Preconditioned optimization algorithms solving the problem of the non unitary joint block diagonalization: application to blind separation of convolutive mixtures

Omar Cherrak, Hicham Ghennioui, Nadège Thirion-Moreau, El Houssein Abarkan

Preconditioned optimization algorithms solving the problem of the non unitary joint block diagonalization: application to blind separation of convolutive mixtures

Omar Cherrak[1,2,3] · **Hicham Ghennioui**[1] · **Nadège Thirion-Moreau**[2,3] · **El Hossain Abarkan**[1]

Abstract This article addresses the problem of the Non Unitary Joint Block Diagonalization ($\mathsf{NU}-\mathsf{JBD}$) of a given set of complex matrices for the blind separation of convolutive mixtures of sources. We propose new different iterative optimization schemes based on Conjugate Gradient, Preconditioned Conjugate Gradient, Levenberg–Marquardt and Quasi-Newton methods. We perform also a study to determine which of these algorithms offer the best compromise between efficiency and convergence speed in the studied context. To be able to derive all these algorithms, a preconditioner has to be computed which requires either the calculation of the complex Hessian matrices or the use of an approximation to these Hessian matrices. Furthermore, the optimal stepsize is also computed algebraically to speed up the convergence of these algorithms. Computer simulations are provided in order to illustrate the behavior of the different algorithms in various contexts: when exactly block-diagonal matrices are considered but also when these matrices are progressively perturbed by an additive Gaussian noise. Finally, it is shown that these algorithms enable solving the blind separation of the convolutive mixtures of sources problem.

Keywords Joint block diagonalization · Blind source separation · Preconditioning · Complex Hessian matrices · Convolutive mixtures · Spatial quadratic time-frequency

✉ Omar Cherrak
omar.cherrak@usmba.ac.ma; omar-cherrak@etud.univ-tln.fr

Hicham Ghennioui
hicham.ghennioui@usmba.ac.ma

Nadège Thirion-Moreau
thirion@univ-tln.fr

El Hossain Abarkan
habarkan@yahoo.fr

1 Faculté des Sciences et Techniques de Fès, LSSC, Université Sidi Mohamed Ben Abdellah, Route Immouzzer, B.P. 2202, Fès, Maroc

2 CNRS, ENSAM, LSIS, UMR 7296, Aix-Marseille Université, 13397 Marseille, France

3 CNRS, LSIS, UMR 7296, Université de Toulon, 83957 La Garde, France

Mathematics Subject Classification 60G35 · 65K10 · 15A69 · 65F08 · 65F10

1 Introduction

With the use of more antennas equipped with more receivers, the volume of available data is continually increasing and advanced signal processing techniques are often required to be able to extract relevant information from this huge amount of data. The problem of the joint decomposition of matrices (or tensors) sets can come within this framework, that is why it has gained a growing attention from the signal processing scientific community over the recent years. Whatever the considered field of application (astronomy, geophysics, remote sensing, biomedical, etc.), algorithms addressing the aforementioned problem are of interest if the problem at hand can finally be formulated as a problem of Array Processing and Direction-Of-Arrival estimation, Blind Source Separation (BSS), blind multidimensional deconvolution or even data mining or analysis.

Because traditionally a pre-whitening stage was used, one of first problem to have been considered was the so-called Joint Diagonalization (JD) of a given matrix set under the unitary constraint. It led to the nowadays well-known JADE (Joint Approximate Diagonalization of Eigenmatrices) Cardoso and Souloumiac 1993 and SOBI (Second Order Blind Identification) Belouchrani et al. 1997 algorithms. The following works have addressed either the problem of the JD of tensors (Comon 1994; Moreau 2001) or the problem of the JD of matrix sets, discarding the unitary constraint (Chabriel and Barrère 2012; Maurandi et al. 2013; Souloumiac 2009; Yeredor 2002). A fairly exhaustive overview of all the suggested approaches is available in Chabriel et al. 2014. This first particular type of matrix decomposition proves useful to solve two kinds of problems i) those of sources localization and direction finding and ii) those of blind separation of instantaneous mixtures of sources.

At the same time, people have started to consider a second type of matrix decompositions - namely joint zero-diagonalization (JZD) -, since it is not uncommon to encounter this problem in different field of applications such as blind source separation (and more specifically spatial quadratic time-frequency domain methods (Belouchrani et al. 2001; Belouchrani et al. 2013)), telecommunications Chabriel and Barrère 2011 and cryptography Walgate et al. 2000. The first proposed algorithms operated under the unitary constraint Belouchrani et al. 2001, since once again, they were applied after a classical pre-whitening stage. But such a preliminary pre-whitening stage establishes a bound with regard to the best performances in the context of BSS that is why the unitary constraint was soon discarded, leading to several solutions among which are (Fadaili et al. 2007; Chabriel et al. 2008).

Recently, a more general problem has been addressed: that of the joint decomposition of a combination of given sets of complex matrices that can follow potentially different decompositions (for example when noncircular complex valued signals are considered and when people want to exploit additional statistical information by using both Hermitian and complex symmetric matrices simultaneously) (Moreau and Adali 2013; Trainini and Moreau 2014; Zeng et al. 2009). Such issues can arise in various signal processing problems, among which are blind identification, separation or multidimensional deconvolution.

In this article, we focus on a fourth type of matrix decompositions, namely the Joint Block-Diagonalization (JBD). It is a rather general problem since the aforementioned JD problem is a special case of this one, but it could be seen as a part of the third problem even if - to our knowledge - only JD problems have been considered in that context up to this point. In such a decomposition, the matrices under consideration assume a specific algebraic structure,

being block diagonal matrices *i.e.* they are block matrices whose diagonal blocks are square matrices of any (possibly even) size while their off-diagonal blocks are all null. First, this problem was addressed considering positive definite and hermitian block-diagonal matrices and unitary joint-block diagonalizers. Jacobi like algorithms were suggested (Belouchrani et al. 1997; De Lathauwer et al. 2002). Then, different alternative methods achieving the same task but for a non unitary joint-block diagonaliser have been proposed (Cherrak et al. 2013b; Ghennioui et al. 2007; Ghennioui et al. 2010; Ghennioui et al. 2007b; Lahat et al. 2012a; Nion 2011; Tichavsky and Koldovsky 2012; Xu et al. 2010; (Zhang et al. (2016); Zhang and Zhang (2016) for video event recognition and object recognition)).

In this article, our goal is to investigate the impact/interest of the preconditioning process in the JBD algorithms to be able to determine which method offers the best compromise between efficiency and convergence speed in the studied context (BSS and blind multidimensional deconvolution). But, first, to be able to derive the different algorithms, the preconditioner has to be determined. It is the reason why, we start with the theoretical calculation of the complex Hessian matrices. Then the different optimization algorithms can be derived and the optimal stepsize can be calculated. Computer simulations are provided in order to illustrate the behavior of the different algorithms in various contexts: when exactly block-diagonal matrices are considered but also when these matrices are progressively perturbed by an additive Gaussian noise. Finally, we show how these algorithms find applications in blind separation of a convolutive mixture of deterministic or random source signals, based on the use of Spatial Quadratic Time Frequency Spectra (SQTFS) or Spatial Quadratic Time Frequency Distributions (SQTFD).

This article is organized as follows: the general problem of NU − JBD and the principle of preconditioning are reminded in Sect. 2. In the Sect. 3, we introduce the algebraic calculations of the Hessian matrices leading to the proposed possibly preconditioned NU − JBD algorithms. In Sect. 4, we show the different classical deterministic iterative optimization schemes considered and we introduce the calculation of the optimal step-size. Computer simulations are provided in Sect. 5 in order to illustrate the behavior of the resulting preconditioned JBD (and possibly JD) algorithms. To that aim, the different algorithms are compared in various contexts comparing them with "state-of the-art approaches". The Sect. 6 enhance the usefulness of these algorithms through one of their possible applications, namely the blind separation of Finite Impulse Response (FIR) convolutive mixtures of non-stationary sources. Computer simulations are performed to illustrate the good performance of the suggested BSS methods. Finally in Sect. 7, a conclusion is drawn.

2 Problem statement and some recalls about optimization

2.1 The non unitary joint block-diagonalization problem and its assumptions

We recall that the problem of the non unitary joint block-diagonalization is stated in the following way Ghennioui et al. 2007. Provided that we have a set $\mathcal{M}$ of N_m square matrices $\mathbf{M}_i \in \mathbb{C}^{M\times M}$, for all $i \in \{1, \ldots, N_m\}$ that all admit the following decomposition:

$$\mathbf{M}_i = \mathbf{A}\mathbf{D}_i\mathbf{A}^H \,, \tag{1}$$

where $(\cdot)^H$ stands for the transpose conjugate (or Hermitian) operator and the N_m square $N \times N$ matrices $\mathbf{D}_i$ are block diagonal matrices, *i.e.* they are all in the form:

$$\mathbf{D}_i = \begin{pmatrix} \mathbf{D}_{i,11} & \mathbf{0}_{12} & \dots & \mathbf{0}_{1r} \\ \mathbf{0}_{21} & \mathbf{D}_{i,22} & \ddots & \vdots \\ \vdots & \ddots & \ddots & \mathbf{0}_{r-1r} \\ \mathbf{0}_{r1} & \dots & \mathbf{0}_{rr-1} & \mathbf{D}_{i,rr} \end{pmatrix}, \tag{2}$$

with r the number of considered blocks ($r \in \mathbb{N}^*$), $\mathbf{D}_{i,jj}$, $i \in \{1, \dots, N_m\}$, $j \in \{1, \dots, r\}$ are $n_j \times n_j$ square matrices so that $n_1 + \cdots + n_r = N$ where $\mathbf{0}_{ij}$ denotes the $(n_i \times n_j)$ null matrix.

$\mathbf{A}$ is a $M \times N$ ($M \geq N$) full rank matrix while $\mathbf{A}^\dagger$ is a pseudo-inverse of $\mathbf{A}$ (or Moore-Penrose generalized matrix inverse). It is a $N \times M$ matrix denoted by $\mathbf{B}$ ($\mathbf{B} = \mathbf{A}^\dagger$). The set of the N_m square matrices $\mathbf{D}_i \in \mathbb{C}^{N \times N}$ is denoted by $\mathcal{D}$. The block sizes n_j for all $j = 1, \dots, r$ are assumed known. In the context of blind source separation, $\mathbf{A}$ is known as the mixing matrix whereas $\mathbf{B}$ is called the separating matrix, N is the number of sources while M is the number of sensors. Many BSS methods go through a joint matrix decomposition stage in order to estimate either the mixing matrix or the separating matrix even if other approaches have been considered too (direct or deflation methods for example). Moreover, this joint matrix decomposition problem often brings us back to a joint diagonalisation (resp. joint block diagonalisation) problem when instantaneous (resp. convolutive) mixtures of sources are considered. An unitary constraint on the matrices $\mathbf{A}$ or $\mathbf{B}$ is the result of a preprocessing stage called spatial whitening of the observations. But studies have proven that this step establishes a bound with regard to the best reachable performances in the context of BSS. That is why recently, there has been a strong interest in methods discarding this constraint.

The general NU − JBD problem consists of estimating the matrix $\mathbf{A}$ and the block-diagonal matrices set $\mathcal{D}$ from only the matrix set $\mathcal{M}$. It was shown in Ghennioui et al. 2010, a standard way consists of minimizing the following quadratic objective function:

$$\mathcal{C}^{(2)}_{\mathsf{JBD}}(\mathbf{B}, \{\widehat{\mathbf{D}}_i\}) = \sum_{i=1}^{N_m} \|\mathsf{OffBdiag}_{(\mathbf{n})}\{\mathbf{B}\mathbf{M}_i\mathbf{B}^H\}\|_F^2 \stackrel{\text{def}}{=} \mathcal{C}_{\mathsf{JBD}}(\mathbf{B}), \tag{3}$$

where $\|\cdot\|_F$ stands for the Frobenius norm and the matrix operator $\mathsf{OffBdiag}_{(\mathbf{n})}\{\cdot\}$ can be defined as (Ghennioui et al. 2010):

$$\mathsf{OffBdiag}_{(\mathbf{n})}\{\mathbf{M}\} = \mathbf{M} - \mathsf{Bdiag}_{(\mathbf{n})}\{\mathbf{M}\} = \begin{pmatrix} \mathbf{0}_{11} & \mathbf{M}_{12} & \dots & \mathbf{M}_{1r} \\ \mathbf{M}_{21} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ \mathbf{M}_{r1} & \mathbf{M}_{r2} & \dots & \mathbf{0}_{rr} \end{pmatrix}, \tag{4}$$

where

$$\mathsf{Bdiag}_{(\mathbf{n})}\{\mathbf{M}\} = \begin{pmatrix} \mathbf{M}_{11} & \mathbf{0}_{12} & \dots & \mathbf{0}_{1r} \\ \mathbf{0}_{21} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ \mathbf{0}_{r1} & \mathbf{0}_{r2} & \dots & \mathbf{M}_{rr} \end{pmatrix}. \tag{5}$$

2.2 Some recalls about optimization and preconditioning

In the literature, one can find many unconstrained optimization methods and some of them, namely gradient-descent, steepest descent, relative gradient and conjugate gradient methods

have already been studied and applied in this specific context of JBD (Ghennioui et al. 2010; Nion 2011). They are known as first-order methods since they only depend on the gradient of the cost function and not on its second derivatives (*i.e.* the Hessian matrix denoted here by $\mathbf{H}_\bullet$. Notice that in the next section we will clarify this notation). To our knowledge, very few, if any, studies have been led on preconditioned optimization methods in such a framework. Yet, when one is designing an algorithm for a specific application, many conflicting requirements have to be taken into account (convergence rate (as few iterations as possible), computation time per iteration (as few floating point operations as possible), stability, sensitivity to numerical errors, robustness versus noise and/or model errors, problems of initialization, storage requirements, potential parallelization, ease of implementation, real time implementation, just to name a few), and at the end, for one given application, one must choose the algorithm which offers the best compromise between all those requirements. That is why we have chosen, here, to look specifically at preconditioned optimization methods to determine whether they are interesting or not for joint block diagonalization problems.

