A Direct Algorithm for Nonorthogonal Approximate Joint Diagonalization

Gilles Chabriel and Jean Barrère

Abstract—**While a pair of $N \times N$ matrices can almost always be exactly and simultaneously diagonalized by a generalized eigendecomposition, no exact solution exists in the case of a set with more than two matrices. This problem, termed approximate joint diagonalization (AJD), is instrumental in blind signal processing. When the set of matrices to be jointly diagonalized includes at least N linearly independent matrices, we propose a suboptimal but closed-form solution for AJD in the direct least-squares sense. The corresponding non-iterative algorithm is given the acronym DIEM (DIagonalization using Equivalent Matrices). Extensive numerical simulations show that DIEM is both fast and accurate compared to the state-of-the-art iterative AJD algorithms.**

Index Terms—**Approximate joint diagonalization (AJD), blind source separation (BSS), direct algorithm, exact joint diagonalization (EJD), non-iterative algorithm, simultaneous diagonalization.**

Notation and Definitions: The index set $\{1, \ldots, M\}$ will be denoted by $\mathcal{I}_M$ generically. Superscripts *, ^{T} and ^{H} denote respectively complex conjugation, matrix transposition and matrix complex conjugate plus transposition. Superscript † denotes the Moore–Penrose inverse matrix. The squared Frobenius norm is denoted by $\|.\|_F^2$.

Function `off(.)` sets the diagonal elements of the matrix argument to zero. Function `diag(.)` returns, when the argument is a matrix, a column-vector whose components are the diagonal entries of the matrix argument and, when the argument is a vector, a diagonal matrix whose diagonal is the vector argument. Operator `vec(.)` is a stacking of the columns of the matrix argument into a vector and $\mathtt{vec}^{-1}(.)$ stands for the inverse operator.

For any matrix $\mathbf{X}$, its (n, m) element is denoted by x_n^m, its column m by $\mathbf{X}^{(m)}$ and its row n by $\mathbf{X}_{(n)}$. The symbols $\otimes$ and $\circ$ represent the Kronecker product and the Khatri–Rao product (column-wise Kronecker product) respectively. Finally, recall that two complex matrices $\mathbf{T}$ and $\mathbf{S}$ are called "congruent" if there exists the relation $\mathbf{T} = \mathbf{QSQ}^H$, where $\mathbf{Q}$ is an invertible matrix. They are called "similar" if there exists the relation $\mathbf{T} = \mathbf{QSQ}^{-1}$, where $\mathbf{Q}$ is an invertible matrix, and they are called "equivalent" if there exists the relation $\mathbf{T} = \mathbf{QSR}$, where $\mathbf{Q}$, $\mathbf{R}$ are two invertible matrices.

Manuscript received March 16, 2010; revised July 31, 2010, January 14, 2011, and July 06, 2011; accepted October 02, 2011. Date of publication October 13, 2011; date of current version December 16, 2011. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Konstantinos I. Diamantaras.

The authors are with the Université du Sud Toulon-Var-IM2NP, CNRS-IM2NP (UMR 6242), F-83957 La Garde Cedex, France (e-mail: gilles.chabriel@im2np.fr; jean.barrere@im2np.fr).

This paper has supplementary downloadable multimedia material available at http://ieeexplore.ieee.org provided by the authors. This includes a Matlab code file (DIEM_I_DIEM.m), providing a solution with DIEM and IDIEM for AJD of random matrix sets. This material is 5.5 kB in size.

Digital Object Identifier 10.1109/TSP.2011.2171682

I. Introduction

Suppose that we are given a set $\tilde{\mathcal{T}}$ of $K \in \mathbb{N}^*$, square (possibly complex) matrices $\tilde{\mathbf{T}}_k$ of size N. These matrices, namely the *target matrices* , span a subspace, say $\mathcal{L}_{\tilde{T}}$, of the space of (complex) $N \times N$ matrices.

