^i = \begin{pmatrix} \sum_k [\mathcal{D}_k^i]_{12}{}^H [\mathcal{D}_k^i]_{12} & & & \mathbf{0} \\ & \ddots & & \\ & & \sum_k [\mathcal{D}_k^i]_{lj}{}^H [\mathcal{D}_k^i]_{lj} & \\ & & & \ddots \\ \mathbf{0} & & & \sum_k [\mathcal{D}_k^i]_{NN-1}{}^H [\mathcal{D}_k^i]_{NN-1} \end{pmatrix} \tag{27}$$

The matrix inversion $(\mathbf{J}^{iH}\mathbf{J}^i)^{-1}$ can be avoided due to the block-diagonal structure. Thus, one can solve our linear problem using simple operations given by

$$(\mathbf{Z}_1^i)_{lj} = (\alpha_{l,j}\beta_{l,j} - \alpha_{l,l}\gamma_{j,l})/\rho_{l,j} \tag{28}$$

$$(\mathbf{Z}_1^i)_{jl} = (\alpha_{l,j}^*\beta_{j,l} - \alpha_{j,j}\gamma_{l,j})/\rho_{l,j} \tag{29}$$

$$(\mathbf{Z}_2^i)_{lj} = (\alpha_{l,j}\gamma_{j,l}^* - \alpha_{j,j}\beta_{l,j}^*)/\rho_{l,j} \tag{30}$$

$$(\mathbf{Z}_2^i)_{jl} = (\alpha_{l,j}^*\gamma_{l,j}^* - \alpha_{l,l}\beta_{j,l})/\rho_{l,j} \tag{31}$$

where $\alpha_{l,j}$, $\beta_{l,j}$, $\gamma_{l,j}$ and $\rho_{l,j}$ are given by

$$\alpha_{l,j} = \sum_{k=1}^{K} (\mathbf{D}_k^i)_{ll} (\mathbf{D}_k^i)_{jj}^* \tag{32}$$

$$\beta_{l,j} = \sum_{k=1}^{K} (\mathbf{D}_k^i)_{ll}^* (\mathbf{E}_k^i)_{lj} \tag{33}$$

$$\gamma_{l,j} = \sum_{k=1}^{K} (\mathbf{D}_k^i)_{ll}^* (\mathbf{E}_k^i)_{jl}^* \tag{34}$$

$$\rho_{l,j} = \alpha_{l,l}\alpha_{j,j} - \alpha_{l,j}^* \alpha_{l,j} \tag{35}$$

Recall that only the off-diagonal elements ($l \neq j$) need to be calculated, and the diagonal terms of $\mathbf{Z}_1^i$ and $\mathbf{Z}_2^i$ are set to zero. Compared to a solution based on applying Eq. (26), where performing inversion and multiplication of large matrices would have made the computational complexity of the order of $\mathcal{O}(KN^6)$, in the new proposed methodology, the computational complexity is notably reduced to $\mathcal{O}(KN^3)$. This reduction in complexity bears significance, particularly with the increase in the dimension N. We refer to this fast solution, described in Algorithm 1, as the GFFDiag algorithm for the Generalized FFDiag algorithm. For comparison, NNAJD [23], NNAJD-ALS [24] and CPD-ALS [39] algorithms have a computational complexity of order $\mathcal{O}(KN^3)$.

Compared to the state-of-the-art, the proposed approach has the same framework as in [22] and [23] where similar off-diagonal criterion was considered and the resulting algorithms are based on Jacobi-like algorithms. However, the proposed approach addresses the problem differently, using a matrix formulation to tackle the global problem. This formulation leads to the block diagonal structure in Eq. (23), Eq. (24), and Eq. 25 which results in a decoupled system of equations of size 4x4 that can be solved more efficiently than considering the direct solution of Eq. (26). Despite differences in the development methodologies, there are similarities between the resulting algorithm GFFDiag and the NNAJD algorithm proposed in [23]. However, both unitary and non-unitary cases are considered in this paper, unlike [23] where only the non-unitary case was considered.

Comments on the rectangular case

Until now, we have considered only the square case where $N_1 = N_2 = N$ in the model of Eq. (1). Let us consider the general case (rectangular) where $N_1 \neq N_2 \neq N$. Recall that the mixing matrices $\mathbf{A}_1 \in \mathbb{C}^{N_1 \times N}$ and $\mathbf{A}_2 \in \mathbb{C}^{N_2 \times N}$ are full column rank i.e. $N \leq N1$ and $N \leq N2$. The NHJD problem is to find the separation matrices $\mathbf{B}_1 \in \mathbb{C}^{N \times N_1}$ and $\mathbf{B}_2 \in \mathbb{C}^{N \times N_2}$ such that $\hat{\mathbf{D}}_k = \mathbf{B}_1 \mathbf{C}_k \mathbf{B}_2^H$ are as diagonal as possible for all $k = 1, ..., K$. Note that the matrices $\mathbf{B}_1$ and $\mathbf{B}_2$ are rectangular matrices such that $\mathbf{B}_1 = \mathbf{P}\boldsymbol{\Delta}_1 \mathbf{A}_1^\dagger$ and $\mathbf{B}_2 = \mathbf{P}\boldsymbol{\Delta}_2 \mathbf{A}_2^\dagger$ for a certain permutation matrix $\mathbf{P} \in \mathbb{R}^{N \times N}$ and a scaling diagonal matrices $\boldsymbol{\Delta}_1$ and $\boldsymbol{\Delta}_2 \in \mathbb{C}^{N \times N}$. Hence, one can initiate the diagonalization algorithm by any full-row rank matrices $\mathbf{B}_1^0$ and $\mathbf{B}_2^0$ for example $\mathbf{B}_1^0 = [\mathbf{I}_N \ \mathbf{0}_{N,N1-N}]$ and $\mathbf{B}_2^0 = [\mathbf{I}_N \ \mathbf{0}_{N,N2-N}]$. Next, we evaluate the $N \times N$ square matrices $\tilde{\mathbf{C}}_k = \mathbf{B}_1^0 \mathbf{C}_k \mathbf{B}_2^{0H}$ for $k = 1, ..., K$. Form now, one can use the GFFDiag algorithm on the $\tilde{\mathbf{C}}_k$ matrices, in order to find the diagonalization matrices $\tilde{\mathbf{B}}_1 \in \mathbb{C}^{N \times N}$ and $\tilde{\mathbf{B}}_2 \in \mathbb{C}^{N \times N}$ such that $\tilde{\mathbf{D}}_k = \tilde{\mathbf{B}}_1 \tilde{\mathbf{C}}_k \tilde{\mathbf{B}}_2^H$ are as diagonal as possible. Finally, one can simply deduce the separation matrices for rectangular matrices $\mathbf{C}_k$ using $\mathbf{B}_1 = \tilde{\mathbf{B}}_1 \mathbf{B}_1^0$ and $\mathbf{B}_2 = \tilde{\mathbf{B}}_2 \mathbf{B}_2^0$.

3.2. Unitary case

Matrix unitary property is an interesting feature that we encounter in many signal processing applications. Here, the unitary case corresponds to an additional constraint on the separation (or mixture) matrices $\mathbf{B}_1$ and $\mathbf{B}_2$. This problem, also known as joint SVD, is formulated as in Eq. (4). In order to solve this problem, we need to choose an updating rule that ensures the unitary property of $\mathbf{B}_1^i$ and $\mathbf{B}_2^i$ at each iteration. Let us recall some matrices notions that are required later.

The matrix exponential, denoted $\mathrm{Expm}(\mathbf{X})$, extends the concept of the exponential function to matrices and is defined by the series $\mathrm{Expm}(\mathbf{X}) = \sum_{k=0}^{\infty} \frac{\mathbf{X}^k}{k!}$. If $\mathbf{X}$ is diagonalizable such that $\mathbf{X} = \mathbf{U}\mathbf{D}\mathbf{U}^{-1}$, the matrix exponential can be efficiently computed as $\mathrm{Expm}(\mathbf{X}) = \mathbf{U}\,\mathrm{Expm}(\mathbf{D})\,\mathbf{U}^{-1}$, where $\mathrm{Expm}(\mathbf{D}) = \mathrm{diag}(e^{d_1}, \ldots, e^{d_N})$ and $\mathbf{D} = \mathrm{diag}(d_1, \ldots, d_N)$.