The main purposes for which preconditioning is generally used, are:

- to further accelerate the convergence since gradient-descent or steepest descent algorithms are well known for their sometimes slow convergence,
- to increase stability and robustness versus noise or model errors,
- to cope with the fact that variables have wildly different magnitudes (since the most simple form of preconditioning is a scaling of the variables (thanks to a diagonal preconditioner) with well chosen coefficients),
- to cope with the fact that the cost function can rapidly change in some directions and slowly in other ones,
- to be able to tackle ill-posed problems,
- to ensure certain constraints such as non-negativity (see for example the nonnegative matrix factorization algorithm (NMF) based on multiplicative updates suggested by Lee and Seung Lee and Seung 2000b which can be interpreted as a preconditioned gradient algorithm).

Sometimes the preconditioning process just speed up the convergence, but there also exist some difficult cases which cannot be solved without good preconditioning. That is why several works have addressed the problem of the design of effective preconditioners. First, the preconditioning has been used in direct methods and, latter this concept has been extended to the case of iterative processes (Chebyschev acceleration) in (Evans 1968; Turing 1948). The key issue remains obviously the choice of the preconditioner denoted by $\mathbf{P}$. In the literature, one can find many works on different types of preconditioners, among which are the Jacobi and Gauss-Seidel preconditioners Westlake 1968. The first one consists of using the diagonal of the desired matrix while the second one consists of decomposing the preconditioner matrix into a lower triangular matrix, a diagonal matrix, and an upper triangular matrix. Yet, for non-symmetric problems, such preconditioners are limited in their effectiveness and robustness Axelsson 1985.

The analysis of the convergence rate of the proposed preconditioned gradient-descent method provides a substantial help in the choice of $\mathbf{P}$ and directions to follow. If the cost function is twice differentiable in the neighborhood of a local minimizer, *i.e.* with Hessian matrix $\mathbf{H}_\bullet \overset{\text{def}}{=} \nabla^2_{\mathbf{B}} \mathcal{C}_{\text{JBD}} \overset{\text{def}}{=} \frac{\partial^2 \mathcal{C}_{\text{JBD}}(\mathbf{B},\mathbf{B}^*)}{\partial \mathbf{B} \partial \mathbf{B}^*}$ then the fastest convergence is obtained when the preconditioner $\mathbf{P}$ minimizes the condition number, namely κ, of the product $\mathbf{PH}_\bullet$. The condition number stands for the ratio of the largest singular value of $\mathbf{PH}_\bullet$ divided by the smallest singular value of this matrix *i.e.* $\kappa \overset{\text{def}}{=} \frac{\lambda_{\max}(\mathbf{PH}_\bullet)}{\lambda_{\min}(\mathbf{PH}_\bullet)}$. For quadratic cost functions, the ideal pre-

conditioner would be $\mathbf{P} = (\mathbf{H}_\bullet)^{-1}$ so that $\mathbf{P}\mathbf{H}_\bullet = \mathbf{I}$ since the identity matrix $\mathbf{I}$ possesses the minimal (unity) condition number. But since it is not always easy to calculate or to compute $(\mathbf{H}_\bullet)^{-1}$ (especially when N or M become large in our case), it remains interesting to develop preconditioners that approximate $(\mathbf{H}_\bullet)^{-1}$ and that are cheaper to construct and to apply (Chen 2005a; Benzi 2002).

To be able to design a suitable preconditioning matrix $\mathbf{P}$ in the context of JBD, we will start with the calculation of the four exact complex Hessian matrices. Then, we will be able to derive several new preconditioned algorithms among which is a preconditioned Conjugate Gradient algorithm. The conjugate gradient method is an iterative optimization method which has become really popular in nonlinear optimization (Hager and Zhang 2006; Paatero 1999) and its efficiency may be improved by a proper choice of the preconditioning matrix Axelsson 1985.

3 New preconditioned joint block-diagonalization algorithms

To estimate the complex matrix $\mathbf{B} \in \mathbb{C}^{N\times M}$, the cost function given in (3) has to be minimized. To that aim, the differential $d\mathcal{C}_{\mathsf{JBD}}$ of $\mathcal{C}_{\mathsf{JBD}}$ has to be derived, involving the calculation of the partial derivatives $\frac{\partial}{\partial \cdot}$ of the objective function $\mathcal{C}_{\mathsf{JBD}}$ with respect to $\mathbf{B}$ and $\mathbf{B}^*$ (it is the reason why, from now on, the objective function is denoted by $\mathcal{C}_{\mathsf{JBD}}(\mathbf{B}, \mathbf{B}^*)$ where $\mathbf{B}^*$ stands for the complex conjugate of the complex matrix $\mathbf{B}$, except in 4.4). Then, the complex gradient matrix can be derived. In fact, the $N \times M$ complex gradient matrix of our real-valued scalar cost function given in (3) denoted $\nabla_{\mathbf{B}}\mathcal{C}_{\mathsf{JBD}}$ has been calculated in Ghennioui et al. 2010 and found to be equal to:

$$\frac{\partial \mathcal{C}_{\mathsf{JBD}}(\mathbf{B}, \mathbf{B}^*)}{\partial \mathbf{B}^*} = \sum_{i=1}^{N_m} \left[\mathsf{OffBdiag}_{(\mathbf{n})}\{\mathbf{B}\mathbf{M}_i\mathbf{B}^H\}\mathbf{B}\mathbf{M}_i^H + \left(\mathsf{OffBdiag}_{(\mathbf{n})}\{\mathbf{B}\mathbf{M}_i\mathbf{B}^H\}\right)^H \mathbf{B}\mathbf{M}_i \right]. \tag{6}$$

Eventually, the second order differential $d^2\mathcal{C}_{\mathsf{JBD}}$ can be calculated too in order to derive the complex Hessian matrices. The complex matrix $\mathbf{H}$ is a square $2NM \times 2NM$ matrix constituted of four $NM \times NM$ square blocks which are the Hessian matrices:

$$\mathbf{H} = \begin{bmatrix} \mathbf{H}_{\mathbf{B},\mathbf{B}^*} & \mathbf{H}_{\mathbf{B}^*,\mathbf{B}^*} \\ \mathbf{H}_{\mathbf{B},\mathbf{B}} & \mathbf{H}_{\mathbf{B}^*,\mathbf{B}} \end{bmatrix}. \tag{7}$$

To calculate the aforementioned matrix $\mathbf{H}$, we follow the same path as in Hjorungnes, A. 2011 (details of this calculation can be found in the Appendix). In short, we have to derive the expression of the second-order complex differential of $\mathcal{C}_{\mathsf{JBD}}(\mathbf{B}, \mathbf{B}^*)$ which is then written as:

$$d^2\mathcal{C}_{\mathsf{JBD}}(\mathbf{B}, \mathbf{B}^*) = \begin{bmatrix} d\mathsf{vec}^T(\mathbf{B}^*) & d\mathsf{vec}^T(\mathbf{B}) \end{bmatrix} \begin{bmatrix} \mathbf{A}_{00} & \mathbf{A}_{01} \\ \mathbf{A}_{10} & \mathbf{A}_{11} \end{bmatrix} \begin{bmatrix} d\mathsf{vec}(\mathbf{B}) \\ d\mathsf{vec}(\mathbf{B}^*) \end{bmatrix}. \tag{8}$$

As shown in Hjorungnes, A. 2011, the four $NM \times NM$ complex matrices $\mathbf{A}_{00}$, $\mathbf{A}_{01}$, $\mathbf{A}_{10}$ and $\mathbf{A}_{11}$ involved in the above expression are linked to the Hessian matrices by:

$$\mathbf{H}_{\mathbf{B},\mathbf{B}^*}\left(\mathcal{C}_{\mathsf{JBD}}(\mathbf{B}, \mathbf{B}^*)\right) = \frac{\mathbf{A}_{00} + \mathbf{A}_{11}^T}{2} = \left(\mathbf{H}_{\mathbf{B}^*,\mathbf{B}}\left(\mathcal{C}_{\mathsf{JBD}}(\mathbf{B}, \mathbf{B}^*)\right)\right)^T, \tag{9}$$

$$\mathbf{H}_{\mathbf{B}^*,\mathbf{B}^*}\left(\mathcal{C}_{\mathsf{JBD}}(\mathbf{B}, \mathbf{B}^*)\right) = \frac{\mathbf{A}_{01} + \mathbf{A}_{01}^T}{2}, \tag{10}$$

$$\mathbf{H}_{\mathbf{B},\mathbf{B}}\left(\mathcal{C}_{\mathsf{JBD}}(\mathbf{B},\mathbf{B}^*)\right)=\frac{\mathbf{A}_{10}+\mathbf{A}_{10}^T}{2}. \tag{11}$$

There are shown to be equal to (see the Appendix 1 for the detailed mathematic developments):

$$\begin{aligned}\mathbf{A}_{00}=&\left(\mathbf{M}_i^T\mathbf{B}^T\otimes\mathbf{I}_N\right)\mathbf{T}_{\text{Boff}}^T\left(\mathbf{B}^*\mathbf{M}_i^*\otimes\mathbf{I}_N\right)+\left(\mathbf{M}_i^*\mathbf{B}^T\otimes\mathbf{I}_N\right)\mathbf{T}_{\text{Boff}}^T\left(\mathbf{B}^*\mathbf{M}_i^T\otimes\mathbf{I}_N\right)\\&+\mathbf{M}_i^*\otimes\mathsf{OffBdiag}_{(\mathbf{n})}\{\mathbf{B}\mathbf{M}_i\mathbf{B}^H\}+\mathbf{M}_i^T\otimes\mathsf{OffBdiag}_{(\mathbf{n})}\{\mathbf{B}\mathbf{M}_i^H\mathbf{B}^H\}=\mathbf{A}_{11}^*,\end{aligned} \tag{12}$$

$$\begin{aligned}\mathbf{A}_{10}=&\mathbf{K}_{N,M}^T\left(\mathbf{I}_N\otimes\mathbf{M}_i\mathbf{B}^H\right)\mathbf{T}_{\text{Boff}}^T\left(\mathbf{B}^*\mathbf{M}_i^*\otimes\mathbf{I}_N\right)\\&+\mathbf{K}_{N,M}^T\left(\mathbf{I}_N\otimes\mathbf{M}_i^H\mathbf{B}^H\right)\mathbf{T}_{\text{Boff}}^T\left(\mathbf{B}^*\mathbf{M}_i^T\otimes\mathbf{I}_N\right)=\mathbf{A}_{01}^*,\end{aligned} \tag{13}$$

where the operator $\otimes$ denotes the Kronecker product, $\mathbf{K}_{N,M}$ is a square commutation matrix of size $NM\times NM$ and $\mathbf{T}_{\text{Boff}}=\mathbf{I}_{N^2}-\mathbf{T}_{\text{Diag}}$, is the $N^2\times N^2$ "transformation" matrix, with $\mathbf{T}_{\text{Diag}}=\mathsf{diag}\{\mathsf{vec}(\mathsf{BDiag}\{\mathbf{1}_N\})\}$, $\mathbf{1}_N$ is the $N\times N$ matrix whose components are all ones, $\mathsf{diag}\{\mathbf{a}\}$ is a square diagonal matrix whose diagonal elements are the elements of the vector $\mathbf{a}$, $\mathbf{I}_{N^2}=\mathsf{Diag}\{\mathbf{1}_{N^2}\}$ is the $N^2\times N^2$ identity matrix, and $\mathsf{Diag}\{\mathbf{A}\}$ is the square diagonal matrix with the same diagonal elements as $\mathbf{A}$.

One can check that the bigger complex matrix $\mathbf{H}$ defined in (7) is hermitian and that two parts can be distinguished in this matrix $\mathbf{H}$: its block-diagonal part and its off-block diagonal part.

According to Shewchuk 1994b, it is advisable to consider only the diagonal (real part) of the Hessian matrix *i.e.* (9) and (12) to ensure a non increase of the cost function, and to easily obtain its inverse. The $NM\times NM$ preconditioning matrix, denoted by $\mathbf{P}$, that we suggest thus, reads:

$$\mathbf{P}=(\widehat{\mathbf{H}})^{-1}, \tag{14}$$

$$\widehat{\mathbf{H}}=\mathsf{Diag}\left\{\mathbf{H}_{\mathbf{B},\mathbf{B}^*}\left(\mathcal{C}_{\mathsf{JBD}}(\mathbf{B},\mathbf{B}^*)\right)\right\}=\mathsf{Diag}\left\{\mathbf{H}_{\mathbf{B}^*,\mathbf{B}}\left(\mathcal{C}_{\mathsf{JBD}}(\mathbf{B},\mathbf{B}^*)\right)\right\}. \tag{15}$$

Hence in the next sections, we can focus on the different possibly preconditioned algorithms that can be derived, and try to better assess their behavior in different contexts.

4 Classical deterministic iterative optimization schemes

To estimate the joint block diagonalizer $\mathbf{B}$, the cost function given in (3) has to be minimized. To that aim different deterministic iterative optimization schemes can be considered. It means that $\mathbf{B}$ has to be re-estimated at each iteration m, so from now on, it will be denoted by $\mathbf{B}^{(m)}$. This matrix can also be stored in a vector $\mathbf{b}^{(m)}$ if the $\mathsf{vec}(\cdot)$ operator is applied *i.e.* $\mathbf{b}^{(m)}=\mathsf{vec}\left(\mathbf{B}^{(m)}\right)$. Depending on the considered optimization scheme, $\mathbf{B}^{(m)}$ for all m is updated according to one of the following adaptation rules. For example, when a classical steepest descent approach is considered (gradient approach if the step size is a fixed positive scalar *i.e.* $\mu^{(m)}=\mu$ for all m), $\mathbf{B}$ is updated at each iteration m ($m=1,2,\ldots$) according to:

$$\mathbf{B}^{(m+1)}=\mathbf{B}^{(m)}-\mu^{(m)}\nabla_{\mathbf{B}}\mathcal{C}_{\mathsf{JBD}}(\mathbf{B}^{(m)},\mathbf{B}^{*(m)}), \tag{16}$$

or equivalently,

$$\mathbf{b}^{(m+1)}=\mathbf{b}^{(m)}-\mu^{(m)}\mathbf{g}_{\mathbf{B}^*}^{(m)}, \tag{17}$$

where $\nabla_{\mathbf{B}}\mathcal{C}_{\mathsf{JBD}}(\mathbf{B}^{(m)},\mathbf{B}^{*(m)})$ is the gradient matrix given in (6) or $\mathbf{g}_{\mathbf{B}^*}^{(m)}$ is its vectorized form *i.e.* $\mathbf{g}_{\mathbf{B}^*}^{(m)}=\left(\mathcal{D}_{\mathbf{B}^*}(\mathcal{C}_{\mathsf{JBD}}(\mathbf{B}^{(m)},\mathbf{B}^{*(m)}))\right)^T$ and $\mu^{(m)}$ is a real positive constant called the step

size (the problem of its choice is treated in the Sect. 4.4). This algorithm may converge slowly, that is why we are interested in preconditioned algorithms in order to speed up the convergence.