The problem of nonorthogonal joint diagonalization of $\tilde{\mathcal{T}}$, we consider, is to seek the "best" $\mathcal{L}_{\tilde{T}}$-fitting subspace, say $\mathcal{L}_T$, spanned by a set $\mathcal{T}$ of K matrices $\mathbf{T}_k$ sharing the following joint congruent transformation:

$$\mathbf{T}_k = \mathbf{A}\Delta_k\mathbf{A}^H,\ k \in \mathcal{I}_K. \tag{1}$$

The (complex) $N \times N$ matrices $\mathbf{\Delta}_k$ are diagonal and the (complex) $N \times N$ matrix $\mathbf{A}$ is assumed to be nonsingular but not necessarily unitary.

In practice, due to finite data sampling or noisy measurements, the joint-diagonality property is not strictly satisfied by the set $\tilde{\mathcal{T}}$ and that is why we talk about approximate joint diagonalization (AJD). A solution for matrix $\mathbf{A}$ will then depend on the criterion chosen to perform the fit.

Note that although matrix $\mathbf{A}$ and the structure imposed by (1) fully define the subspace $\mathcal{L}_T$, the reverse is not true. Let us take any matrix $\mathbf{A}'$ such that $\mathbf{A}' = \mathbf{ADP}$, where $\mathbf{D}$ is a regular diagonal matrix and where $\mathbf{P}$ is a permutation matrix. We have, $\forall k \in \mathcal{I}_K$,

$$\mathbf{T}'_k \triangleq \mathbf{A}'\Delta_k\mathbf{A}'^H = \mathbf{A}\Delta'_k\mathbf{A}^H$$

where $\Delta'_k \triangleq \mathbf{DP}\Delta_k\mathbf{P}^T\mathbf{D}^H$ is a diagonal matrix. It follows that $\{\mathbf{T}'_k\}_{k\in\mathcal{I}_K}$ is still a spanning set of the subspace $\mathcal{L}_T$ and consequently, from a given $\mathcal{L}_T$, matrix $\mathbf{A}$ is only defined up to a permutation matrix and up to a diagonal matrix. These two indeterminacies are well known in blind source separation (BSS). AJD is a fundamental tool in BSS problems where matrix $\mathbf{B} \triangleq \mathbf{A}^{-1}$ denotes the separation matrix. It has to be identified (up to the two above-mentioned inherent indeterminacies) from L snapshots of a N-vector $\tilde{\mathbf{x}}[l]$ termed the *observation vector* :

$$\tilde{\mathbf{x}}[l] = \tilde{\mathbf{A}}\tilde{\mathbf{s}}[l] + \tilde{\mathbf{n}}[l],\ l \in \mathcal{I}_L.$$

Its components are measurements of different linear mixtures (rows of matrix $\tilde{\mathbf{A}}$) of noise corrupted *source signals* . Vector $\tilde{\mathbf{s}}[l]$ (respectively, $\tilde{\mathbf{n}}[l]$) is the N-vector of the statistically independent sources (respectively, the additive noise).

An *observation space* $\mathcal{L}_{\tilde{T}}$ is spanned from target matrices often gathering estimated statistical properties of $\tilde{\mathbf{x}}[l]$. For example, one can work with estimated covariance matrices at different lags k:

$$\tilde{\mathbf{T}}_k = L^{-1} \sum_{l=1}^{L} \tilde{\mathbf{x}}[l]\tilde{\mathbf{x}}[l-k]^H,\ k \in \mathcal{I}_K. \tag{2}$$

The fitting subspace $\mathcal{L}_T$ has then to be sought as spanned by structured matrices $\mathbf{T}_k$ as in (1). Here, the matrices Δ_k denote the (theoretically) diagonal covariance matrices of the statistically independent source signals.

A widely used way to fit the subspace $\mathcal{L}_T$ is to seek a matrix $\mathbf{B}$ as a minimizer of the so called indirect least-squares (ILS) criterion:

$$\mathcal{C}_{\text{ILS}}(\mathbf{B}) = \sum_{k=1}^{K} \left\| \texttt{off}\left(\mathbf{B}\tilde{\mathbf{T}}_k\mathbf{B}^H\right)\right\|_F^2. \tag{3}$$

Numerous iterative algorithms minimizing off-diagonality have been proposed in literature for many years until recently (see e.g., [5]–[8], [10], [12], and [14]–[19]). These algorithms differ, essentially, according to the constraint chosen to avoid convergence to the trivial solution $\mathbf{B} = 0$ and/or to degenerate solutions.