Algorithm 1 GFFDiag

Require: $\mathbf{C}_k$ for $k = 1, \cdots, K$, threshold τ and the maximum number of iterations N_{it}.

Ensure: $\mathbf{B}_1$, $\mathbf{B}_2$ and $\mathbf{D}_k$ for $k = 1, \cdots, K$.

1: **Initialization**: $\mathbf{B}_1^0 = \mathbf{B}_2^0 = \mathbf{I}_N$ and calculate $\mathbf{F}_k^0$ using Eq. (12).
2: **while** $\sum_{k=1}^{K} \|\text{off}(\mathbf{F}_k^i)\|_F^2 > \tau$ and ($i < N_{it}$) **do**
3: Find $\mathbf{D}_k^i$ and $\mathbf{E}_k^i$ using Eq. (14).
4: Calculate $\alpha_{l,j}$, $\beta_{l,j}$, $\gamma_{l,j}$ and $\rho_{l,j}$ using Eq. (32), Eq. (33), Eq. (34) and Eq. (35).
5: Calculate $\mathbf{Z}_1^i$ and $\mathbf{Z}_2^i$ using Eq. (28) , Eq. (29) , Eq. (30) and Eq. (31).
6: Normalize $\mathbf{Z}_1^i$ and $\mathbf{Z}_2^i$ using Eq. (8).
7: Update the estimation of $\mathbf{B}_1^i$ and $\mathbf{B}_2^i$ using Eq. (6).
8: Calculate $\mathbf{F}_k^{i+1}$ for $k = 1, ..., K$.
9: **end while**

The matrix exponential is closely related to unitary matrices. Specifically, if $\tilde{\mathbf{X}}$ is skew-Hermitian[1], then its matrix exponential $\mathbf{Y} = \text{Expm}(\tilde{\mathbf{X}})$ is a unitary matrix, satisfying $\mathbf{Y}^H\mathbf{Y} = \mathbf{I}$.

Since the product of two unitary matrices remains a unitary matrix, we propose to solve the problem presented in Eq.(4) using the same approach as in the non-unitary case, with the final update modified to incorporate a multiplicative rule:

$$\mathbf{B}_m^{i+1} = \text{Expm}(\tilde{\mathbf{Z}}_m^i)\mathbf{B}_m^i \quad \text{for} \quad m = 1, 2 \tag{36}$$

Assuming that $\mathbf{B}_1^i$ and $\mathbf{B}_2^i$ are unitary in the previous iteration (we choose a unitary initialization), solving our problem comes down to finding $\tilde{\mathbf{Z}}_1^i$ and $\tilde{\mathbf{Z}}_2^i$ that minimize the off-diagonal criterion under the constraint to be skew-Hermitian.

This multiplicative update is closely related to the first additive update

[1] A square matrix $\mathbf{X}$ is said to be skew-Hermitian if $\mathbf{X} + \mathbf{X}^H = 0$. Thus, for any square matrix $\mathbf{X}$, the matrix $\tilde{\mathbf{X}} = \mathbf{X} - \mathbf{X}^H$ is inherently skew-Hermitian.

used in the non-unitary case. In fact, if we take the first-order exponential approximation of the update in Eq. (36), we recover approximately the additive type of update in Eq. (6) as it is shown below:

$$\mathbf{B}_m^{i+1} = \text{Expm}(\tilde{\mathbf{Z}}^i{}_m)\mathbf{B}_m^i \tag{37}$$

$$= (\mathbf{I} + \tilde{\mathbf{Z}}_m^i + (\tilde{\mathbf{Z}}_m^i)^2/2 + \dots)\mathbf{B}_m^i \tag{38}$$

$$\approx (\mathbf{I} + \tilde{\mathbf{Z}}_m^i + \mathcal{O}(\tilde{\mathbf{Z}}_m^i))\mathbf{B}_m^i \tag{39}$$

Since the estimated matrices $\tilde{\mathbf{Z}}_1^i$ and $\tilde{\mathbf{Z}}_2^i$ must be skew-Hermitian at each iteration, only one of each pair of (l, j) entries of these matrices need to be calculated. We then use the relation $\tilde{\mathbf{Z}}_1^i = \mathbf{Z}_1^i - \mathbf{Z}_1^{iH}$ and $\tilde{\mathbf{Z}}_2^i = \mathbf{Z}_2^i - \mathbf{Z}_2^{iH}$ to force both $\tilde{\mathbf{Z}}_1^i$ and $\tilde{\mathbf{Z}}_2^i$ to be skew-Hermitians. Compared to the non-unitary case, the computation time is reduced by half if an approximate unitary version is considered, i.e. $\tilde{\mathbf{Z}}_1^i$ and $\tilde{\mathbf{Z}}_2^i$ are skew-Hermitians and we use the update given by Eq. (6). If we are looking for a strict unitary version, we use the matrix exponential, i.e. $\tilde{\mathbf{Z}}_1^i$ and $\tilde{\mathbf{Z}}_2^i$ are skew-Hermitians and we use the update given by Eq. (36). In that case, we add the matrix exponential's computational complexity, which is of the order of $\mathcal{O}(N^3)$. We refer to this algorithm, described in Algorithm 2, as UGFFDiag for the Unitary Generalized FFDiag algorithm and as UGFFDiag-Approx when the approximated unitary is considered in step 6.

4. Numerical simulations

In this section, we present the numerical simulations that assess the performance of the proposed algorithms. They are compared to NNAJD [23], NNAJD-ALS [24] and CPD-ALS [39] (implemented in Tensorlab toolbox [40]) algorithms as representative of the state-of-art. Also, in the case of unitary diagonalization, we compare the proposed algorithm UGFFDiag with the NOHTJD [25] and CPD3-SGSD [41] (implemented in Tensorlab toolbox [40]) algorithms. Note that CPD-ALS and NOHTJD algorithms are designed for the diagonalization of higher-order tensors, but we are considering only the studied case of third-order

Algorithm 2 UGFFDiag

Require: $\mathbf{C}_k$ for $k = 1, \cdots, K$, threshold τ and the maximum number of iterations N_{it}.

Ensure: $\mathbf{B}_1$, $\mathbf{B}_2$ unitary and $\mathbf{D}_k$ for $k = 1, \cdots, K$.

1: **Initialization**: $\mathbf{B}_1^0 = \mathbf{B}_2^0 = \mathbf{I}_N$ and calculate $\mathbf{F}_k^0$ using Eq. (12).

2: **while** $\sum_{k=1}^{K} \|\text{off}(\mathbf{F}_k^i)\|_F^2 > \tau$ and (#iteration $< N_{it}$) **do**

3: Find $\mathbf{D}_k^i$ and $\mathbf{E}_k{}^i$ using Eq. (14).

4: Calculate $\alpha_{l,j}$, $\beta_{l,j}$, $\gamma_{l,j}$ and $\rho_{l,j}$ using Eq. (32), Eq. (33), Eq. (34) and Eq. (35) for $l < j$.

5: Calculate $\tilde{\mathbf{Z}}_1^i$ and $\tilde{\mathbf{Z}}_2^i$ using

$$\begin{cases} \text{Eq. (28)} & \text{and} \quad (\tilde{\mathbf{Z}}_1^i)_{j,l} = -(\tilde{\mathbf{Z}}_1^i)_{l,j}^* \\ \text{Eq. (30)} & \text{and} \quad (\tilde{\mathbf{Z}}_2^i)_{j,l} = -(\tilde{\mathbf{Z}}_2^i)_{l,j}^* \end{cases}$$

6: Update the estimation of $\mathbf{B}_1^i$ and $\mathbf{B}_2^i$ using

$$\begin{cases} \text{Eq. (6)} & \text{Approximately unitary} \\ \text{Eq. (36)} & \text{Strictly unitary} \end{cases}$$

7: Calculate $\mathbf{F}_k^{i+1}$ for $k = 1, ..., K$

8: **end while**

tensors. The parameters for all algorithms are in accordance with literature recommendations.