4.1 Preconditioned non linear conjugate gradient

To further accelerate convergence, conjugate gradient methods modify the search directions to ensure that they are approximately mutually conjugate *i.e.* orthogonal with respect to an inner product related to the Hessian of the cost function. Thus, when the preconditioned conjugate gradient algorithm is considered, the search direction, $\mathbf{d}$, is given at the first iteration by:

$$\mathbf{d}^{(1)} = -\mathbf{P}^{(1)}\mathbf{g}_{\mathbf{B}^*}^{(1)}. \tag{18}$$

Then, $\forall m = 2, \ldots$, the following adaptation rule is used:

$$\begin{cases} \mathbf{b}^{(m+1)} = \mathbf{b}^{(m)} + \mu^{(m)}\mathbf{d}^{(m)}, \\ \mathbf{d}^{(m+1)} = -\mathbf{P}^{(m+1)}\mathbf{g}_{\mathbf{B}^*}^{(m+1)} + \beta^{(m)}\mathbf{d}^{(m)}, \end{cases} \tag{19}$$

where the expression of the preconditioner $\mathbf{P}$ is given in (14)-(15). It should be a positive-definite matrix, so non positive diagonal elements are prohibited: if it occurs, the algorithm is no more preconditioned ($\mathbf{P}$ is then set to the identity matrix $\mathbf{I}_{NM}$ and we come back to the classical non linear conjugate gradient algorithm). In exact line search method, we use the Polak-Ribière (β_{PR}) formula Polak 1997a which is given by:

$$\beta_{\mathsf{PR}}^{(m+1)} = \frac{(\mathbf{g}_{\mathbf{B}^*}^{(m+1)} - \mathbf{g}_{\mathbf{B}^*}^{(m)})^H \mathbf{P}^{(m+1)}\mathbf{g}_{\mathbf{B}^*}^{(m+1)}}{(\mathbf{g}_{\mathbf{B}^*}^{(m)})^H \mathbf{P}^{(m)}\mathbf{g}_{\mathbf{B}^*}^{(m)}}. \tag{20}$$

However, several expressions for β are classically used: the Fletcher-Reeves (β_{FR}) Shewchuk 1994b and the Dai-Yuan (β_{DY}) Yuan 1999. The resulting algorithm is denoted by $\mathsf{JBD}_{\mathsf{PCG}_{\mathsf{EH}}}$ (EH stands for Exact Hessian).

4.2 Levenberg–Marquardt algorithm

When the Hessian matrix $\widehat{\mathbf{H}}$ tends to lose its positive-definiteness property through the iterations, and hence may fail to construct descent directions, it is better to stabilize it using trust region techniques that modify $\widehat{\mathbf{H}}$ by adding a multiple of the identity matrix. It is the principle of the Levenberg–Marquardt approach Luenberger 1969a:

$$\mathbf{b}^{(m+1)} = \mathbf{b}^{(m)} - \mu^{(m)} \left(\widehat{\mathbf{H}}^{(m)} + \alpha \mathbf{I}_{\mathbf{NM}}\right)^{-1} \mathbf{g}_{\mathbf{B}^*}^{(m)}, \tag{21}$$

where α is a relaxation coefficient. We notice that by setting $\widehat{\mathbf{H}} = \mathbf{I}_{NM}$ in (21) (or by considering that α is chosen high enough), the gradient algorithm in (17) may be obtained. It can also be seen as a variant of the preconditioned conjugate gradient algorithm given in (19) where $\beta = 0$ and where the preconditioner is not directly the inverse of the Hessian but is slightly modified. With regards to $\widehat{\mathbf{H}}$, we can use either our solution given in (14)-(15).

The resulting algorithm is denoted by $\mathsf{JBD}_{\mathsf{LM}_{\mathsf{EH}}}$ (again EH stands for Exact Hessian).

4.3 Quasi-Newton algorithm

In (21), by setting $\alpha = 0$ and considering that the preconditioner $\mathbf{P}$ is a $NM \times NM$ calculation of the inverse of the Hessian matrix given by (14) and (15), we obtain the Quasi-Newton

adaptation rule. In this case, the algorithm is initialized using for $\mathbf{P}^{(1)}$ (or $\widehat{\mathbf{H}}^{(1)}$), a symmetric, $NM \times NM$ positive-definite matrix. The resulting algorithm is denoted by $\mathsf{JBD}_{\mathsf{QN}_{\mathsf{EH}}}$ (QN standing for Quasi-Newton).

4.4 Choice of the step size: Enhanced Line Search or seek of the optimal step-size

It remains essential to find a good stepsize μ in the given search direction $\mathbf{d}^{(m)}$ at each iteration m. This stepsize $\mu^{(m)}$ can be found in different ways: it can be fixed or decreased versus the iterations, or we can opt for a global search with Enhanced Line Search (ELS) or for an approximation by a line search method like backtracking Vandenberghe 2004 (which is a locally optimal stepsize method).

This subsection is dedicated to the calculation of the optimal stepsize which is denoted by μ_{opt}. When the preconditioned conjugate gradient algorithm is considered for example, it implies the algebraical calculation of the following quantity: $\mathcal{C}_{\mathsf{JBD}}(\mathbf{B}^{(m+1)}) = \mathcal{C}_{\mathsf{JBD}}\left(\mathbf{B}^{(m)} - \mu\mathbf{D}^{(m)}\right)$, where $\mathbf{D}^{(m)} = \mathsf{unvec}\left(\mathbf{d}^{(m)}\right)$ or equivalently $\mathbf{d}^{(m)} = \mathsf{vec}\left(\mathbf{D}^{(m)}\right)$ where the vector $\mathbf{d}^{(m)}$ is given in (19) (operator unvec enables to come back to a matrix). It is a 4th-degree polynomial whose expression is given by (to simplify the different expressions, the dependency upon the iteration m is omitted):

$$\mathcal{C}_{\mathsf{JBD}}\left(\mathbf{B} - \mu\mathbf{D}\right) = a_0 + a_1\mu + a_2\mu^2 + a_3\mu^3 + a_4\mu^4, \tag{22}$$

where the five coefficients a_0, a_1, a_2, a_3 and a_4 are found equal to (see the Appendix 1 for the details mathematic developments),

$$a_0 = \sum_{i=1}^{N_m} \mathsf{tr}\left\{\mathbf{C}_0^H\, \mathsf{OffBdiag}_{(\mathbf{n})}\left\{\mathbf{C}_0\right\}\right\}, \tag{23}$$

$$a_1 = -\sum_{i=1}^{N_m} \mathsf{tr}\left\{\mathbf{C}_1^H\, \mathsf{OffBdiag}_{(\mathbf{n})}\left\{\mathbf{C}_0\right\} + \mathbf{C}_0^H\, \mathsf{OffBdiag}_{(\mathbf{n})}\left\{\mathbf{C}_1\right\}\right\}, \tag{24}$$

$$a_2 = \sum_{i=1}^{N_m} \mathsf{tr}\left\{\mathbf{C}_2^H\, \mathsf{OffBdiag}_{(\mathbf{n})}\left\{\mathbf{C}_0\right\} + \mathbf{C}_1^H\, \mathsf{OffBdiag}_{(\mathbf{n})}\left\{\mathbf{C}_1\right\} + \mathbf{C}_0^H\, \mathsf{OffBdiag}_{(\mathbf{n})}\left\{\mathbf{C}_2\right\}\right\}, \tag{25}$$

$$a_3 = -\sum_{i=1}^{N_m} \mathsf{tr}\left\{\mathbf{C}_2^H\, \mathsf{OffBdiag}_{(\mathbf{n})}\left\{\mathbf{C}_1\right\} + \mathbf{C}_1^H\, \mathsf{OffBdiag}_{(\mathbf{n})}\left\{\mathbf{C}_2\right\}\right\}, \tag{26}$$

$$a_4 = \sum_{i=1}^{N_m} \mathsf{tr}\left\{\mathbf{C}_2^H\, \mathsf{OffBdiag}_{(\mathbf{n})}\left\{\mathbf{C}_2\right\}\right\}, \tag{27}$$

with

$$\mathbf{C_0} = \mathbf{B}\mathbf{M}_i\mathbf{B}^H, \tag{28}$$

$$\mathbf{C_1} = \mathbf{B}\mathbf{M}_i\mathbf{D}^H + \mathbf{D}\mathbf{M}_i\mathbf{B}^H, \tag{29}$$

$$\mathbf{C_2} = \mathbf{D}\mathbf{M}_i\mathbf{D}^H. \tag{30}$$

Then, the third order polynomial (31) corresponding to the derivative with respect to μ of the 4th-degree polynomial given in (22) is calculated and its three roots are derived:

$$\frac{\partial \mathcal{C}_{\mathsf{JBD}}\left(\mathbf{B} - \mu\mathbf{D}\right)}{\partial \mu} = 4a_4\mu^3 + 3a_3\mu^2 + 2a_2\mu + a_1. \tag{31}$$

Finally the optimal stepsize, μ_{opt}, corresponds to the root that leads to the smallest value of (22).

4.5 Algorithms proposed

The non-unitary JBD proposed algorithms are summarized below:

Data: N_m square matrices $\mathbf{M}_1, \mathbf{M}_2, \ldots, \mathbf{M}_{N_m}$, stopping criterion ϵ, step-size μ, max. number of iterations N_{max}, number used for restart N_0, relaxation coefficient α
Result: Estimation of the joint block diagonalizer $\mathbf{B}$
initialize: $\mathbf{B}^{(0)}$; $\mathbf{D}^{(0)}$; $m = 0$;
repeat
  if *(For* (P)CG *algorithms only)* $m \mod N_0 = 0$ **then**
    restart
  else
    Calculate optimal step-size $\mu_{\text{opt}}^{(m)}$ (root of (31) leading to the smallest value of (22))
    Compute gradient matrix $\nabla_{\mathbf{B}}\mathcal{C}_{\text{JBD}}(\mathbf{B}^{(m+1)}, (\mathbf{B}^{(m+1)})^*)$ given in (6)
    Compute exact Hessian matrix $\mathbf{H}$ given in (15)
    ——— JBD_{CG} **algorithm** ———
    Compute matrix $\mathbf{B}^{(m+1)}$ thanks to (16) or (17)
    ——— $\text{JBD}_{\text{PCG}_{\text{EH}}}$ **algorithm** ———
    Compute exact preconditioner $\mathbf{P}^{(m+1)}$ given in (14)
    Compute matrix $\mathbf{B}^{(m+1)}$ thanks to (19)
    Compute $\beta_{\text{PR}}^{(m)}$ given by (20)
    Compute the search direction $\mathbf{D}^{(m+1)}$ thanks to (19)
    ——— $\text{JBD}_{\text{LM}_{\text{EH}}}$ **algorithm** ———
    Compute matrix $\mathbf{B}^{(m+1)}$ thanks to (21)
    Calculate the error $e^{(m+1)} = \frac{1}{N_m}\mathcal{C}_{\text{JBD}}(\mathbf{B}^{(m+1)})$
    if $e^{(m+1)} > e^{(m+1)}$ **then**
      $\alpha = \frac{\alpha}{10}$, $e^{(m+1)} = e^{(m)}$
    else
      $\alpha = 10\alpha$
    end
    ——— $\text{JBD}_{\text{QN}_{\text{EH}}}$ **algorithm** ———
    Compute exact preconditioner $\mathbf{P}^{(m+1)}$ given in (14)
    Compute matrix $\mathbf{B}^{(m+1)}$ thanks to (21) with $\alpha = 0$
    ———
    $m = m + 1$;
  end
until *(*$(\|\mathbf{B}^{(m+1)} - \mathbf{B}^{(m)}\|_F^2 \leq \epsilon)$ *or* $(m \geq N_{max})$*)*;

4.6 Complexity of algorithms

To compute the gradient matrix $\nabla_{\mathbf{B}}\mathcal{C}_{\text{JBD}}\left(\mathbf{B}^{(m+1)}, (\mathbf{B}^{(m+1)})^*\right)$ whose expression is given in (6), the algorithmic complexity C_1 is:

$M \neq N$	$4N_m NM(M+N) + 2N^2 N_m$
$M \gg N$	$4N_m NM^2$
$M = N$	$8N_m N^3 + 2N_m N^2$

To compute the preconditioning matrix according to $\mathbf{P} = \text{Diag}\left\{\mathbf{H}_{\mathbf{B},\mathbf{B}^*}\left(\mathcal{C}_{\text{JBD}}(\mathbf{B}, \mathbf{B}^*)\right)\right\}$, the complexity C_2 is:

$M \neq N$	$2N_m(MN(4M+3N)+MN^4(M+N))$
$M \gg N$	$8N_m M^2 N + 2N_m M^2 N^4$
$M = N$	$8N_m N^3 + 2N^6$

To update the search direction given in (19), using the Polak-Ribière formula $\beta^{(m)}$ given in (20), the complexity is:

$M \neq N$	$2M^3N^3 + 4N_m(MN(4M+3N)+2MN^4(M+N)) + 2NM(M+N)$
$M \gg N$	$2M^3N^3 + 16N_m M^2 N + 8N_m M^2 N^4 + 2NM^2$
$M = N$	$8N^6 + 28N_m N^3 + 4N^4$