One can choose for example the constraints $\mathbf{B}\mathbf{B}^H = \mathbf{I}$ or $\texttt{diag}(\mathbf{B}\tilde{\mathbf{T}}_1\mathbf{B}^H) = \mathbb{1}$, where $\mathbf{I}$ is the identity matrix and where $\mathbb{1}$ is the vector with all components set to one. $\tilde{\mathbf{T}}_1$ is any positive definite matrix picked up in the target set.

The first constraint is restrictive since it assumes that $\mathbf{B}$ is orthogonal (or unitary). It has been used for example in the Cardoso and Souloumiac's orthogonal joint diagonalization algorithm [5] and needs a spatial hard-whitening phase. It is well known that such a hard-whitening process limits the performance but, in return, it prevents degenerate solutions. The second constraint (used for example in [8], [14], and [16]) is less restrictive but allows degenerate solutions. Thus, a term proportional to $-\log|\texttt{det}(\mathbf{B})|$ can then been added to (3) in order to be minimized (see [17]). Another way (see, e.g., [8], [9], [11], and [16]) is to estimate a matrix $\mathbf{A}$ and (which is not of interest in BSS) a set of K diagonal matrices Δ_k, minimizing the so called direct least-squares (DLS) criterion:

$$\mathcal{C}_{\text{DLS}}(\mathbf{A}, \{\Delta_k\}_{k\in\mathcal{I}_K}) = \sum_{k=1}^{K} \left\| \tilde{\mathbf{T}}_k - \mathbf{A}\Delta_k\mathbf{A}^H \right\|_F^2. \tag{4}$$

The Frobenius norm being unitarily invariant, the above two criteria are equivalent when matrix $\mathbf{A}$ is unitary. Otherwise, the matrix $\mathbf{B}$ that fits (3) will not necessary be the inverse of the matrix $\mathbf{A}$ that fits (4). However, in a BSS context, the intent of AJD is that, when a "reasonable" noise level is considered, the solution for $\mathbf{B}$ in (3) or for $\mathbf{A}^{-1}$ in (4) will be as close as possible to a separating matrix, i.e.,

$$\mathbf{B}\tilde{\mathbf{A}} = \mathbf{D}\mathbf{P} \tag{5}$$

where $\mathbf{D}$ and $\mathbf{P}$ are a diagonal matrix and a permutation matrix respectively.

To the best of our knowledge, all AJD algorithms are actually iterative for $N > 2$. In this study, we show that it is possible to find a suboptimal but closed-form solution for the direct LS criterion. In addition to its robustness, we will see that the corresponding noniterative algorithm exhibits a very interesting accuracy-speed tradeoff.

The paper is organized as follows. In Section II, we propose replacing the previous $\mathcal{C}_{\text{DLS}}$ criterion applied onto $K(K \geq N)$ initial target matrices by an equivalent one, namely $\mathcal{C}_{\text{DLS}}^{(2)}$, but applied onto a set of N "representative" matrices.

In Section III, $\mathcal{C}_{\text{DLS}}^{(2)}$ is equivalently rewritten as an *under equivalence transformation* LS criterion. Steps for the DIEM algorithm that provide a suboptimal but possibly satisfactory closed-form solution for $\mathbf{A}$ minimizing $\mathcal{C}_{\text{DLS}}^{(2)}$ are described. Then, we propose an improved version for DIEM, termed IDIEM, that computes a closed-form solution for $\mathbf{A}$ satisfying the joint equivalence transformation in the LS sense. Section IV is devoted to numerical simulations for the generic AJD problem and in a BSS context. The last section gives a general conclusion.

II. Notion of "Representative" Matrix Set

In [13], Yeredor has an interesting discussion on how to build from the initial target matrices, the set of the N "more representative" matrices whose exact diagonalization gives a lower bound for the direct LS criterion minimization. Let us now recall its main derivations.

First of all, note that in a noise-free condition, the dimension of the subspace spanned by the matrices $\tilde{\mathbf{T}}_k$ cannot exceed N. Here, it is essential for the following that it is assumed equal to N. A necessary condition is obviously that at least N target matrices are available.