Performance factors

In order to compare the capability of the proposed methods to recover mixing matrices $\mathbf{A}_1$ and $\mathbf{A}_2$, we adopt the following performance index PI:

$$PI(\mathbf{G_1}, \mathbf{G_2}) = Pi(\mathbf{G_1}) + Pi(\mathbf{G_2}) \tag{40}$$

The matrices $\mathbf{G_1}$ and $\mathbf{G_2}$ are normalized global matrices defined as $\mathbf{G_1} = \mathbf{B}_1\mathbf{A}_1$ and $\mathbf{G_2} = \mathbf{B}_2\mathbf{A}_2$. The function $Pi(\mathbf{G})$ is defined as in [22], which is invariant

to the scaling and permutation ambiguities of the considered problem.

$$Pi(\mathbf{G}) = \frac{1}{2N(N-1)}\Big[\sum_{l=1}^{N}(\sum_{j=1}^{N}\frac{|\mathbf{G}_{lj}|^2}{\max_k(|\mathbf{G}_{lk}|^2)} - 1) + \sum_{j=1}^{N}(\sum_{l=1}^{N}\frac{|\mathbf{G}_{lj}|^2}{\max_k(|\mathbf{G}_{kj}|^2)} - 1)\Big] \tag{41}$$

In order to express the diagonalization error, we use the performance factor based on the squared normalized off-diagonal elements of the reconstructed diagonal matrices as explained below:

$$\mathbf{J}(\mathbf{B}_1, \mathbf{B}_2) = \sum_{k=1}^{K} Pi\Big(\mathbf{B}_1\mathbf{C}_k\mathbf{B}_2^H\Big) \tag{42}$$

Simulation 1

The target matrices are generated according to the model given by:

$$\mathbf{C}_k = \mathbf{A}_1\mathbf{D}_k\mathbf{A}_2^H + \epsilon\mathbf{E}_k \quad \text{for} \quad k = 1, ..., K \tag{43}$$

where, for $k = 1, ..., K$, the $N \times N$ complex matrices $\mathbf{A}_1$,$\mathbf{A}_2$, and $\mathbf{E}_k$ are drawn randomly with their real and imaginary elements are i.i.d. from the standard normal distribution $\mathcal{N}(0, 1)$. The complex diagonal matrices $\mathbf{D}_k$ have also i.i.d diagonal entries drawn from the same distribution. The scalar factor ϵ is related to the $SNR(dB)$ by the following equation:

$$SNR = 10\log_{10}\Big(\frac{\|\mathbf{A}_1\mathbf{D}_k\mathbf{A}_2^H\|_F^2}{\epsilon^2\|\mathbf{E}_k\|_F^2}\Big) \tag{44}$$

In the first scenario, we consider 100 Monte Carlo realizations of the noiseless case with the parameters $N = K = 10$. We evaluate the mean recovering error PI and the off-diagonalization error for 300 iterations of every algorithm. Figure 1 shows that all algorithms reach perfect recovery of the separation matrices, however, the proposed algorithm has slightly faster convergence than the NNAJD algorithm and significantly faster convergence than the NNAJD-ALS and CPD-ALS algorithms in terms of iterations (approximately 15 iterations). This behaviour is confirmed in Fig. 2 where we plot the off-diagonalization error of each algorithm. We prefer this kind of plot where we track the performance

for different random realizations to illustrate the robustness of the algorithm. It can be observed that the NNAJD-ALS and CPD-ALS algorithms require a greater number of iterations to converge than the NNAJD and the proposed algorithm. In contrast, the proposed algorithm requires fewer than 15 iterations, whereas the NNAJD algorithm requires up to 20 iterations. This demonstrates the superior convergence speed of the proposed method in terms of iteration number.

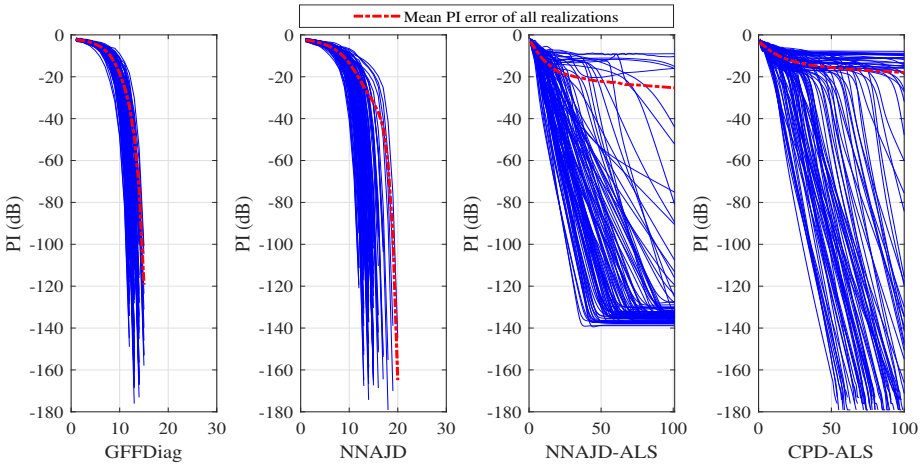


Figure 1: Recovering error $PI(dB)$ versus the number of iterations for 100 different realizations with $N = K = 10$ in the noiseless case.

In the second scenario, we consider the noisy case with $SNR = 20dB$, while keeping the other parameters unchanged ($N = K = 10$). We distinguish in Fig. 3 and Fig. 4 a faster convergence of the GFFDiag algorithm compared to CPD-ALS and NNAJD-ALS algorithms, even though the mean squared error PI reaches approximately the same levels. However, in terms of off-diagonalization error, the proposed algorithm reaches in a small number of iterations low levels compared to these algorithms. Compared to the NNAJD algorithm, the proposed algorithm has approximately the same performance in terms of mean squared error PI and off-diagonalization error.

As stated before, one of the key advantages of the proposed algorithm GFFDiag is its reduced computational complexity. Table 1 presents the mean exe-

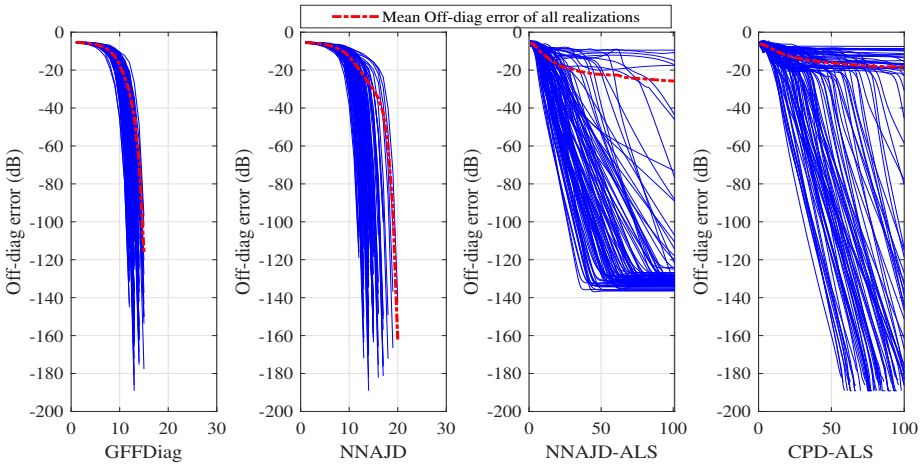


Figure 2: Off-diagonalization error versus the number of iterations for 100 different realizations with $N = K = 10$ in the noiseless case.

cution time per 100 iterations of all four algorithms for different cases of K and N. The simulation was run in MATLAB on a PC with an i7-9750H CPU and 16 Go RAM. In addition to the fast convergence of the GFFDiag algorithm in terms of the number of iterations, as seen in previous simulations, Tab. 1 demonstrates that it is the quickest algorithm regarding the execution time required per iteration.

In the third scenario, we study the effect of changing the number of slices K on the performance. Hence, we consider a $SNR = 20dB$ and $N = 8$ while varying K from 2 until 62. After 100 iterations of each algorithm, the results in Fig. 5b and Fig. 5a demonstrate that the NNAJD and GFFDiag algorithms exhibit comparable performance. In comparison to the NNAJD-ALS and CPD-ALS algorithms, this performance is markedly superior in terms of off-diagonalization error, particularly for small values of K in comparison to N. In terms of mean recovering error PI, the performance is superior when $K < N$, and approximately equivalent when K continues to grow, with the NNAJD-ALS algorithm showing an advantage when K is large in comparison to N.