To compute the coefficients $a_0^{(m)}, \ldots, a_4^{(m)}$ thanks to (23)–(27), and then to obtain the optimal step-size $\mu_{\text{opt}}^{(m)}$ by the search of the root of the polynomial given in (31) attaining the minimum in the polynomial given in (22), the complexity is:

$M \neq N$	$24MNN_m(M+N) + 9N^2N_m(1+N)$
$M \gg N$	$24M^2NN_m$
$M = N$	$57N^3N_m + 9N^2N_m$

Finally, the total complexity for the different algorithms that we were able to derive is summed up in the following table:

Algorithm	Algorithmic cost	
JBD_{CG}	$M \neq N$	$28N_mNM(M+N) + 11N^2N_m + 2N^3M^3 + NM(M+N) + 9N^3N_m$
	$M \gg N$	$11N^2N_m + 28N_mNM^2 + 2N^3M^3 + 9N^3N_m + NM^2$
	$M = N$	$17N_mN^3 + 11N^2N_m + 2N^6 + 2N^3 + 48N_mN^3$
$\text{JBD}_{\text{PCG}_{\text{EH}}}$	$M \neq N$	$2N_mNM(20M+17N+5N^3(M+N)) + 2N^3M^3 + 19N^2N_m + 9N^3N_m$
	$M \gg N$	$40N_mNM^2 + 10M^2N^4 + 2N^3M^3 + 19N^2N_m + 9N^3N_m$
	$M = N$	$83N_mN^3 + 20N^4 + 2N^6 + 19N^2N_m$
$\text{JBD}_{\text{LM}_{\text{EH}}}$	$M \neq N$	$2N_mNM(18M+17N+N^3(M+N)) + 11N^2N_m + 2N^3M^3 + 9N^3N_m$
	$M \gg N$	$36N_mM^2N + 11N^2N_m + 2N_mM^2N^4 + 2N^3M^3 + 9N^3N_m$
	$M = N$	$88N_mN^3 + 4N_mN^6 + 2N^6 + 2N^2N_m$
$\text{JBD}_{\text{QN}_{\text{EH}}}$	$M \neq N$	$2N_mNM(18M+17N) + 11N^2N_m + 2N_mMN^4(M+N) + 9N^3N_m$
	$M \gg N$	$36N_mNM^2 + 11N^2N_m + 2N_mM^2N^4 + 9N^3N_m$
	$M = N$	$69N_mN^3 + 11N^2N_m + 4N_mN^6$

Finally, notice that the global complexity of the different algorithms has to be multiplied by the total number of iterations N_i needed to reach the convergence. In the case of practical applications, the computational time necessary to build the set of the N_m matrices should be counted too.

5 Computer simulations and comparisons

In this section, we present simulations to illustrate the behavior of the suggested preconditioned JBD algorithms ($\mathsf{JBD}_{\mathsf{PCG}_{\mathsf{EH}}}$, $\mathsf{JBD}_{\mathsf{LM}_{\mathsf{EH}}}$ and $\mathsf{JBD}_{\mathsf{QN}_{\mathsf{EH}}}$ but also of the $\mathsf{JBD}_{\mathsf{CG}}$). All these algorithms are compared with the conjugate gradient $\mathsf{JBD}_{\mathsf{CG}_{\mathsf{NION}}}$ proposed in Nion 2011 and other algorithms proposed in Ghennioui et al. 2010 and Ghennioui et al. 2008a: The first one is based on a relative gradient approach denoted $\mathsf{JBD}_{\mathsf{RGrad}}$ and the second one is based on a (absolute) gradient approach denoted $\mathsf{JBD}_{\mathsf{AbsGrad}}$. To that aim, we build a set $\mathcal{D}$ of N_m complex block-diagonal matrices $\mathbf{D}_i$ (for all $i = 1, \ldots, N_m$), randomly drawn according to a Gaussian law with zero mean and unit variance. Initially, these matrices are considered as exactly block-diagonal, then, a random noise matrix of mean 0 and variance σ_N^2 is added on the off-diagonal blocks of the matrices $\mathbf{D}_i$, for all $i = 1, \ldots, N_m$. The signal to noise ratio is defined as $\mathsf{SNR} = 10\log(\frac{1}{\sigma_N^2})$.

To better assess the quality of the estimation and to be able to compare the different algorithms, an error index is required. We are using the one, $I(\mathbf{G})$ ($\mathbf{G} = \hat{\mathbf{B}}\mathbf{A}$), that we introduced in Ghennioui et al. 2007 and defined as :

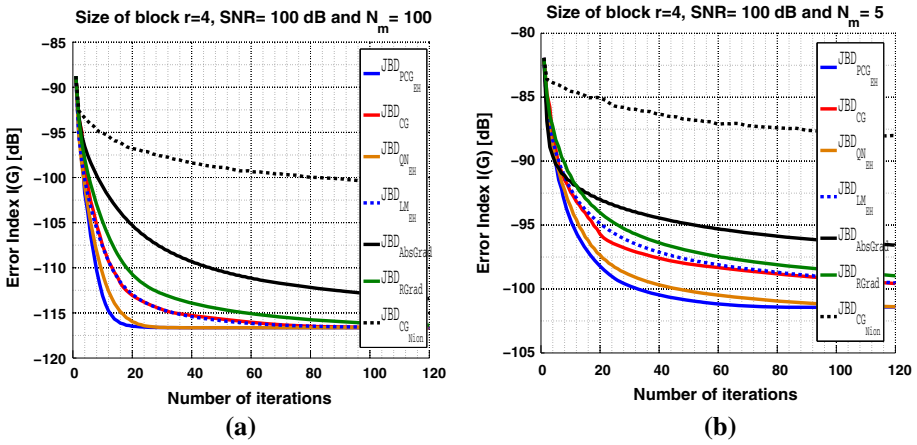
$$\frac{1}{r(r-1)}\left[\sum_{i=1}^{r}\left(\sum_{j=1}^{r}\frac{\|(\mathbf{G})_{i,j}\|_F^2}{\max\limits_{\ell}\|(\mathbf{G})_{i,\ell}\|_F^2}-1\right)+\sum_{j=1}^{r}\left(\sum_{i=1}^{r}\frac{\|(\mathbf{G})_{i,j}\|_F^2}{\max\limits_{\ell}\|(\mathbf{G})_{\ell,j}\|_F^2}-1\right)\right],$$

where $(\mathbf{G})_{i,j}$ for all $i, j \in \{1, \ldots, r\}$ is the (i, j)-th (square) block matrix of $\mathbf{G} = \hat{\mathbf{B}}\mathbf{A}$. This error index will be used again in the BSS application (Sect. 6). The best results are obtained when the error index $I(\cdot)$ is found to be close to 0 in linear scale ($-\infty$ in logarithmic scale). Regarding to the charts, $I(\cdot)$ is given in dB and is then defined by $I(\cdot)$ dB $= 10\log(I(\cdot))$. The matrix $\mathbf{A}$ has been randomly chosen in all simulations. Moreover, all displayed results have been averaged over 30 Monte-Carlo trials.

We consider $M = N = 8$ ($\mathbf{A}$ is square) and $r = 4$ (with $r = 4$ for all $j = 1, \ldots, 4$). The set $\mathcal{M}$ of N_m square matrices $\mathbf{M}_i$, $i = 1, \ldots, N_m$ consists of $N_m = 100$ (resp. $N_m = 5$) $M \times M$ matrices. All algorithms are initialized using the same initialization provided by the generalized eigenvalue decomposition (GEVD) as suggested in Nion 2011. Notice that the maximal number of iterations allowed is set at N_{max} =120 iterations.

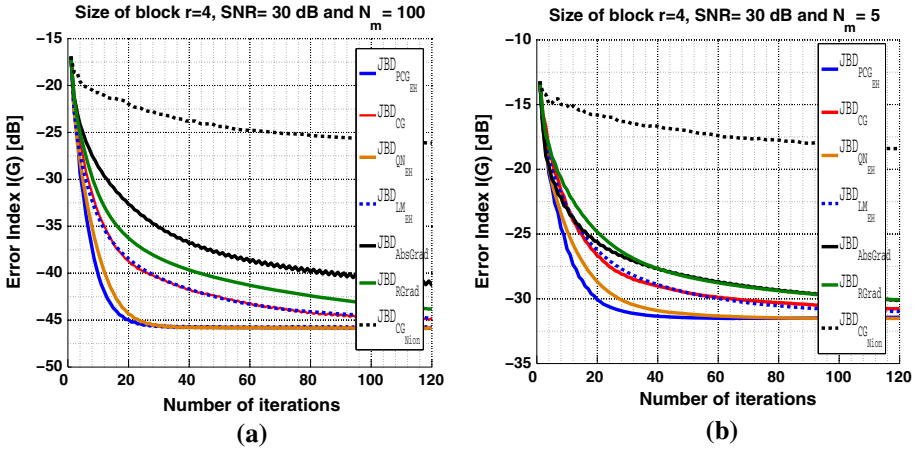
The evolution of the error index versus the number of iterations is illustrated emphasizing the influence of i) the number N_m of matrices to be block diagonalized, ii) and SNR. In fact, two cases are considered: i) on the left N_m=100 ii) and on the right $N_m = 5$ for the considered matrices set. In addition, we illustrate in the Fig. 1 (resp. the Fig. 2) the evolution of the error index versus the number of iterations in the noiseless case ($\mathsf{SNR} = 100$ dB) (resp. the noisy case ($\mathsf{SNR} = 30$ dB)).

From these curves, several observations can be drawn. First we observe that the convergence of algorithms based on an exact preconditioning namely $\mathsf{JBD}_{\mathsf{PCG}_{\mathsf{EH}}}$, $\mathsf{JBD}_{\mathsf{LM}_{\mathsf{EH}}}$ and $\mathsf{JBD}_{\mathsf{QN}_{\mathsf{EH}}}$ are the fastest algorithms. Moreover, comparing the convergence of different gradient approaches namely $\mathsf{JBD}_{\mathsf{CG}_{\mathsf{NION}}}$ and $\mathsf{JBD}_{\mathsf{RGrad}}$, we check the interest of our proposed preconditioner based on the exact Hessian matrices. Furthermore, we deduce that the convergence speed of the algorithm $\mathsf{JBD}_{\mathsf{CG}}$ is higher than $\mathsf{JBD}_{\mathsf{RGrad}}$, $\mathsf{JBD}_{\mathsf{AbsGrad}}$ and especially the algorithm $\mathsf{JBD}_{\mathsf{CG}_{\mathsf{NION}}}$ which follows the same adaptation rule but operates on a different cost function. However, the best performances are generally achieved using the $\mathsf{JBD}_{\mathsf{PCG}_{\mathsf{EH}}}$ algorithm in difficult contexts: with noise and a set containing a small number of matrices to be joint block diagonalized. In addition, we note that $\mathsf{JBD}_{\mathsf{PCG}_{\mathsf{EH}}}$ and $\mathsf{JBD}_{\mathsf{QN}_{\mathsf{EH}}}$ converge



(a) (b)

Fig. 1 Comparison of the different algorithms: evolution of the error index $I(\mathbf{G})$ versus the number of iterations in the noiseless case (SNR =100 dB) for different sizes of the matrix sets. **a** $N_m = 100$, **b** $N_m = 5$



(a) (b)

Fig. 2 Comparison of the different algorithms: evolution of the error index $I(\mathbf{G})$ versus the number of iterations in the noisy case (SNR =30 dB) for different sizes of the matrix sets. **a** $N_m = 100$, **b** $N_m = 5$

to the same solution approximately. However, the others require more iterations to reach the same performances.

We observe also that the more matrices to be joint block-diagonalized we have, the best the obtained results are. In addition, whatever the considered algorithm, they all reach the same level of performance (which not too surprising either), yet, the $\mathsf{JBD}_{\mathsf{CG}_{\mathsf{NION}}}$, $\mathsf{JBD}_{\mathsf{RGrad}}$ and $\mathsf{JBD}_{\mathsf{AbsGrad}}$ algorithms need more iterations. Even though when smaller subsets of matrices and the SNR value are considered, the results remain relatively good. Not too surprisingly, the noise affects the performances.

6 Application to blind source separation

In this section, we show how to use the proposed JBD algorithms for solving the FIR convolution of non-stationary sources. We recall that the principle of the BSS problem is to

restore multiple sources mixed through an unknown mixing system that we must estimate from the system outputs only called "the observations"'. To do this, we are basically interested in methods based on spatial time-frequency distributions or spectra. It is well known that this kind of BSS methods generally need four main steps to guarantee successful signal separation.

6.1 The matrix model of the source separation problem

Considering m ($m \in \mathbb{N}^*$) observation signals $x_i(t),\ i = 1, \ldots, m,\ t \in \mathbb{Z}$, and n ($n \in \mathbb{N}^*$) sources $s_j(t)$, for all $j = 1, \ldots, n$ which are mixed through a linear FIR multichannel system and represented by $\mathbf{H}(t) = (H_{ij}(t))$. The convolutive model can be simply described by the following expression :

$$x_i(t) = \sum_{j=1}^{n} \sum_{\ell=0}^{L} H_{ij}(\ell) s_j(t-\ell) + n_j(t), \tag{32}$$

where $H_{ij}(t)$ is the impulse response function between the i-th sensor and j-th source with an overall extent of $L+1$ taps and $n_i(t)$, for all $i = 1, \ldots, m$ are noises. Hence, the convolutive system can be written as an instantaneous model of source separation problem:

$$\mathbf{X}(t) = \mathbf{A}\mathbf{S}(t) + \mathbf{N}(t), \tag{33}$$

where:

- The $(M \times N)$ mixing matrix $\mathbf{A}$ is a block-matrix given by $\mathbf{A} = \left(\mathbf{A}_{ij}\right)$ for all $i = 1, \ldots, m$ and $j = 1, \ldots, n$ whose blocks $\mathbf{A}_{ij}$ are $(L' \times Q)$ Toeplitz matrices:

$$\mathbf{A}_{ij} = \begin{pmatrix} H_{ij}(0) & \ldots & H_{ij}(L) & 0 & \ldots & 0 \\ 0 & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & \ldots & 0 & H_{ij}(0) & \ldots & H_{ij}(L) \end{pmatrix}. \tag{34}$$

- The $(N \times 1)$ vector $\mathbf{S}(t) = [\mathbf{s}_1(t)^T, \mathbf{s}_2(t)^T, \ldots, \mathbf{s}_n(t)^T]^T$ containing sources where the $(Q \times 1)$ vectors $\mathbf{s}_i(t)$ (for all $i = 1, \ldots, n$) stands for $\mathbf{s}_i(t) = [s_i(t), s_i(t-1), \ldots, s_i(t-Q+1)]^T$,
- The $(M \times 1)$ vector $\mathbf{X}(t) = [\mathbf{x}_1(t)^T, \mathbf{x}_2(t)^T, \ldots, \mathbf{x}_m(t)^T]^T$ (resp. $\mathbf{N}(t) = [\mathbf{n}_1(t)^T, \mathbf{n}_2(t)^T, \ldots, \mathbf{n}_m(t)^T]^T$) containing the observed signals (resp. the noise signals) where the $(L' \times 1)$ vectors $\mathbf{x}_i(t)$ (resp. $\mathbf{n}_i(t)$) stands for $\mathbf{x}_i(t) = [x_i(t), x_i(t-1), \ldots, x_i(t-L'+1)]^T$ (resp. $\mathbf{n}_i(t) = [n_i(t), n_i(t-1), \ldots, n_i(t-L'+1)]^T$),
- $M = mL'$, $N = n(L+L') = nQ$ (with $Q = L + L'$ and $L' \in \mathbb{N}^*$) and L' is chosen such that $M \geq N$ to maintain an over-determined model.