In the fourth scenario, we study the effect of changing the matrix dimension N on the performance. We consider a $SNR = 20dB$ and $K = 8$ while varying

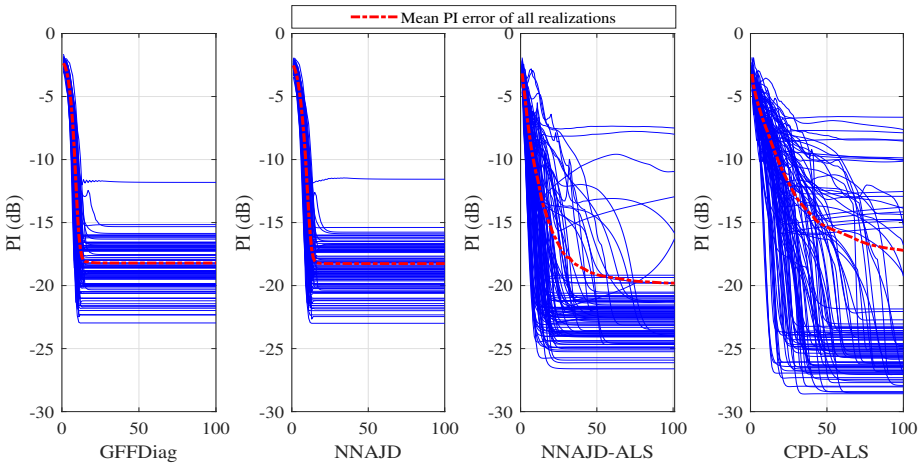


Figure 3: Recovering error $PI(dB)$ versus the number of iterations for 100 different realizations with $N = K = 10$ and $SNR = 20dB$.

N from 2 until 62. After 100 iterations, Fig. 6b and Fig. 6a demonstrate that the GFFDiag algorithm performs similarly to the NNAJD algorithm. In terms of off-diagonalization error, the GFFDiag algorithm outperforms the other two algorithms, particularly when $N > K$. The CPD-ALS algorithm is shown to have the best performance in terms of mean recovering error PI when N is large compared to K. However, the proposed algorithm is observed to have good performance for all values of N.

In the fifth scenario, we investigate the behaviour of the proposed algorithm for different SNR values. We consider $N = K = 10$, 100 iterations, and we increase the SNR from 0 until $50dB$. In terms of mean recovering error PI, Fig. 7a shows that the proposed algorithm has a similar performance to NNAJD algorithm and a comparable performance for low SNR values compared to ALS-like algorithms. However, for high SNR values, the proposed algorithm outperforms the ALS-like algorithms. This is related to the considered objective function, where in ALS-like algorithms, the least-squares function is considered, which is known to be more robust for low SNR values. The proposed algorithm and the NNAJD algorithms are based on the off-diagonalization error function which is more effective in high SNR (more noticeable in the noiseless case).

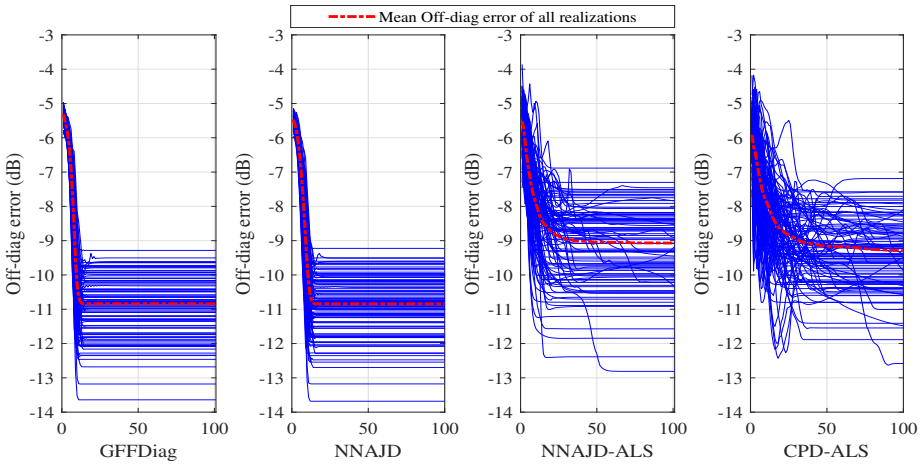
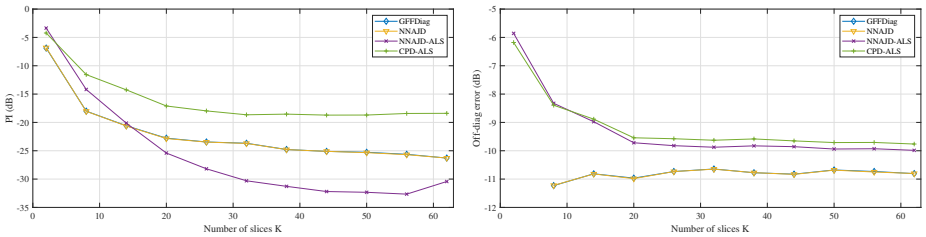


Figure 4: Off-diagonalization error versus the number of iterations for 100 different realizations with $N = K = 10$ and $SNR = 20dB$.



(a) (b)

Figure 5: Mean recovering error PI and off-diagonalization error versus K for 100 realizations after 100 iterations with $N = 8$ and $SNR = 20dB$.

This also explains the superiority of the proposed algorithm in terms of off-diagonalization error since it is the actual function that has to be minimized.

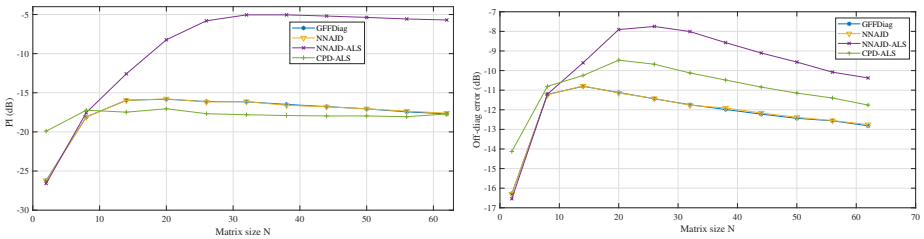
Simulation 2

In this subsection, we consider the unitary case, where the targeted matrices are generated according to the same model presented in Eq. (43) while forcing the matrices $\mathbf{A}_1$ and $\mathbf{A}_2$ to be unitary.

In the first scenario, we consider 100 Monte Carlo realizations of the noiseless case with the parameters $N = K = 10$. Figure 8 shows that, after only 11 iterations, the UGFFDiag algorithm outperforms the other algorithms in

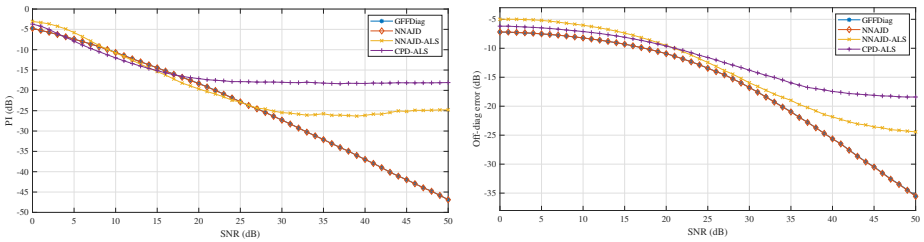
N	K	GFFDiag	NNAJD	NNAJD-ALS	CPD-ALS
5	5	**49**	111	106	566
10	10	**98**	295	212	551
20	20	**272**	925	913	711
8	4	**61**	213	145	542
8	30	**167**	307	259	615
8	60	**264**	363	378	661
4	8	**56**	80	110	540
30	8	**283**	1744	1280	746
60	8	**893**	5538	11822	2350

Table 1: Execution time per 100 iterations (ms) for non-unitary algorithms.



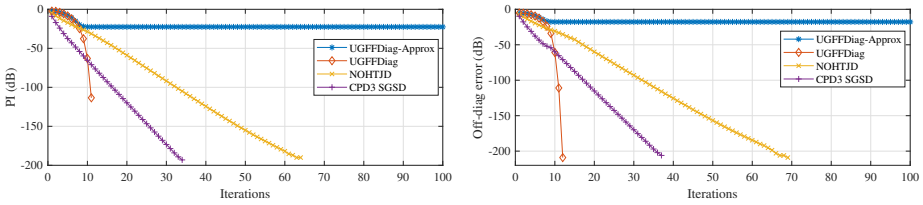
(a) (b)

Figure 6: Mean recovering error PI and off-diagonalization error versus N for 100 realizations after 100 iterations with $K = 8$ and $SNR = 20dB$.



(a) (b)

Figure 7: Mean recovering error PI and off-diagonalization error versus $SNR(dB)$ for 100 realizations after 100 iterations with $N = K = 10$.



(a) (b)

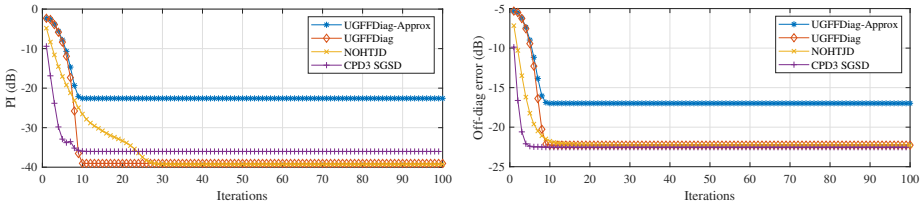
Figure 8: Mean recovering error PI and off-diagonalization error versus the number of iterations for 100 different realizations with $N = K = 10$ in the noiseless case.

terms of mean recovering error PI and off-diagonalization error. The approximated unitary version of the proposed algorithm UGFFDiag-Approx struggles to converge to the exact solution. This can be explained by evaluating the deviation from the unitarity of the matrices $\mathbf{B}_1$ and $\mathbf{B}_2$, which can be calculated by $PI(\mathbf{B}_1^H\mathbf{B}_1, \mathbf{B}_2^H\mathbf{B}_2)$. The UGFFDiag and NOHTJD algorithms have approximately null deviation, whereas, the CPD3-SGSD and UGFFDiag-Approx algorithms have 5×10^{-4} and 2.5×10^{-2}, respectively.

In the second scenario, we repeat the same experience but with a $SNR = 20dB$, keeping the other parameters unchanged ($N = K = 10$). Figure 9 illustrates that the UGFFDiag algorithm converges to the lowest level of mean recovering error PI and off-diagonalization error compared to the other algorithms. The NOHTJD algorithm requires more iterations to reach the same level of PI. Table 2 shows the mean computational time for each algorithm per 100 iterations for different parameters N and K. The UGFFDiag and UGFFDiag-Approx algorithms require less time per iteration compared to NOHTJD and CPD3-SGSD algorithms.

In the third scenario, we study the effect of changing the number of slices K. Hence, we consider a $SNR = 20dB$ and $N = 8$ while varying K from 4 until 30. After 100 iterations of each algorithm, the results in Fig. 10 show that, except the UGFFDiag-Approx algorithm, increasing K enhances the performance for all algorithms.

In the fourth scenario, we investigate the effect of increasing the matrix size

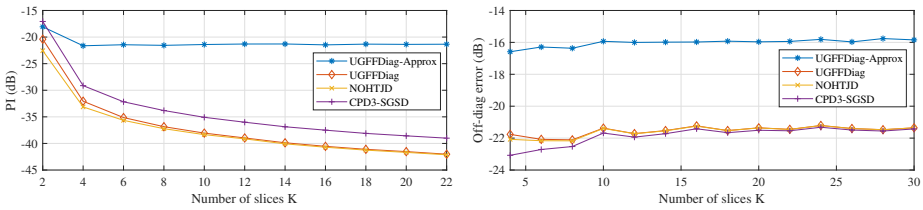


(a) (b)

Figure 9: Mean recovering error PI and off-diagonalization error versus the number of iterations for 100 different realizations with $N = K = 10$ and $SNR = 20dB$.

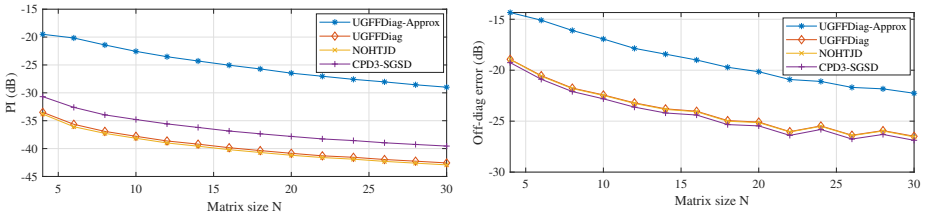
N	K	UGFFDiag	UGFFDiag-Approx	NOHTJD	CPD3-SGSD
10	10	**147**	**85**	4720	755
20	20	**317**	**221**	23057	3529
8	4	**121**	**52**	2949	415
8	30	**168**	**146**	3026	728
8	60	**205**	**161**	3086	1301
4	8	**54**	**35**	631	159
30	8	**326**	**247**	52199	3882
60	8	**964**	**777**	408978	26652

Table 2: Execution time per 100 iterations (ms) for unitary algorithms.



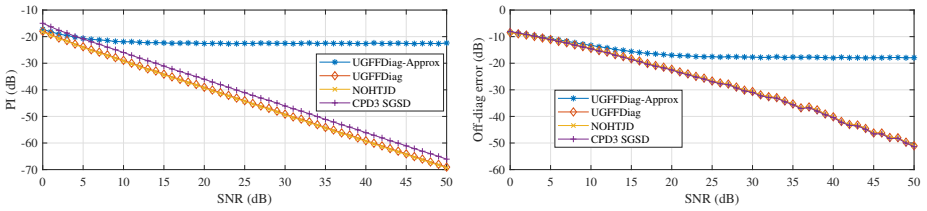
(a) (b)

Figure 10: Mean recovering error PI and off-diagonalization error versus K for 100 realizations after 100 iterations with $N = 8$ and $SNR = 20dB$.



(a) (b)

Figure 11: Mean recovering error PI and off-diagonalization error versus M for 100 realizations after 100 iterations with $K = 8$ and $SNR = 20dB$.



(a) (b)

Figure 12: Mean recovering error PI and off-diagonalization error versus $SNR(dB)$ for 100 realizations after 100 iterations with $N = K = 10$.

N from 4 until 30, while fixing $K = 8$ for $SNR = 20dB$. Figure 11 illustrates that increasing the matrix size M enhances the performance. UGFFDiag and NOHTJD algorithms have the lowest PI, followed by the CPD3-SGSD algorithm, then the UGFFDiag-Approx algorithm.

In the fifth scenario, we plot the mean recovering error PI and the off-diagonalization error versus SNR for the parameters $N = 10$ and $K = 10$. Figure 12 shows that UGFFDiag and NOHTJD algorithms have the same performance which enhances when increasing SNR. The UGFFDiag-Approx algorithm has the highest error compared to the other algorithms.

Real world data

We will consider two scenarios where the proposed algorithm will be applied to the JBSS problem. The first scenario is blind speech separation, in which we consider synthetic mixtures of real speech recordings. In this case, we have

access to the ground truth (simulated) mixing matrices. The second scenario is the extraction of fetal ECG from maternal ECG, in which real-world acquisitions are used. However, the ground truth mixing matrices in this case are unknown.