We assume that the noises are stationary, white, centered random signals, mutually uncorrelated and independent from the source signals.

6.2 Construction of the *t-f* matrices set to be joint block-diagonalized

The Spatial Quadratic Time-Frequency Spectrum (SQTFS) of the observations across the array at a given *t-f* point is a $(M \times M)$ matrix, admits the following decomposition:

$$\begin{aligned}\mathbf{D_X}(t,\nu) &= \mathbf{A}\mathbf{D_S}(t,\nu)\mathbf{A}^H + \mathbf{D_N}(t,\nu) + \mathbf{A}\mathbf{D_{SN}}(t,\nu) + \mathbf{D_{NS}}(t,\nu)\mathbf{A}^H \\ &= \mathbf{A}\mathbf{D_S}(t,\nu)\mathbf{A}^H + \mathbf{D_N}(t,\nu),\end{aligned} \tag{35}$$

where,

- $\mathbf{D_S}(t,\nu)$ (resp. $\mathbf{D_N}(t,\nu)$) is the $(N \times N)$ source SQTFS (resp. the $(M \times M)$ noise SQTFS).
- $\mathbf{D_{SN}}(t,\nu)$ and $\mathbf{D_{NS}}(t,\nu)$ are the Spatial Bilinear Time-Frequency Spectrum (SBTFS) between the sources and the noises. These SBTFS are null since the noises are independent from the source signals.

We have to recall a very important property, as expressed in Ghennioui et al. 2010, which allows to solve the BSS problem such that the matrix $\mathbf{D_S}(t,\nu)$ takes a specific algebraic structure. In fact, it is block-diagonal with one single non null $(Q \times Q)$ block on the block-diagonal. Moreover, the block-diagonal matrix with one single non null block is the only possibility of block-diagonal matrix when these matrices outcome of spatial time-frequency distributions or spectrum (SQTFD(or S)).

In order to build the set of matrices denoted $\mathcal{M}_{\mathsf{JBD}}$ for joint block-diagonalization. We follow the same procedures for the detector proposed in Ghennioui et al. 2010 denoted $\mathsf{C}_{\mathsf{Conv}_{\mathsf{GH}}}$. All the matrices from the set $\mathcal{M}_{\mathsf{JBD}}$ admit the decomposition given by equation (35) or (1) (after the noise reducing process Ghennioui et al. 2010), where the source SQTFD(or S) matrices $\mathbf{D_S}(t,\nu)$ assume a very specific algebraic structure (block diagonal matrices). As a result, to tackle the BSS problem, we use our proposed JBD algorithms.

6.3 Estimation of the separation matrix

The matrices belonging to the set $\mathcal{M}_{\mathsf{JBD}}$ of size N_m ($N_m \in \mathbb{N}^*$) can be decomposed into $\mathbf{A}\mathbf{D_S}(t,\nu)\mathbf{A}^H$ with $\mathbf{D_S}(t,\nu)$ a block-diagonal matrix with only one non null $(Q \times Q)$ block on its block-diagonal. One possible way to be able to recover the mixing matrix $\mathbf{A}$ (or its pseudo-inverse: the separation matrix $\mathbf{B}$) is to directly joint block diagonalize the matrices set $\mathcal{M}_{\mathsf{JBD}}$. As known, the sources recovered are obtained up to a permutation and up to a filter which are the classical indeterminations of the BSS in the convolutive context.

Four BSS methods can be then derived: the first one is denoted $\mathsf{JBD}_{\mathsf{PCG}_{\mathsf{EH-TF}}}$ since it combines the JBD algorithm based on a preconditioned conjugate gradient approach $\mathsf{JBD}_{\mathsf{PCG}_{\mathsf{EH}}}$ together with the automatic time-frequency points detector $\mathsf{C}_{\mathsf{Conv}_{\mathsf{GH}}}$. The three other methods denoted respectively $\mathsf{JBD}_{\mathsf{QN}_{\mathsf{EH-TF}}}$, $\mathsf{JBD}_{\mathsf{LM}_{\mathsf{EH-TF}}}$ and $\mathsf{JBD}_{\mathsf{CG}_{\mathsf{TF}}}$ consist of replacing the preconditioned conjugate gradient-based JBD algorithm by respectively the Quasi-Newton algorithm $\mathsf{JBD}_{\mathsf{QN}_{\mathsf{EH}}}$, the Levenberg–Marquardt algorithm $\mathsf{JBD}_{\mathsf{LM}_{\mathsf{EH}}}$ and the conjugate gradient algorithm $\mathsf{JBD}_{\mathsf{CG}}$.

6.4 Computer simulations

In this section, computer simulations are performed in order to illustrate the good performance of the four suggested methods and to compare them again with the same kind of existing approaches: the first one is denoted by $\mathsf{JBD}_{\mathsf{CG}_{\mathsf{NION-TF}}}$ and combines the non-unitary JBD algorithm proposed in Nion 2011 with the *t-f* point detector $\mathsf{C}_{\mathsf{Conv}_{\mathsf{GH}}}$. The other algorithms denoted $\mathsf{JBD}_{\mathsf{RGrad-TF}}$ and $\mathsf{JBD}_{\mathsf{AbsGrad-TF}}$ combine respectively the non-unitary JBD algorithm proposed in Ghennioui et al. 2010 and the non-unitary JBD algorithm proposed in Ghennioui et al. 2008a with the same detector.

We consider $m = 4$ mixtures of $n = 2$ sources of 128 time samples. The first source (resp. the second source) is a sinusoidal frequency modulation (resp. a linear frequency

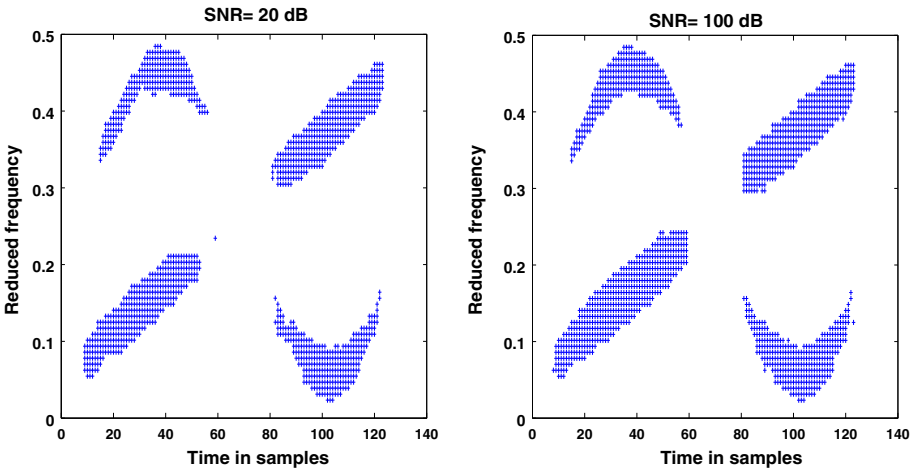


Fig. 3 The selected *t-f* points with automatic points detector $\mathsf{C}_{\mathsf{Conv}_{\mathsf{GH}}}$

modulation), $L = 3$ and $L' = 2$. These sources are mixed according to a mixture matrix $\mathbf{A}(t)$ whose components were randomly generated and whose z-transform $\mathbf{A}[z]$ is given by:

$$\begin{pmatrix} -0.0134 + 0.2221z^{-1} + 0.9749z^{-2} & -0.6579 + 0.7521z^{-1} - 0.0382z^{-2} \\ -0.7425 - 0.6639z^{-1} - 0.0891z^{-2} & 0.7287 + 0.6049z^{-1} - 0.3210z^{-2} \\ 0.6173 + 0.3921z^{-1} - 0.6820z^{-2} & 0.9477 - 0.2211z^{-1} - 0.2301z^{-2} \\ 0.7119 - 0.6532z^{-1} - 0.2578z^{-2} & 0.2533 - 0.7511z^{-1} - 0.6096z^{-2} \end{pmatrix},$$

We use the Spatial Pseudo Wigner-Ville Spectra (SPWVS) with a Hamming smoothing window of size 128 and 64 frequency bins. In this example, 30 realizations of signals are computed. Then, the SPWVS of each realization is calculated. For the noises, we use random entries chosen from a Gaussian distribution with zero mean and variance σ_N^2.

In the Fig. 4, we have plotted the error index I(G) versus the SNR (≈ 1410 time-frequency matrices were selected when SNR ≥ 20 dB (see the Fig. 3 for the selected *t-f* points). In fact, the more the SNR decreases (for SNR ≤ 20 dB), the more the performance decreases too. The resulting error index shows that the four proposed methods show the best performances. We observe again that the performances of proposed algorithms based on an exact preconditioning namely $\mathsf{JBD}_{\mathsf{PCG}_{\mathsf{EH-TF}}}$, $\mathsf{JBD}_{\mathsf{QN}_{\mathsf{EH-TF}}}$ and $\mathsf{JBD}_{\mathsf{LM}_{\mathsf{EH-TF}}}$ are the fastest algorithms. The $\mathsf{JBD}_{\mathsf{CG}_{\mathsf{TF}}}$ method still performs better than $\mathsf{JBD}_{\mathsf{CG}_{\mathsf{NION-TF}}}$ based on conjugate gradient and which exhibits, in this simulation, the worst performances especially in the noisy context. However, the $\mathsf{JBD}_{\mathsf{PCG}_{\mathsf{EH-TF}}}$ and $\mathsf{JBD}_{\mathsf{QN}_{\mathsf{EH-TF}}}$ methods that provide the best results in the same context. Finally, all *t-f* methods reach the same performances especially when SNR ≥ 40 dB except, logically, $\mathsf{JBD}_{\mathsf{CG}_{\mathsf{NION-TF}}}$ which operates on a different cost function Nion 2011.

7 Discussion and conclusion

In this article, we are interested in non unitary joint block diagonalization problem which is of great interest in different fields of application such that array processing for wideband signals and blind sources separation when convolutive mixtures are considered. Our

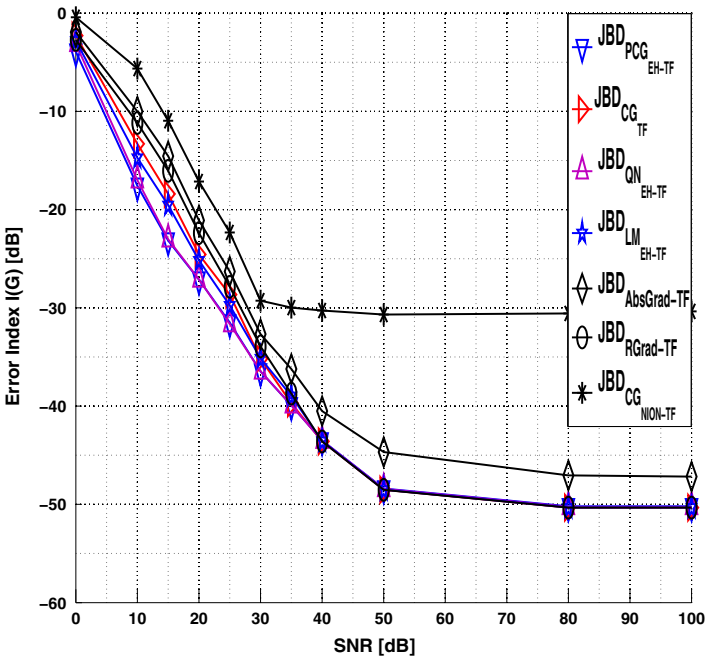


Fig. 4 Comparison of the different algorithms in the BSS context: evolution of the error index $I(\mathbf{G})$ versus the SNR

aim, here, was to investigate the relevance/interest of a preconditioning for those algorithms. Such a procedure usually requires either the calculation of the exact complex Hessian matrix at each iteration. In this article, the exact calculation has been performed. Then, different iterative preconditioned joint block diagonalization algorithms have been derived and compared. We have focused our attention on Quasi-Newton, Preconditioned Conjugate Gradient and Levenberg–Marquardt algorithms. Computer simulations presented in difficult contexts (noise and/or very few matrices to be joint block diagonalized) have emphasized the good behavior of the preconditioned conjugate gradient algorithm. This algorithm offers the best compromise between good performance and fast convergence speed even in complicated scenarios. The main advantage of the approach suggested here remains its really general aspect. In fact, it does not rely on restrictive assumptions neither about the considered matrix set (they only have to be complex but not necessarily hermitian matrices) nor about the joint block-diagonalizer (it is not necessarily an unitary matrix). Finally, we have used the proposed algorithms to show their usefulness in the BSS context. All these algorithms have brought four time-frequency based BSS methods combined with a *t-f* points detector $\mathsf{C}_{\mathsf{Conv}_{\mathsf{GH}}}$.

Appendix A: Calculation of the complex Hessian matrices

We consider three square matrices, $\mathbf{D}_1$, $\mathbf{D}_2$ and $\mathbf{D}_3$ in $\mathbb{C}^{M\times M}$ and five rectangular matrices $\mathbf{D}_4, \mathbf{D}_5, \mathbf{D}_6$ in $\mathbb{C}^{N\times M}$, $\mathbf{D}_7$ in $\mathbb{C}^{M\times N}$ and $\mathbf{D}_8$ in $\mathbb{C}^{F\times G}$. Let $\mathsf{tr}\,\{\cdot\}$, $d\,(\cdot)$ (or sometimes simply d), $\mathsf{vec}\,(\cdot)$, $\mathsf{OffBdiag}_{(\mathbf{n})}\{\cdot\}$ and $\mathbf{T}_{\mathrm{Boff}}$ respectively denote the trace operator of a square matrix, the differential operator, the vectorization operator that stacks the columns of a matrix in a

long column vector, the zero-block-diagonal operator defined in Ghennioui et al. 2010 and the $N^2 \times N^2$ "transformation" matrix defined before. Our developments are based on the following properties (they can be found in (Ghennioui et al. 2010; Hjorungnes, A. 2011; Brewer 1978) or Magnus and Neudecker 1999b):

$\mathbf{P}_0$. $\|\mathsf{OffBdiag}_{(\mathbf{n})}\{\mathbf{D}_1\}\|_F^2 = \mathsf{tr}\left\{\mathbf{D}_1^H \mathsf{OffBdiag}_{(\mathbf{n})}\{\mathbf{D}_1\}\right\}$.
$\mathbf{P}_1$. $(\mathbf{D}_4 + \mathbf{D}_5)^T = \mathbf{D}_4^T + \mathbf{D}_5^T$.
$\mathbf{P}_2$. $(\mathbf{D}_4\mathbf{D}_7)^T = \mathbf{D}_7^T\mathbf{D}_4^T$.
$\mathbf{P}_3$. $(\mathbf{D}_4\mathbf{D}_7)^* = \mathbf{D}_4^*\mathbf{D}_7^*$.
$\mathbf{P}_4$. $\mathsf{tr}\{\mathbf{D}_1 + \mathbf{D}_2\} = \mathsf{tr}\{\mathbf{D}_1\} + \mathsf{tr}\{\mathbf{D}_2\}$.
$\mathbf{P}_5$. $\mathsf{tr}\left\{\mathbf{D}_4^H\mathbf{D}_5\right\} = (\mathsf{vec}(\mathbf{D}_4))^H \mathsf{vec}(\mathbf{D}_5)$.
$\mathbf{P}_6$. $d(\mathsf{tr}\{\mathbf{D}_1\}) = \mathsf{tr}\{d(\mathbf{D}_1)\}$.
$\mathbf{P}_7$. $\mathsf{vec}(\mathbf{D}_1\mathbf{D}_2\mathbf{D}_3) = (\mathbf{D}_3^T \otimes \mathbf{D}_1)\mathsf{vec}(\mathbf{D}_2) \Rightarrow \mathsf{vec}(\mathbf{D}_1\mathbf{D}_2) = (\mathbf{I}_N \otimes \mathbf{D}_1)\mathsf{vec}(\mathbf{D}_2) = (\mathbf{D}_2^T \otimes \mathbf{I}_N)\mathsf{vec}\{\mathbf{D}_1\}$
$\mathbf{P}_{7b}$. $\mathsf{vec}(\mathbf{D}_4\mathbf{D}_1\mathbf{D}_7) = (\mathbf{D}_7^T \otimes \mathbf{D}_4)\mathsf{vec}(\mathbf{D}_1) \Rightarrow \mathsf{vec}(\mathbf{D}_4\mathbf{D}_1) = (\mathbf{I}_M \otimes \mathbf{D}_4)\mathsf{vec}(\mathbf{D}_1) = (\mathbf{D}_1^T \otimes \mathbf{I}_N)\mathsf{vec}\{\mathbf{D}_4\}$
$\mathbf{P}_8$. $\mathsf{vec}\left(\mathbf{D}_4^T\right) = \mathbf{K}_{N,M}\mathsf{vec}(\mathbf{D}_4)$ with $\mathbf{K}_{N,M}$ the unique square $NM \times NM$ commutation matrix defined as: $\mathbf{K}_{N,M} = \sum_n^N \sum_m^M \mathbf{E}_{nm} \otimes \mathbf{E}_{nm}^T$ where $\mathbf{E}_{nm}$ are $N \times M$ matrices with a 1 in the (n, m) position and zeros elsewhere: $(\mathbf{E}_{nm}) = (\delta_{nm})$ (δ_{nm} standing for the Kronecker Delta). As a consequence, the permutation matrix $\mathbf{K}_{N,M}$ has one single "1" in each row and in each column.
$\mathbf{P}_{8b}$. $\mathbf{K}_{N,M}\mathbf{K}_{M,N} = \mathbf{I}_{MN}$ and $\mathbf{K}_{N,M} = \mathbf{K}_{M,N}^{-1} = \mathbf{K}_{M,N}^T$.
$\mathbf{P}_9$. $\mathsf{vec}(\mathsf{OffBdiag}\{\mathbf{D}_1\}) = \mathbf{T}_{\text{Boff}}\mathsf{vec}(\mathbf{D}_1)$.
$\mathbf{P}_{10}$. $d(\mathsf{vec}(\mathbf{D}_4\}) = \mathsf{vec}(d(\mathbf{D}_4))$.
$\mathbf{P}_{11}$. $d(\mathbf{D}_4 + \mathbf{D}_5) = d(\mathbf{D}_4) + d(\mathbf{D}_5)$.
$\mathbf{P}_{12}$. $d(\mathbf{D}_4\mathbf{D}_7) = d(\mathbf{D}_4)\mathbf{D}_7 + \mathbf{D}_4 d(\mathbf{D}_7)$.
$\mathbf{P}_{13}$. $d\left(\mathbf{D}_4^*\right) = d(\mathbf{D}_4)^*$.
$\mathbf{P}_{14}$. $\mathsf{tr}\{\mathbf{D}_1) = \mathsf{tr}\left\{\mathbf{D}_1^T\right\}$.
$\mathbf{P}_{15}$. $(\mathbf{D}_4 \otimes \mathbf{D}_8)^T = \mathbf{D}_4^T \otimes \mathbf{D}_8^T$.
$\mathbf{P}_{16}$. $(\mathbf{D}_4 \otimes \mathbf{D}_8)^H = \mathbf{D}_4^H \otimes \mathbf{D}_8^H$.
$\mathbf{P}_{17}$. $\mathsf{vec}(\mathbf{D}_4 + \mathbf{D}_5) = \mathsf{vec}(\mathbf{D}_4) + \mathsf{vec}(\mathbf{D}_5)$.
$\mathbf{P}_{18}$. $\mathbf{D}_4^H = \left(\mathbf{D}_4^*\right)^T$.
$\mathbf{P}_{19}$. $(\mathbf{D}_4\mathbf{D}_7)^H = \mathbf{D}_7^H\mathbf{D}_4^H$.
$\mathbf{P}_{20}$. $(\mathbf{D}_1 \otimes \mathbf{D}_2)(\mathbf{D}_3 \otimes \mathbf{D}_7) = (\mathbf{D}_1\mathbf{D}_3 \otimes \mathbf{D}_2\mathbf{D}_7)$.
$\mathbf{P}_{21}$. $(\mathbf{D}_4 + \mathbf{D}_5)^H = \left(\mathbf{D}_4^H + \mathbf{D}_5^H\right)$.
$\mathbf{P}_{22}$. $\mathsf{OffBdiag}_{(\mathbf{n})}\{\mathbf{D}_1 + \mathbf{D}_2\} = \mathsf{OffBdiag}_{(\mathbf{n})}\{\mathbf{D}_1\} + \mathsf{OffBdiag}_{(\mathbf{n})}\{\mathbf{D}_2\}$.
$\mathbf{P}_{23}$. $d\mathbf{D}_4^* = (d\mathbf{D}_4)^*$.

According to Hjorungnes, A. 2011, the first-order differential of the cost function $\mathcal{C}_{\mathsf{JBD}}(\mathbf{B}, \mathbf{B}^*)$ defined in (3) is given by:

$$d\mathcal{C}_{\mathsf{JBD}}(\mathbf{B}, \mathbf{B}^*) = \mathcal{D}_{\mathbf{B}}\left(\mathcal{C}_{\mathsf{JBD}}(\mathbf{B}, \mathbf{B}^*)\right) d\mathsf{vec}(\mathbf{B}) + \mathcal{D}_{\mathbf{B}^*}\left(\mathcal{C}_{\mathsf{JBD}}(\mathbf{B}, \mathbf{B}^*)\right) d\mathsf{vec}(\mathbf{B}^*), \tag{36}$$

if the $1 \times NM$ complex vectors $\mathcal{D}_{\mathbf{B}}(\mathcal{C}_{\mathsf{JBD}}(\mathbf{B}, \mathbf{B}^*))$ and $\mathcal{D}_{\mathbf{B}^*}(\mathcal{C}_{\mathsf{JBD}}(\mathbf{B}, \mathbf{B}^*))$ are defined as:

$$\begin{aligned}
\mathcal{D}_{\mathbf{B}}\left(\mathcal{C}_{\mathsf{JBD}}(\mathbf{B}, \mathbf{B}^*)\right) &= \mathsf{vec}^T\left(\frac{\partial \mathcal{C}_{\mathsf{JBD}}(\mathbf{B}, \mathbf{B}^*)}{\partial \mathbf{B}}\right), \\
\mathcal{D}_{\mathbf{B}^*}\left(\mathcal{C}_{\mathsf{JBD}}(\mathbf{B}, \mathbf{B}^*)\right) &= \mathsf{vec}^T\left(\frac{\partial \mathcal{C}_{\mathsf{JBD}}(\mathbf{B}, \mathbf{B}^*)}{\partial \mathbf{B}^*}\right).
\end{aligned} \tag{37}$$

We also recall that the two partial derivatives that are involved *i.e.* $\frac{\partial \mathcal{C}_{\mathsf{JBD}}(\mathbf{B},\mathbf{B}^*)}{\partial \mathbf{B}}$ and $\frac{\partial \mathcal{C}_{\mathsf{JBD}}(\mathbf{B},\mathbf{B}^*)}{\partial \mathbf{B}^*}$ were calculated in Ghennioui et al. 2010 using some of the previous properties $\mathbf{P}_0$-$\mathbf{P}_{20}$. They were found to be equal to:

$$\frac{\partial \mathcal{C}_{\mathsf{JBD}}(\mathbf{B},\mathbf{B}^*)}{\partial \mathbf{B}} = \sum_{i=1}^{N_m} \left[\left(\mathsf{OffBdiag}_{(\mathbf{n})}\{\mathbf{B}\mathbf{M}_i\mathbf{B}^H\} \right)^T \mathbf{B}^*\mathbf{M}_i^* + \left(\mathsf{OffBdiag}_{(\mathbf{n})}\{\mathbf{B}\mathbf{M}_i\mathbf{B}^H\} \right)^* \mathbf{B}^*\mathbf{M}_i^T \right], \quad (38)$$

$$\begin{aligned} \frac{\partial \mathcal{C}_{\mathsf{JBD}}(\mathbf{B},\mathbf{B}^*)}{\partial \mathbf{B}^*} &= \sum_{i=1}^{N_m} \left[\mathsf{OffBdiag}_{(\mathbf{n})}\{\mathbf{B}\mathbf{M}_i\mathbf{B}^H\}\mathbf{B}\mathbf{M}_i^H + \left(\mathsf{OffBdiag}_{(\mathbf{n})}\{\mathbf{B}\mathbf{M}_i\mathbf{B}^H\} \right)^H \mathbf{B}\mathbf{M}_i \right] \\ &= \left(\frac{\partial \mathcal{C}_{\mathsf{JBD}}(\mathbf{B},\mathbf{B}^*)}{\partial \mathbf{B}} \right)^*. \end{aligned} \quad (39)$$

$$\mathcal{D}_{\mathbf{B}^*}\left(\mathcal{C}_{\mathsf{JBD}}(\mathbf{B},\mathbf{B}^*)\right) = (\mathcal{D}_{\mathbf{B}}(\mathcal{C}_{\mathsf{JBD}}(\mathbf{B},\mathbf{B}^*)))^*. \quad (40)$$

As shown in Hjorungnes, A. 2011, the second order differential is then given by:

$$d^2\mathcal{C}_{\mathsf{JBD}}(\mathbf{B},\mathbf{B}^*) = d\mathcal{D}_{\mathbf{B}}\left(\mathcal{C}_{\mathsf{JBD}}(\mathbf{B},\mathbf{B}^*)\right) d\mathsf{vec}(\mathbf{B}) + d\mathcal{D}_{\mathbf{B}^*}\left(\mathcal{C}_{\mathsf{JBD}}(\mathbf{B},\mathbf{B}^*)\right) d\mathsf{vec}(\mathbf{B}^*). \quad (41)$$