1. Blind speech sources separation: In this scenario, we consider $N1 = N2 = 6$ audio signals, each of 7 seconds in duration, sampled at 8 kHz. These signals consist of 3 male and 3 female adults reading short English texts. The instantaneous mixing model is considered which effectively simulates realistic approximations of speech signal mixtures in scenarios where reverberation and time delays are negligible. We generated two mixed datasets, $\mathbf{x}_1$ and $\mathbf{x}_2$ such that:

$$\mathbf{x}_i(t) = \mathbf{A}_i\mathbf{s}_i(t) + \mathbf{n}_i(t) \qquad \text{for} \qquad \text{i=1,2.} \tag{45}$$

where $\mathbf{s}_1(t)$ and $\mathbf{s}_2(t)$ are signal vectors. In order to ensure the correlation between the two datasets, the second dataset's sources are synthesized as in [42][25]. This involves randomly permuting and scaling each speech signal by a random factor (uniformly distributed between 0 and 1). The mixing matrices $\mathbf{A_1} \in \mathcal{R}^{6\times 6}$ and $\mathbf{A_2} \in \mathcal{R}^{6\times 6}$ are randomly generated according to a Gaussian distribution, with zero mean and unit variance. The noise vectors $\mathbf{n}_1$ and $\mathbf{n}_2$ are a zero-mean additive random white noise uncorrelated with the signal. The noise power is fixed according to the SNR level defined as $SNR(dB) = 10\log_{10}\frac{\mathbf{E}[\|\mathbf{A_i s}(t)\|_F^2]}{\mathbf{E}[\|\mathbf{n}_i\|_F^2]}$.

In order to transform the JBSS problem into a NHJD problem, we have followed the same framework as in [19][5][23] by estimating time-lagged cross-correlation matrices as:

$$\mathbf{C}_k = \mathbb{E}[\mathbf{x}_1(t)\mathbf{x}_2(t+\tau_k)^H] \quad \text{for} \quad k = 1, ..., K. \tag{46}$$

Under the mild assumption that sound sources are decorrelated, the model in (45) is equivalent to the NHJD model (1). The number of estimated time-lagged cross-correlation matrices is set at $K = 10$. The GFFDiag algorithm was employed for the blind separation of mixed signals across a range of SNR levels. For example, at $SNR = 30dB$, Fig. 13 demonstrates the effectiveness of the proposed algorithm in successfully separating the mixed signals. The per-

mutation ambiguity has been resolved using the correlation between the original sources and the separated sources. We compared the proposed algorithm with NNAJD, NNAJD-ALS, and CPD-ALS algorithms. In addition, we added two JBSS algorithms in the comparison: SOBI [14] algorithm which is based on the diagonalization of a set of delayed correlation matrices, and JADE [1] algorithm which diagonalizes eigen-matrices based on forth-order moments. Both algorithms are evaluated for each set $\mathbf{x}_1$ and $\mathbf{x}_2$ separately, then the averaged mean recovery errors between $\mathbf{A}_1$ and $\mathbf{A}_2$ was considered. The same reasoning holds for off-diagonalization errors. Figure14b demonstrates that the proposed algorithm outperforms all other methods in terms of off-diagonalization errors across different SNR levels. Regarding the mean recovery error (PI), GFFDiag exhibits superior performance at low SNR levels, while the JADE algorithm achieves the best performance at higher SNR values. This disparity can be attributed to the fact that JADE leverages higher-order statistics, whereas the other algorithms, including GFFDiag, rely on second-order statistics such as correlations and cross-correlations.

2. Extraction of fetal ECG: Accurate interpretation of electrocardiograms (ECG) is essential for diagnosing a wide range of cardiac conditions. The non-invasive extraction of fetal electrocardiograms (FECG) is particularly crucial for medical diagnostics during pregnancy [43]. However, FECG signals are often mixed with maternal ECG (MECG) signals, along with noise from electrode movement or maternal motion, making the separation of ECG sources from the observed data a complex challenge. In this study, we considered signals available on Lieven De Lathauwer's personal webpage, as used in [44]. These signals were recorded using eight skin electrodes placed at various locations on a pregnant woman's body. The signals were sampled at 250 Hz, with a total recording duration of 10 seconds. The first five recordings correspond to electrodes positioned on the mother's abdominal region, while the remaining three were placed on the thoracic region. Figure 15 shows the recorded 8-channel ECG. Unlike classical methods that rely on using all 8 channels simultaneously and without prior information, we propose a framework that takes advantage of the spatial

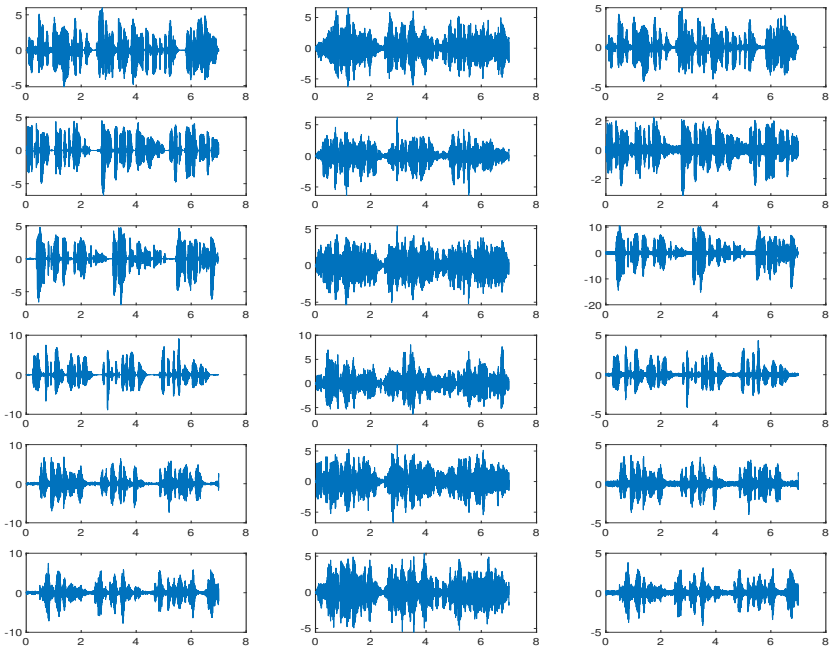
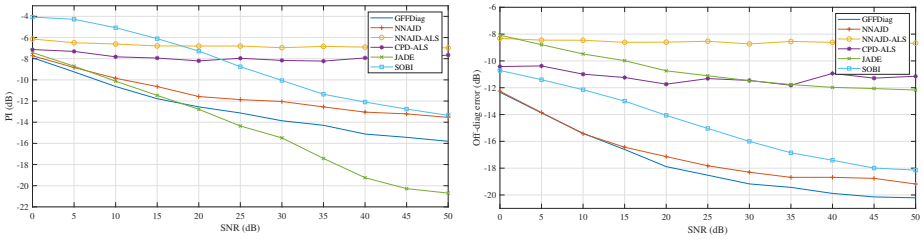


Figure 13: Illustration of the amplitude vs time (s) of the 6 sources (Left), the mixed signals (middle), and the separated signals (right), for $SNR = 30dB$.



(a) (b)

Figure 14: Mean recovering error PI and off-diagonalization error versus $SNR(dB)$ for 100 realizations after 100 iterations with $N = 6$ and $K = 10$.

configuration of the electrodes by dividing the channels into two sub-groups: 5 abdominal electrodes, which are more likely to capture both maternal and fetal cardiac activity, and 3 thoracic electrodes, which mainly capture maternal cardiac activity. The diagonalization of the delayed cross-correlation matrices of those two datasets will eventually capture both inter-set dependencies (same sources observed from different electrode positions) and intra-set independence (independent sources).