Thus, we have to evaluate the differential of the two derivatives $\mathcal{D}_{\mathbf{B}}$ and $\mathcal{D}_{\mathbf{B}^*}$, *i.e.* $d\mathcal{D}_{\mathbf{B}}$ and $d\mathcal{D}_{\mathbf{B}^*}$. We start with $d\mathcal{D}_{\mathbf{B}}$. Using the properties $\mathbf{P}_2$, $\mathbf{P}_3$, $\mathbf{P}_{10}$-$\mathbf{P}_{14}$, $\mathbf{P}_{17}$-$\mathbf{P}_{19}$ in (38), we have:

$$\begin{aligned} d\mathcal{D}_{\mathbf{B}}\left(\mathcal{C}_{\mathsf{JBD}}(\mathbf{B},\mathbf{B}^*)\right) = \sum_{i=1}^{N_m} \Bigg[& \mathsf{vec}^T\left(\left(\mathsf{OffBdiag}_{(\mathbf{n})}\{d\mathbf{B}^*\mathbf{M}_i^T\mathbf{B}^T\} \right) \mathbf{B}^*\mathbf{M}_i^* \right. \\ & \left. + \left(\mathsf{OffBdiag}_{(\mathbf{n})}\{\mathbf{B}^*\mathbf{M}_i^T d\mathbf{B}^T\} \right) \mathbf{B}^*\mathbf{M}_i^* \right) \\ & + \mathsf{vec}^T\left(\left(\mathsf{OffBdiag}_{(\mathbf{n})}\{d\mathbf{B}^*\mathbf{M}_i^*\mathbf{B}^T\} \right) \mathbf{B}^*\mathbf{M}_i^T \right. \\ & \left. + \left(\mathsf{OffBdiag}_{(\mathbf{n})}\{\mathbf{B}^*\mathbf{M}_i^* d\mathbf{B}^T\} \right) \mathbf{B}^*\mathbf{M}_i^T \right) \\ & + \mathsf{vec}^T\left(\left(\mathsf{OffBdiag}_{(\mathbf{n})}\{\mathbf{B}\mathbf{M}_i\mathbf{B}^H\} \right)^T d\mathbf{B}^*\mathbf{M}_i^* \right. \\ & \left. + \left(\mathsf{OffBdiag}_{(\mathbf{n})}\{\mathbf{B}\mathbf{M}_i^H\mathbf{B}^H\} \right)^T d\mathbf{B}^*\mathbf{M}_i^T \right) \Bigg]. \end{aligned} \quad (42)$$

Then, using the properties $\mathbf{P}_1$, $\mathbf{P}_7$, and $\mathbf{P}_{17}$ we obtain:

$$\begin{aligned} d\mathcal{D}_{\mathbf{B}}\left(\mathcal{C}_{\mathsf{JBD}}(\mathbf{B},\mathbf{B}^*)\right) = \sum_{i=1}^{N_m} \Bigg[& \left((\mathbf{B}^*\mathbf{M}_i^*)^T \otimes \mathbf{I}_N \right) \mathsf{vec}\left(\mathsf{OffBdiag}_{(\mathbf{n})}\{d\mathbf{B}^*\mathbf{M}_i^T\mathbf{B}^T\} \right) \\ & + \left((\mathbf{B}^*\mathbf{M}_i^*)^T \otimes \mathbf{I}_N \right) \mathsf{vec}\left(\mathsf{OffBdiag}_{(\mathbf{n})}\{\mathbf{B}^*\mathbf{M}_i^T d\mathbf{B}^T\} \right) \\ & + \left(\left(\mathbf{B}^*\mathbf{M}_i^T \right)^T \otimes \mathbf{I}_N \right) \mathsf{vec}\left(\mathsf{OffBdiag}_{(\mathbf{n})}\{d\mathbf{B}^*\mathbf{M}_i^*\mathbf{B}^T\} \right) \\ & + \left(\left(\mathbf{B}^*\mathbf{M}_i^T \right)^T \otimes \mathbf{I}_N \right) \mathsf{vec}\left(\mathsf{OffBdiag}_{(\mathbf{n})}\{\mathbf{B}^*\mathbf{M}_i^* d\mathbf{B}^T\} \right) \\ & + \left(\mathbf{I}_N \otimes \left(\mathsf{OffBdiag}_{(\mathbf{n})}\{\mathbf{B}\mathbf{M}_i\mathbf{B}^H\} \right)^T \right) \left(\mathbf{M}_i^H \otimes \mathbf{I}_N \right) \mathsf{vec}\left(d\mathbf{B}^* \right) \end{aligned}$$

$$+\left(\mathbf{I}_N \otimes \left(\mathsf{OffBdiag}_{(\mathbf{n})}\{\mathbf{B}\mathbf{M}_i^H\mathbf{B}^H\}\right)^T\right)(\mathbf{M}_i \otimes \mathbf{I}_N)\,\mathsf{vec}\left(d\mathbf{B}^*\right)\Big]^T. \tag{43}$$

While properties $\mathbf{P}_2$, $\mathbf{P}_3$, $\mathbf{P}_7$, $\mathbf{P}_8$, $\mathbf{P}_9$, and $\mathbf{P}_{18}$ involve:

$$\begin{aligned} d\mathcal{D}_{\mathbf{B}}\left(\mathcal{C}_{\mathsf{JBD}}(\mathbf{B},\mathbf{B}^*)\right) = \sum_{i=1}^{N_m}\Big[& \left(\mathbf{M}_i^H\mathbf{B}^H \otimes \mathbf{I}_N\right)\mathbf{T}_{\mathrm{Boff}}\left(\mathbf{B}\mathbf{M}_i \otimes \mathbf{I}_N\right)\mathsf{vec}(d\mathbf{B}^*) \\ & + \left(\mathbf{M}_i^H\mathbf{B}^H \otimes \mathbf{I}_N\right)\mathbf{T}_{\mathrm{Boff}}\left(\mathbf{I}_N \otimes \mathbf{B}^*\mathbf{M}_i^T\right)\mathbf{K}_{N,M}\mathsf{vec}(d\mathbf{B}) \\ & + \left(\mathbf{M}_i\mathbf{B}^H \otimes \mathbf{I}_N\right)\mathbf{T}_{\mathrm{Boff}}\left(\mathbf{B}\mathbf{M}_i^H \otimes \mathbf{I}_N\right)\mathsf{vec}(d\mathbf{B}^*) \\ & + \left(\mathbf{M}_i\mathbf{B}^H \otimes \mathbf{I}_N\right)\mathbf{T}_{\mathrm{Boff}}\left(\mathbf{I}_N \otimes \mathbf{B}^*\mathbf{M}_i^*\right)\mathbf{K}_{N,M}\mathsf{vec}(d\mathbf{B}) \\ & + \left(\mathbf{I}_N \otimes \left(\mathsf{OffBdiag}_{(\mathbf{n})}\{\mathbf{B}\mathbf{M}_i\mathbf{B}^H\}\right)^T\right)\left(\mathbf{M}_i^H \otimes \mathbf{I}_N\right)\mathsf{vec}(d\mathbf{B}^*) \\ & + \left(\mathbf{I}_N \otimes \left(\mathsf{OffBdiag}_{(\mathbf{n})}\{\mathbf{B}\mathbf{M}_i^H\mathbf{B}^H\}\right)^T\right)(\mathbf{M}_i \otimes \mathbf{I}_N)\,\mathsf{vec}(d\mathbf{B}^*)\Big]^T. \end{aligned} \tag{44}$$

The properties $\mathbf{P}_1$, $\mathbf{P}_2$, $\mathbf{P}_3$, $\mathbf{P}_{15}$, $\mathbf{P}_{16}$, $\mathbf{P}_{17}$, $\mathbf{P}_{18}$, $\mathbf{P}_{19}$, and $\mathbf{P}_{20}$ imply the following result:

$$\begin{aligned} d\mathcal{D}_{\mathbf{B}}\left(\mathcal{C}_{\mathsf{JBD}}(\mathbf{B},\mathbf{B}^*)\right) = \sum_{i=1}^{N_m}\mathsf{vec}^T(d\mathbf{B})\Big[& \left(\mathbf{K}_{N,M}\right)^T\left(\mathbf{I}_N \otimes \mathbf{M}_i\mathbf{B}^H\right)\mathbf{T}_{\mathrm{Boff}}^T\left(\mathbf{B}^*\mathbf{M}_i^* \otimes \mathbf{I}_N\right) \\ & + \left(\mathbf{K}_{N,M}\right)^T\left(\mathbf{I}_N \otimes \mathbf{M}_i^H\mathbf{B}^H\right)\mathbf{T}_{\mathrm{Boff}}^T\left(\mathbf{B}^*\mathbf{M}_i^T \otimes \mathbf{I}_N\right)\Big] \\ & + \mathsf{vec}^T\left(d\mathbf{B}^*\right)\Big[\left(\mathbf{M}_i^T\mathbf{B}^T \otimes \mathbf{I}_N\right)\mathbf{T}_{\mathrm{Boff}}^T\left(\mathbf{B}^*\mathbf{M}_i^* \otimes \mathbf{I}_N\right) \\ & + \left(\mathbf{M}_i^*\mathbf{B}^T \otimes \mathbf{I}_N\right)\mathbf{T}_{\mathrm{Boff}}^T\left(\mathbf{B}^*\mathbf{M}_i^T \otimes \mathbf{I}_N\right) \\ & + \left(\mathbf{M}_i^* \otimes \left(\mathsf{OffBdiag}_{(\mathbf{n})}\{\mathbf{B}\mathbf{M}_i\mathbf{B}^H\}\right)\right) \\ & + \left(\mathbf{M}_i^T \otimes \left(\mathsf{OffBdiag}_{(\mathbf{n})}\{\mathbf{B}\mathbf{M}_i^H\mathbf{B}^H\}\right)\right)\Big]. \end{aligned} \tag{45}$$

Finally this $1 \times NM$ vector can be rewritten as follows:

$$d\mathcal{D}_{\mathbf{B}}\left(\mathcal{C}_{\mathsf{JBD}}(\mathbf{B},\mathbf{B}^*)\right) = \left[\, d\mathsf{vec}^T(\mathbf{B}^*)\; d\mathsf{vec}^T(\mathbf{B})\,\right]\begin{bmatrix}\mathbf{A}_{00}\\ \mathbf{A}_{10}\end{bmatrix}, \tag{46}$$

which leads to the expressions of $\mathbf{A}_{00}$ and $\mathbf{A}_{10}$ given in (12)-(13).

Concerning the second differential $d\mathcal{D}_{\mathbf{B}^*}$, we are taking advantage of (40). Using property $\mathbf{P}_{23}$, we finally have $d\mathcal{D}_{\mathbf{B}^*} = \left(d\mathcal{D}_{\mathbf{B}}\left(\mathcal{C}_{\mathsf{JBD}}(\mathbf{B},\mathbf{B}^*)\right)\right)^*$, which leads to:

$$d\mathcal{D}_{\mathbf{B}^*}\left(\mathcal{C}_{\mathsf{JBD}}(\mathbf{B},\mathbf{B}^*)\right) = \left[\, d\mathsf{vec}^T(\mathbf{B})\; d\mathsf{vec}^T(\mathbf{B}^*)\,\right]\begin{bmatrix}\mathbf{A}_{00}^*\\ \mathbf{A}_{10}^*\end{bmatrix}. \tag{47}$$

But this $1 \times NM$ vector was supposed to be written as:

$$d\mathcal{D}_{\mathbf{B}^*}\left(\mathcal{C}_{\mathsf{JBD}}(\mathbf{B},\mathbf{B}^*)\right) = \left[\, d\mathsf{vec}^T(\mathbf{B}^*)\; d\mathsf{vec}^T(\mathbf{B})\,\right]\begin{bmatrix}\mathbf{A}_{01}\\ \mathbf{A}_{11}\end{bmatrix}, \tag{48}$$

implying that: $\mathbf{A}_{11} = \mathbf{A}_{00}^*$ and that $\mathbf{A}_{10}^* = \mathbf{A}_{01}$.

By incorporating (46) and (48) in (41), the results stated by (8) can be found, where $\mathbf{A}_{ij}$ for $i, j = 1, \ldots, 2$ are given by (12)-(13). The exact expression of the four complex Hessian matrices given in (9), (10) and (11) can be derived.

Appendix B: Finding the coefficients of the 4th-degree polynomial involved in the calculation of the optimal stepsize

To determine at each iteration the optimal stepsize, we have to evaluate $\mathcal{C}_{\mathsf{JBD}}(\mathbf{B}^{(m+1)}) = \mathcal{C}_{\mathsf{JBD}}\left(\mathbf{B}^{(m)} - \mu\mathbf{D}^{(m)}\right)$. In the following, to simplify the different expressions, the dependency upon the iteration m will be omitted. Using the properties $\mathbf{P}_{19}$ and $\mathbf{P}_{21}$, the cost function $\mathcal{C}_{\mathsf{JBD}}(\mathbf{B} - \mu\mathbf{D})$ can be expressed as:

$$\begin{aligned}
\mathcal{C}_{\mathsf{JBD}}(\mathbf{B} - \mu\mathbf{D}) &= \sum_{i=1}^{N_m} \left|\left| \mathsf{OffBdiag}_{(\mathbf{n})} \left\{ \mu^2 \mathbf{D}\mathbf{M}_i\mathbf{D}^H - \mu\left(\mathbf{B}\mathbf{M}_i\mathbf{D}^H + \mathbf{D}\mathbf{M}_i\mathbf{B}^H\right) + \mathbf{B}\mathbf{M}_i\mathbf{B}^H \right\} \right|\right|_F^2 \\
&= \sum_{i=1}^{N_m} \left|\left| \mathsf{OffBdiag}_{(\mathbf{n})} \left\{ \mu^2\mathbf{C}_2 - \mu\mathbf{C}_1 + \mathbf{C}_0 \right\} \right|\right|_F^2 \\
&= \sum_{i=1}^{N_m} \mathsf{tr}\left\{ (\mu^2\mathbf{C}_2 - \mu\mathbf{C}_1 + \mathbf{C}_0)^H \mathsf{OffBdiag}_{(\mathbf{n})} \left\{ \mu^2\mathbf{C}_2 - \mu\mathbf{C}_1 + \mathbf{C}_0 \right\} \right\}, \qquad (49)
\end{aligned}$$

where $\mathbf{C_0}$, $\mathbf{C_1}$ and $\mathbf{C_2}$ are given in (28)-(30).