Hence, we have considered the two datasets: $\mathbf{x}_1$ for the recordings from the abdominal region and $\mathbf{x}_2$ for the recordings from the thoracic region ($N_1 = 5, N_2 = 3$). After estimating $K = 20$ delayed cross-correlation matrices, we have applied the rectangular version of the GFFDiag algorithm in order to separate $N = 3$ sources (FECG, MECG and noise). In classical separation methods, the number of sources is generally assumed equal to the number of channels. In cases where this condition is not met (as we suppose here), dimension reduction, frequently achieved through Principal Component Analysis (PCA), is required. For comparison, we have used JADE, INFOMAX [45], SOBI and CPD-ALS to evaluate the proposed algorithm GFFDiag. By visually comparing the results of the separation using these algorithms, as shown in Fig. 16, we can identify the first separated signal (a normalization and permutation has been done) as the FECG by its higher frequency compared to the estimated MECG signal. There is no ground truth or standard of actual fetal ECG recordings that can be used as a reference to quantify and benchmark the performance of these algorithms. Nevertheless, we have considered the performance factor described in [46] where maternal and fetal R-peak positions are estimated from the mixed ECG and the estimated fECG. Those positions are used to evaluate the mean values of the interfering (maternal) and desired (fetal) ECGs at the R-peak positions in the estimated fECG. This provides an estimate of the residual mECG in the extracted fECG. If the mECG has been effectively removed, the residual values should be minimal. At the same time, the estimated fECG values at the R-peak positions should approximate those of the respective points in the original mixed signals. Note that all signals have been centred and normalized

	Mean of maternal R-peak values	Mean of fetal R-peak values
Original mixture	7.05 ± 0.86	2.83 ± 1.04
GFFDiag	**0.95 ± 0.64**	**3.49 ± 1.47**
SOBI	2.27 ± 1.38	2.87 ± 1.66
JADE	1.82 ± 0.96	3.13 ± 1.49
INFOMAX	1.74 ± 0.92	2.96 ± 1.58
CPD-ALS	1.76 ± 1.10	3.38 ± 1.56

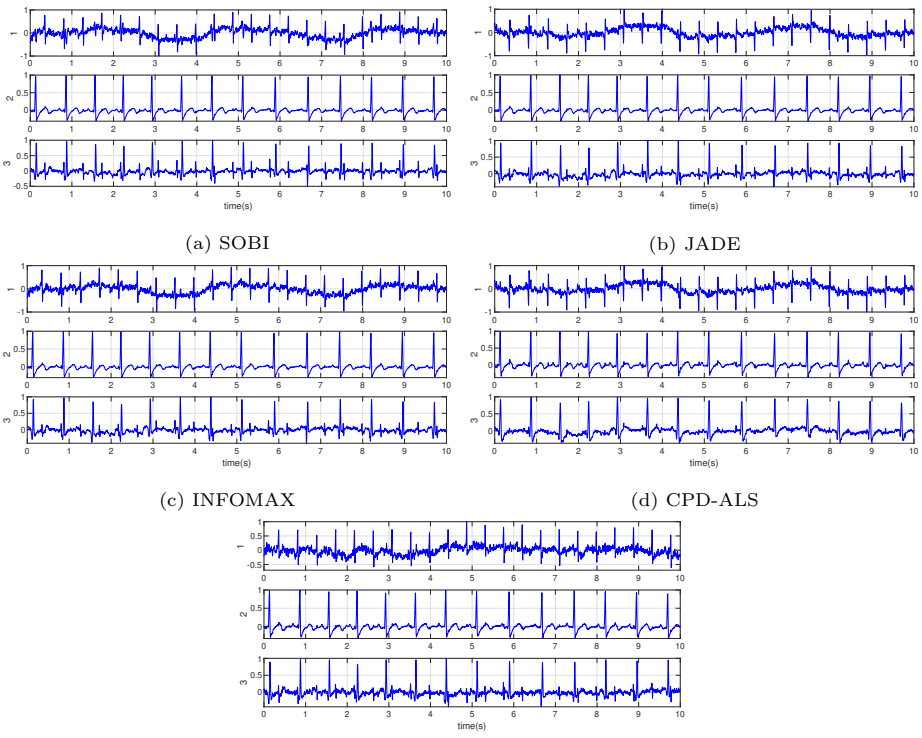
Table 3: Maternal and fetal R peak values in the estimated fECG (mean±standard deviation)$\times 10^{-2}$.

to the unit norm. Table 3 illustrates a notable decrease in the maternal R-peak amplitudes in comparison to the fetal R-peak amplitudes, which remain comparable to their values in the original mixture. This outcome indicates effective separation. Furthermore, the proposed GFFDiag algorithm achieves the lowest maternal R-peak values and the highest fetal R-peak values, which highlights its superior performance in extracting the fECG while minimizing maternal signal interference. A more advanced comparison study requires a larger database and collaboration with medical experts to thoroughly evaluate the quality of separation.

5. Conclusion

This paper investigates the problem of jointly diagonalizing a set of non-Hermitian matrices. It generalizes the approach of the FFDiag algorithm for Hermitian matrices, to address non-Hermitian matrices. We developed algorithms for both the unitary and the non-unitary diagonalization scenarios, namely GFFDiag, UGFFDiag, and its variant UGFFDiag-Approx. In both scenarios, the numerical simulations show that the proposed algorithms outperform existing algorithms while being more computationally efficient.

Figure 15: Eight ECG channels are presented over time(s): the initial five lie in the abdominal region, while the last three lie in the thoracic region.



(a) SOBI

(b) JADE

(c) INFOMAX

(d) CPD-ALS

(e) GFFDiag

Figure 16: Results of separating FECG and MECG with SOBI, JADE, INFOMAX, CPD-ALS and GFFDiag algorithms.

References

[1] J. F. Cardoso, A. Souloumiac, Blind beamforming for non-Gaussian signals, IEEE Proceedings F - Radar and Signal Processing 140 (6) (1993) 362–370. doi:10.1049/ip-f-2.1993.0054.

[2] J. Coloigner, L. Albera, A. Kachenoura, F. Noury, L. Senhadji, Semi-nonnegative joint diagonalization by congruence and semi-nonnegative ica, Signal Processing 105 (2014) 185–197. doi:https://doi.org/10.1016/j.sigpro.2014.05.017.

[3] Y. Xiang, K. Abed-Meraim, Y. Hua, New iterative techniques for second order blind source separation, in: Proc. IEEE Int. Work on Intelligent Sig. Proc. and Comm. Sys. (Melbourne), Deakin University, 1998, pp. 289 – 293.

[4] S. Choi, A. Cichocki, H.-M. Park, S.-Y. Lee, Blind source separation and independent component analysis: A review, Neural Information Processing-Letters and Reviews 6 (1) (2005) 1–57.

[5] X.-L. Li, T. Adalı, M. Anderson, Joint blind source separation by generalized joint diagonalization of cumulant matrices, Signal Processing 91 (10) (2011) 2314–2322. doi:https://doi.org/10.1016/j.sigpro.2011.04.016.

[6] A. Mesloub, K. Abed-Meraim, A. Belouchrani, A new algorithm for complex non-orthogonal joint diagonalization based on shear and givens rotations, IEEE Transactions on Signal Processing 62 (8) (2014) 1913–1925. doi:10.1109/TSP.2014.2303947.

[7] S. Meziani, A. Mesloub, A. Belouchrani, K. Abed-Meraim, Extended joint evd algorithm for widely linear arma source separation, IEEE Transactions on Signal Processing 71 (2023) 3667–3678. doi:10.1109/TSP.2023.3322807.

[8] X. Liu, S. Bourennane, C. Fossati, Denoising of hyperspectral images using the parafac model and statistical performance analysis, IEEE Transactions on Geoscience and Remote Sensing 50 (10) (2012) 3717–3724.

[9] W.-T. Zhang, S.-T. Lou, X.-J. Li, J. Guo, Tracking multiple targets in mimo radar via adaptive asymmetric joint diagonalization, IEEE Transactions on Signal Processing 64 (11) (2016) 2880–2893.

[10] T.-Q. Xia, Joint diagonalization based 2d-dod and 2d-doa estimation for bistatic mimo radar, Signal Processing 116 (2015) 7–12. doi:https://doi.org/10.1016/j.sigpro.2015.04.014.

[11] N. Sidiropoulos, G. Giannakis, R. Bro, Blind parafac receivers for ds-cdma systems, IEEE Transactions on Signal Processing 48 (3) (2000) 810–823. doi:10.1109/78.824675.

[12] Z. Zhou, J. Fang, L. Yang, H. Li, Z. Chen, S. Li, Channel estimation for millimeter-wave multiuser mimo systems via parafac decomposition, IEEE Transactions on Wireless Communications 15 (11) (2016) 7501–7516. doi:10.1109/TWC.2016.2604259.

[13] Z. Zhou, J. Fang, L. Yang, H. Li, Z. Chen, R. S. Blum, Low-rank tensor decomposition-aided channel estimation for millimeter wave mimo-ofdm systems, IEEE Journal on Selected Areas in Communications 35 (7) (2017) 1524–1538.

[14] A. Belouchrani, K. Abed-Meraim, J.-F. Cardoso, E. Moulines, A blind source separation technique using second-order statistics, IEEE Transactions on Signal Processing 45 (2) (1997) 434–444. doi:10.1109/78.554307.