Using properties $\mathbf{P}_0$, $\mathbf{P}_4$,$\mathbf{P}_{19}$, $\mathbf{P}_{21}$ and $\mathbf{P}_{22}$, we find that:

$$\begin{aligned}
\mathcal{C}_{\mathsf{JBD}}(\mathbf{B} - \mu\mathbf{D}) = &\sum_{i=1}^{N_m} \mathsf{tr}\left\{ \mathbf{C}_0^H \mathsf{OffBdiag}_{(\mathbf{n})}\{\mathbf{C}_0\} \right\} \\
&- \mu \sum_{i=1}^{N_m} \mathsf{tr}\left\{ \mathbf{C}_1^H \mathsf{OffBdiag}_{(\mathbf{n})}\{\mathbf{C}_0\} + \mathbf{C}_0^H \mathsf{OffBdiag}_{(\mathbf{n})}\{\mathbf{C}_1\} \right\} \\
&+ \mu^2 \sum_{i=1}^{N_m} \mathsf{tr}\left\{ \mathbf{C}_2^H \mathsf{OffBdiag}_{(\mathbf{n})}\{\mathbf{C}_0\} + \mathbf{C}_1^H \mathsf{OffBdiag}_{(\mathbf{n})}\{\mathbf{C}_1\} + \mathbf{C}_0^H \mathsf{OffBdiag}_{(\mathbf{n})}\{\mathbf{C}_2\} \right\} \\
&- \mu^3 \sum_{i=1}^{N_m} \mathsf{tr}\left\{ \mathbf{C}_2^H \mathsf{OffBdiag}_{(\mathbf{n})}\{\mathbf{C}_1\} + \mathbf{C}_1^H \mathsf{OffBdiag}_{(\mathbf{n})}\{\mathbf{C}_2\} \right\} \\
&+ \mu^4 \sum_{i=1}^{N_m} \mathsf{tr}\left\{ \mathbf{C}_2^H \mathsf{OffBdiag}_{(\mathbf{n})}\{\mathbf{C}_2\} \right\} = a_0 + a_1\mu + a_2\mu^2 + a_3\mu^3 + a_4\mu^4. \qquad (50)
\end{aligned}$$

It finally leads to the results stated by Eqs. (23)–(27).

References

Axelsson, O. (1985). A survey of preconditioned iterative methods for linear systems of algebraic equations. *BIT*, *25*(1), 166–187.

Belouchrani, A., Abed-Meraïm, K., Amin, M. G., & Zoubir, A. (2001). Joint anti-diagonalization for blind source separation. In *Proceedings IEEE ICASSP*, Salt Lake City, USA (pp. 2196–2199).

Belouchrani, A., Abed-Meraïm, K., Cardoso, J.-F., & Moulines, E. (1997). A blind source separation technique using second order statistics. *IEEE Transactions on Signal Processing*, *45*, 434–444.

Belouchrani, A., Amin, M. G., & Abed-Meraïm, K. (1997). Direction finding in correlated noise fields based on joint block-diagonalization of spatio-temporal correlation matrices. *IEEE Signal Processing Letters*, *4*(9), 266–268.

Belouchrani, A., Amin, M. G., Thirion-Moreau, N., & Zang, Y. (2013). Source separation and localization using time-frequency distributions. *IEEE Signal Processing Magazine*, *30*, 97–107.

Benzi, M. (2002). Preconditioning techniques for large linear systems: A survey. *Journal of Computational Physics*, *182*, 418–477.

Brewer, J. (1978). Kronecker products and matrix calculus in system theory. *IEEE Transactions on Circuits and Systems*, *25*(9), 772–781.

Cardoso, J.-F., & Souloumiac, A. (1993). Blind beamforming for non-gaussian signals. *IEE Proceedings F (Radar and Signal Processing)*, *140*(6), 362–370.

Chabriel, G., Barrère, J., Thirion-Moreau, N., & Moreau, E. (2008). Algebraic joint zero-diagonalization and blind source separation. *IEEE Transactions on Signal Processing*, *56*(3), 980–989.

Chabriel, G., Kleinsteuber, M., Moreau, E., Shen, H., Tichavsky, P., & Yeredor, A. (2014). Joint matrices decompositions and blind source separation: A survey of methods, identification, and applications. *IEEE Signal Processing Magazine*, *31*(3), 34–43.

Chabriel, G., & Barrère, J. (2011). Non-symmetrical joint zero-diagonalization and mimo zero-division multiple access. *IEEE Transactions on Signal Processing*, *59*(5), 2296–2307.

Chabriel, G., & Barrère, J. (2012). A direct algorithm for nonorthogonal approximate joint diagonalization. *IEEE Transactions on Signal Processing*, *60*(1), 39–47.

Chen, K. (2005). Matrix Preconditioning Techniques and Applications (Cambridge Monographs on Applied and Computational Mathematics).

Cherrak, O., Ghennioui, H., Thirion-Moreau, N., & Abarkan, E.-H. (2013). Non-unitary joint block diagonalization of matrices using a levenberg–marquardt algorithm. In *Proceedings of the EUSIPCO*, Marrakech, Morocco (pp. 1–5).

Comon, P. (1994). Independant component analysis, a new concept ? *Signal Process*, *36*, 287–314.

De Lathauwer, L., Févotte, C., De Moor, B. & Vandewalle, J. (2002). Jacobi algorithm for block diagonalization in blind identification. In *Proceedings of symposium information theory in the Benelux*, Louvain-la-Neuve, Belgium (pp. 155–162).

Evans, D. J. (1968). The use of pre-conditioning in iterative methods for solving linear systems with symmetric positive definite matrices. *IMA Journal of Applied Mathematics*, *4*, 295–314.

Fadaili, E. M., Thirion-Moreau, N., & Moreau, E. (2007). Non-orthogonal joint diagonalization/zero-diagonalization for source separation based on time-frequency distributions. *IEEE Transactions on Signal Processing*, *55*(5), 1673–1687.

Ghennioui, H., Fadaili, E.-M., Thirion-Moreau, N., Adib, A., & Moreau, E. (2007a). A nonunitary joint block diagonalization algorithm for blind separation of convolutive mixtures of sources. *IEEE Signal Processing Letters*, *14*(11), 860–863.

Ghennioui, H., Thirion-Moreau, N., Moreau, E., & Aboutajdine, D. (2010). Gradient-based joint block diagonalization algorithms: Application to blind separation of fir convolutive mixtures. *Signal Processing*, *90*(6), 1836–1849.

Ghennioui, H., Thirion-Moreau, N., Moreau, E., Aboutajdine, D., Adib, A. (2008). Two new gradient based non-unitary joint block-diagonalization algorithms. In *2008 16th European signal processing conference*, (pp. 1–5).

Ghennioui, H., Thirion-Moreau, N., Moreau, E., Adib, A., & Aboutajdine, D. (2007b). Non unitary joint-block diagonalization of complex matrices using a gradient approach. In et al., M. E. D., editor, *7th international conference on independent component analysis and signal separation*, London, UK. Springer-Verlag Berlin Heidelberg, LNCS 4666 (pp. 201–208).

Hager, W., & Zhang, H. (2006). A survey of nonlinear conjugate gradient methods. *Pacific Journal of Optimizations*, *2*, 35–58.

Hjorungnes, A. (2011). *Complex-valued matrix derivatives with applications in signal processing and communications*. Cambridge: University Press.

Lahat, D. Cardoso, J.-F., & Messer, H. (2012). Joint block diagonalization algorithms for optimal separation of multidimensional components. In Theis, F., Cichocki, A., Yeredor, A., & Zibulevsky, M. E., editors, *J. In Latent Variable Analysis and Signal Separation, ser. LNCS*, (Vol. 7191, pp. 155–162). Heidelberg: Springer.

Lee, D. D. & Seung, H. S. (2000). Algorithms for non-negative matrix factorization. In *In NIPS*, (pp. 556–562), MIT Press.

Luenberger, D. G. (1969). *Optimization by vector space methods*. New York: Wiley.

Magnus, J. R., & Neudecker, H. (1999). *Matrix differential calculus with applications in statistics and econometrics* (3rd ed.). New York: Wiley.

Maurandi, V. De Luigi, C. & Moreau, E. (2013). Fast jacobi like algorithms for joint diagonalization of complex symmetric matrices. In *Proceedings of the 21st European signal processing conference (EUSIPCO), 2013*, (pp. 1–5).
Moreau, E. (2001). A generalization of joint-diagonalization criteria for source separation. *IEEE Transactions Signal Procssing*, *49*(3), 530–541.
Moreau, E., & Adali, T. (2013). *Blind identification and separation of complex valued signals*. Focus Digital Signal and Image Processing Series.
Nion, D. (2011). A tensor framework for non-unitary joint block diagonalization. *IEEE Transactions Signal Processing*, *59*(10), 4585–4594.
Paatero, P. (1999). The multilinear engine-a table-driven, least-squares program for solving multi-linear problems, including the n-way parallel factor analysis mode. *Journal of Computational and Graphical Statistics*, *8*(4), 854–888.
Polak, E. (1997). *Optimization algorithms and consistent approximations*. Berlin: Springer.
Shewchuk, J. R. (1994). Introduction to the conjugate gradient method without the agonizing pain. Technical report, Carnegie Mellon University, Pittsburgh, Pennsylvania, USA.
Souloumiac, A. (2009). Nonorthogonal joint diagonalization by combining givens and hyperbolic rotations. *IEEE Transactions on Signal Processing*, *57*(6), 2222–2231.
Tichavsky, P. & Koldovsky, Z. (2012). Algorithms for nonorthogonal approximate joint block-diagonalization. In *2012 Proceedings of the 20th European signal processing conference (EUSIPCO)*, (pp. 2094–2098).
Trainini, T., & Moreau, E. (2014). A coordinate descent algorithm for complex joint diagonalization under hermitian and transpose congruences. *IEEE Transactions on Signal Processing*, *62*(19), 4974–4983.
Turing, A. M. (1948). Rounding-off errors in matrix processes. *The Quarterly Journal of Mechanics and Applied Mathematics*, *1*, 287–308.
Vandenberghe, S. B. L. (2004). *Convex optimization*. Cambridge: Cambridge University Press.
Walgate, J., Short, A. J., Hardy, L., & Vedral, V. (2000). Local distinguishability of multipartite orthogonal quantum states. *Physical Review Letter*, *85*, 4972–4975.
Westlake, J. R. (1968). *A handbook of numerical matrix inversion and solution of linear equations*. New York: Wiley.
Xu, X.-F., Feng, D.-Z., Zheng, W. X., & Zhang, H. (2010). Convolutive blind source separation based on joint block toeplitzation and block-inner diagonalization. *Signal Processing*, *90*(1), 119–133.
Yeredor, A. (2002). Non-orthogonal joint-diagonalization in the least squares sense with application in blind sources separation. *IEEE Transactions on Signal Processing*, *50*(7), 1545–1553.
Yuan, Y. H. D. Y. (1999). A nonlinear conjugate gradient method with a strong global convergence property. *SIAM Journal on Optimization*, *10*, 177–182.
Zeng, W.-J., Li, X.-L., Zhang, X.-D., & Jiang, X. (2009). An improved signal-selective direction finding algorithm using second-order cyclic statistics. In *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, Taipei, Taiwan (pp. 2141–2144).
Zhang, L., & Zhang, D. (2016). Robust visual knowledge transfer via extreme learning machine-based domain adaptation. *IEEE Transactions on Image Processing*, *25*(10), 4959–4973.
Zhang, L., Zuo, W., & Zhang, D. (2016). Lsdt: Latent sparse domain transfer learning for visual adaptation. *IEEE Transactions on Image Processing*, *25*(3), 1177–1191.

Omar Cherrak was born in Fez Morocco. He received the Bachelor degree in 2009 in the field of Electronics Telecommunications and Computer Sciences and the Master degree of Microelectronic, Telecommunications and Computer Industry Systems in 2011, both from the Université Sidi Mohamed Ben Abdellah (USMBA), Faculté des Sciences et Techniques (FST), Fez Morocco. He obtained his Ph.D. degree on March 2016 in the area of "Signal, Telecommunications, Image and Radar" from Université de Toulon and this thesis was carried out in cotutelle with USMBA. His main research interests are blind source separation, telecommunications, joint matrix decompositions, maritime surveillance system, time-frequency representation, smart grid and DoA estimation.

Hicham Ghennioui obtained the Maîtrise degree in Telecommunications from the Faculty of Sciences and Technologies (FST), Fez, Morocco, in 2002. He got the D.E.S.A. degree in Computer and Telecommunications from the Faculty of Sciences, Rabat, Morocco, in 2004 and the Ph.D. degree in engineering sciences in 2008, from Mohamed V GÇô Agdal University, Morocco, and the University of Toulon, France, respectively. From May 2008 to December 2009, he was a Research & Development Engineer at Amesys Bull, and form January 2010 to May 2011, he was a Signal/Image Researcher at Moroccan foundation for Advanced Science, Innovation and Research (MASCIR), Rabat, Morocco. Since 2011, he is an Assistant Professor at the Electrical Engineering Department of the Faculty of Sciences and Techniques, Sidi Mohamed Ben Abdellah University, Fez, Morocco. His main research interests are signal/image processing including blind sources separation, deconvolution, deblurring, time-frequency representations and cognitive radio.

Nadège Thirion-Moreau was born in Montbéliard France. She received the DEA degree in 1992 and the Ph.D. degree in 1995, both in the field of signal processing and from the Ecole Nationale Supérieure de Physique (ENSPG) Institut National Polytechnique de Grenoble (INPG), France. From 1996 to 1998, she was an Assistant Professor at the Ecole Suprieure des Procédés Electroniques et Optiques (ESPEO), Orléans, France. Since 1998, she has been with the Institut des Sciences de l'Ingénieur de Toulon et du Var (ISITV), La Valette, France, as an Assistant Professor, in the Department of Telecommunications. Her main research interests are in deterministic and statistical signal processing including array processing, blind sources separation/equalization, high-order statistics, nonstationary signals, time-frequency representations, and decision/classification.

El Hossain Abarkan was born in 1953, Nador, Morocco. He graduated the bachelor degree in physics from Mohammed V University, Rabat, Morocco, in 1978. He obtained the DEA, the Doctorate and the Ph.D. degrees from University of Languedoc, Montpellier, France, in 1979, 1981 and 1987 respectively. He got the Ph.D. degree from the University of Sidi Mohamed Ben Abdellah, Fez, Morocco, in 1992. Currently, he is a Professor in Sidi Mohamed Ben Abdellah University. From 1996 to 2014, he was the founder and the director of the Laboratory of Signals, Systems and Components, Sidi Mohamed Ben Abdellah University. His main research interests are in electronics, Modeling, characterization and CAD in integrated circuits.