[15] A. Ziehe, P. Laskov, G. Nolte, K.-R. MÃžller, A fast algorithm for joint diagonalization with non-orthogonal transformations and its application to blind source separation, Journal of Machine Learning Research 5 (Jul) (2004) 777–800.

[16] G.-H. Cheng, S.-M. Li, E. Moreau, New jacobi-like algorithms for non-orthogonal joint diagonalization of hermitian matrices, Signal Processing 128 (2016) 440–448.

[17] P. Ablin, J.-F. Cardoso, A. Gramfort, Faster independent component analysis by preconditioning with hessian approximations, IEEE Transactions on Signal Processing 66 (15) (2018) 4040–4049. doi:10.1109/TSP.2018.2844203.

[18] S. Saito, K. Oishi, An alternating least-squares algorithm for approximate joint diagonalization of hermitian matrices, Digital Signal Processing 129 (2022) 103633. doi:https://doi.org/10.1016/j.dsp.2022.103633.

[19] X.-L. Li, M. Anderson, T. Adalı, Second and higher-order correlation analysis of multiple multidimensional variables by joint diagonalization, in: V. Vigneron, V. Zarzoso, E. Moreau, R. Gribonval, E. Vincent (Eds.), Latent Variable Analysis and Signal Separation, Springer Berlin Heidelberg, Berlin, Heidelberg, 2010, pp. 197–204.

[20] M. Congedo, R. Phlypo, D.-T. Pham, Approximate joint singular value decomposition of an asymmetric rectangular matrix set, IEEE Transactions on Signal Processing 59 (1) (2011) 415–424.

[21] J. Antoni, S. Chauhan, A non-hermitian joint diagonalization based blind source separation algorithm for operational modal analysis, in: R. Allemang, J. De Clerck, C. Niezrecki, J. Blough (Eds.), Topics in Modal Analysis I, Volume 5, Springer New York, New York, NY, 2012, pp. 211–221.

[22] J. Miao, G. Cheng, Y. Cai, J. Xia, Approximate joint singular value decomposition algorithm based on givens-like rotation, IEEE Signal Processing Letters 25 (5) (2018) 620–624. doi:10.1109/LSP.2018.2815584.

[23] W. Li, J. Miao, G. Cheng, A jacobi-like algorithm for the general joint diagonalization problem with its application to blind source separation, in: 2019 12th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI), 2019, pp. 1–5. doi:10.1109/CISP-BMEI48845.2019.8965896.

[24] J. Miao, G. Cheng, W. Li, G. Zhang, Non-orthogonal approximate joint diagonalization of non-hermitian matrices in the least-squares sense, Neurocomputing 364 (2019) 63–76. doi:https://doi.org/10.1016/j.neucom.2019.07.022.

[25] J. Miao, G. Cheng, W. Li, E. Moreau, A unitary joint diagonalization algorithm for nonsymmetric higher-order tensors based on givens-like rotations, Numerical Linear Algebra with Applications 27 (3) (2020) e2291. arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/nla.2291, doi:https://doi.org/10.1002/nla.2291.

[26] T. Kim, H. T. Attias, S.-Y. Lee, T.-W. Lee, Blind source separation exploiting higher-order frequency dependencies, IEEE transactions on audio, speech, and language processing 15 (1) (2006) 70–79.

[27] D. Lahat, T. Adali, C. Jutten, Multimodal data fusion: An overview of methods, challenges, and prospects, Proceedings of the IEEE 103 (9) (2015) 1449–1477. doi:10.1109/JPROC.2015.2460697.

[28] A. Cichocki, D. Mandic, L. De Lathauwer, G. Zhou, Q. Zhao, C. Caiafa, H. A. Phan, Tensor decompositions for signal processing applications: From two-way to multiway component analysis, IEEE signal processing magazine 32 (2) (2015) 145–163.

[29] P. Comon, Independent component analysis, a new concept?, Signal Processing 36 (3) (1994) 287–314, higher Order Statistics. doi:https://doi.org/10.1016/0165-1684(94)90029-9.

[30] L. De Lathauwer, B. De Moor, J. Vandewalle, Blind source separation by simultaneous third-order tensor diagonalization, in: 1996 8th European Signal Processing Conference (EUSIPCO 1996), 1996, pp. 1–4.

[31] L. De Lathauwer, A link between the canonical decomposition in multilinear algebra and simultaneous matrix diagonalization, SIAM Jour-

nal on Matrix Analysis and Applications 28 (3) (2006) 642–666. doi:10.1137/040608830.

[32] X. Luciani, L. Albera, Semi-algebraic canonical decomposition of multi-way arrays and joint eigenvalue decomposition, in: 2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2011, pp. 4104–4107. doi:10.1109/ICASSP.2011.5947255.

[33] F. Roemer, M. Haardt, A semi-algebraic framework for approximate cp decompositions via simultaneous matrix diagonalizations (secsi), Signal Processing 93 (9) (2013) 2722–2738. doi:https://doi.org/10.1016/j.sigpro.2013.02.016.

[34] P. Tichavskỳ, A.-H. Phan, A. Cichocki, Non-orthogonal tensor diagonalization, Signal Processing 138 (2017) 313–320.

[35] V. Maurandi, E. Moreau, Non-orthogonal simultaneous diagonalization of k-order complex tensors for source separation, IEEE Signal Processing Letters 24 (11) (2017) 1621–1625. doi:10.1109/LSP.2017.2751038.

[36] F. Bouchard, B. Afsari, J. Malick, M. Congedo, Approximate joint diagonalization with riemannian optimization on the general linear group, SIAM Journal on Matrix Analysis and Applications 41 (1) (2020) 152–170. doi:10.1137/18M1232838.

[37] B. Afsari, Sensitivity analysis for the problem of matrix joint diagonalization, SIAM Journal on Matrix Analysis and Applications 30 (3) (2008) 1148–1171. doi:10.1137/060655997.

[38] C.-H. Lin, J. M. Bioucas-Dias, Nonnegative blind source separation for ill-conditioned mixtures via john ellipsoid, IEEE Transactions on Neural Networks and Learning Systems 32 (5) (2021) 2209–2223. doi:10.1109/TNNLS.2020.3002618.

[39] R. A. Harshman, M. E. Lundy, Parafac: Parallel factor analysis, Computational Statistics & Data Analysis 18 (1) (1994) 39–72. doi:https://doi.org/10.1016/0167-9473(94)90132-5.

[40] N. Vervliet, O. Debals, L. Sorber, M. Van Barel, L. De Lathauwer, Tensorlab 3.0, Available online. (Mar. 2016).
URL `https://www.tensorlab.net`

[41] L. De Lathauwer, B. De Moor, J. Vandewalle, Computation of the canonical decomposition by means of a simultaneous generalized schur decomposition, SIAM Journal on Matrix Analysis and Applications 26 (2) (2004) 295–327. doi:10.1137/S089547980139786X.

[42] L. Zou, X. Chen, Z. J. Wang, Underdetermined joint blind source separation for two datasets based on tensor decomposition, IEEE Signal Processing Letters 23 (5) (2016) 673–677. doi:10.1109/LSP.2016.2546687.

[43] R. Kahankova, R. Martinek, R. Jaros, K. Behbehani, A. Matonia, M. Jezewski, J. A. Behar, A review of signal processing techniques for non-invasive fetal electrocardiography, IEEE Reviews in Biomedical Engineering 13 (2020) 51–73.

[44] L. de Lathauwer, B. de Moor, J. Vandewalle, Fetal electrocardiogram extraction by blind source subspace separation, IEEE Transactions on Biomedical Engineering 47 (5) (2000) 567–572. doi:10.1109/10.841326.

[45] A. J. Bell, T. J. Sejnowski, An information-maximization approach to blind separation and blind deconvolution, Neural Computation 7 (1995) 1129–1159.
URL `https://api.semanticscholar.org/CorpusID:1701422`

[46] M. Niknazar, H. Becker, B. Rivet, C. Jutten, P. Comon, Robust 3-way tensor decomposition and extended state kalman filtering to extract fetal ecg, in: 21st European Signal Processing Conference (EUSIPCO 2013), 2013, pp. 1–5.