Use of operator $\texttt{vec}(.)$ (see [1] and [2]) in expression (4) leads to

$$\mathcal{C}_{\text{DLS}}\left(\mathbf{A}, \{\Delta_k\}_{k\in\mathcal{I}_K}\right) = \sum_{k=1}^{K} \|\text{vec}(\mathbf{T}_k) - (\mathbf{A}^* \circ \mathbf{A})\delta_k\|^2$$

where $\delta_k \triangleq \texttt{diag}(\Delta_k),\ \forall k \in \mathcal{I}_K$.

Now, defining $\tilde{\mathbf{t}}_k \triangleq \texttt{vec}(\tilde{\mathbf{T}}_k), \forall k \in \mathcal{I}_K$ and $\mathring{\mathbf{A}} \triangleq \mathbf{A}^* \circ \mathbf{A}$, we obtain a linear expression for the criterion to be minimized with respect to (w.r.t.) the "new" unknowns $\mathring{\mathbf{A}}$ and $\{\delta_k\}_{k\in\mathcal{I}_K}$:

$$\mathcal{C}_{\text{DLS}}(\mathbf{A}, \{\delta_k\}_{k\in\mathcal{I}_K}) = \sum_{k=1}^{K} \left\|\tilde{\mathbf{t}}_k - \mathring{\mathbf{A}}\delta_k\right\|^2.$$

This multi-objective optimization problem can be split into two since each kth term in the previous sum can be individually minimized in the LS sense w.r.t. δ_k using $\delta_k = \mathring{\mathbf{A}}^\dagger\tilde{\mathbf{t}}_k$. Matrix $\mathbf{A}$ has then to be sought as a minimizer of the following criterion:

$$\mathcal{C}_{\text{DLS}}^{(1)}(\mathbf{A}) = \sum_{k=1}^{K} \left\|\tilde{\mathbf{t}}_k - \mathring{\mathbf{A}}\mathring{\mathbf{A}}^\dagger\tilde{\mathbf{t}}_k\right\|^2.$$

Defining $\mathbf{P}^\perp(A) \triangleq \mathbf{I} - \mathring{\mathbf{A}}\mathring{\mathbf{A}}^\dagger$ the (Hermitian and idempotent) projection matrix onto the subspace complementary to the subspace spanned by the column of $\mathring{\mathbf{A}}$, we obtain

$$\mathcal{C}_{\text{DLS}}^{(1)}(\mathbf{A}) = \texttt{trace}\left(\mathbf{P}^\perp(A)\sum_{k=1}^{K}\tilde{\mathbf{t}}_k\tilde{\mathbf{t}}_k^H\right).$$

Let us denote by $\lambda_1, \ldots, \lambda_{N^2}$ the eigenvalues of the semi-positive definite $N^2 \times N^2$ Hermitian matrix $\tilde{\mathbf{T}} \triangleq \sum_{k=1}^{K}\tilde{\mathbf{t}}_k\tilde{\mathbf{t}}_k^H$.

Let us further denote by $\tilde{\mathbf{r}}_1, \ldots, \tilde{\mathbf{r}}_{N^2}$ the associated eigenvectors and assume that the eigenvalues are sorted in descending order. It was established in [13] that

$$\mathcal{C}^{(1)}_{\text{DLS}}(\mathbf{A}) \geq \sum_{m=N+1}^{N^2} \lambda_m. \tag{6}$$

The sum of the $N^2 - N$ smallest eigenvalues of $\tilde{\mathbf{T}}$, denoted by $\mathcal{B} \triangleq \sum_{m=N+1}^{N^2} \lambda_m$, is thus a lower bound for the direct LS criterion minimization. Yeredor shows that this bound is reached if one chooses for $\mathbf{A}$, the matrix that exactly and simultaneously diagonalizes the N matrices $\tilde{\mathbf{R}}_m \triangleq \mathtt{vec}^{-1}(\tilde{\mathbf{r}}_m)$ associated with the N largest eigenvalues, namely $\lambda_1, \ldots, \lambda_N$. Note that since $\mathtt{dim}\, \mathcal{L}_{\tilde{T}} \geq N$, these eigenvalues are necessarily nonzero. Matrices $\tilde{\mathbf{R}}_m,\ m \in \mathcal{I}_N$ are called "representative" matrices of the subspace $\mathcal{L}_{\tilde{T}}$. Obviously, such a solution does not exist in a general case and consequently the lower bound is not necessarily attainable. Nevertheless, the important point here is to conclude that an equivalent objective for the minimization of $\mathcal{C}^{(1)}_{\text{DLS}}(\mathbf{A})$ is to use a similar criterion but only applied onto the N "more representative" matrices $\tilde{\mathbf{R}}_m,\ m \in \mathcal{I}_N$. We then define the following new direct LS-fit criterion to be minimized w.r.t. $\mathbf{A}$ and $\{\Delta'_m\}_{m\in\mathcal{I}_N}$:

$$\mathcal{C}^{(2)}_{\text{DLS}}\left(\mathbf{A}, \{\Delta'_m\}_{m\in\mathcal{I}_N}\right) = \sum_{m=1}^{N} \left\|\tilde{\mathbf{R}}_m - \mathbf{A}\Delta'_m\mathbf{A}^H\right\|_F^2. \tag{7}$$

In [13], a direct solution is proposed through the exact joint diagonalization (EJD) of the two "more representative" matrices, namely $\tilde{\mathbf{R}}_1$ and $\tilde{\mathbf{R}}_2$. This can almost always be done by a generalized eigendecomposition of the matrix-pencil $(\tilde{\mathbf{R}}_1, \tilde{\mathbf{R}}_2)$. Yeredor recognizes that optimality is then not guaranteed but points out advantages to avoid the need for an iterative solution.

As will be explained in the following sections, we propose a more "balanced" closed-form solution involving the whole set of matrices $\tilde{\mathbf{R}}_m,\ m \in \mathcal{I}_N$. We first have to equivalently rewrite the previous direct LS-fit criterion (7) using *congurent matrices* to a constrained version using *equivalent matrices* .

III. AJD Using Equivalent Matrices

Let us denote by $\mathbf{T}^{(n)}_m$, a column n of the $N\times N$ matrix $\mathbf{T}_m \triangleq \mathbf{A}\Delta'_m\mathbf{A}^H$. Simple algebraic calculations (see the appendix for proof) show that

$$\mathbf{T}^{(n)}_m = \mathbf{A}\mathtt{diag}(\mathbf{A}^*_{(n)})\delta'_m,\ \forall(m,n) \in \mathcal{I}_N^2 \tag{8}$$

where $\mathbf{A}^*_{(n)}$ is the nth row of matrix $\mathbf{A}^*$ and where $\delta'_m \triangleq \mathtt{diag}(\Delta'_m)$.

Now, let us define column-wise the following $N\times N$ matrices $\mathbf{M}_n, \forall n \in \mathcal{I}_N$:

$$\mathbf{M}_n \triangleq \left[\mathbf{T}^{(n)}_1, \mathbf{T}^{(n)}_2, \ldots, \mathbf{T}^{(n)}_m, \ldots, \mathbf{T}^{(n)}_N\right].$$

According to (8), we have $\forall n \in \mathcal{I}_N$,

$$\mathbf{M}_n = \mathbf{A}\mathtt{diag}(\mathbf{A}^*_{(n)})\mathbf{C}, \tag{9}$$

with $\mathbf{C} \triangleq [\delta'_1, \delta'_2, \ldots, \delta'_m, \ldots, \delta'_N]$. Since the matrices $\mathbf{A}$ and $\mathbf{C}$ are assumed invertible, each matrix $\mathbf{M}_n$ is thus equivalent to a diagonal matrix $\mathtt{diag}(\mathbf{A}^*_{(n)})$.

Then, let us build from the N "more representative" matrices $\tilde{\mathbf{R}}_m,\ m \in \mathcal{I}_N$ a set of N matrices $\tilde{\mathbf{M}}_n$ of size N defined column-wise, $\forall n \in \mathcal{I}_N$, as

$$\tilde{\mathbf{M}}_n \triangleq \left[\tilde{\mathbf{R}}(n)_1, \tilde{\mathbf{R}}(n)_2, \ldots, \tilde{\mathbf{R}}(n)_m, \ldots, \tilde{\mathbf{R}}^{(n)}_N\right].$$

Since the Frobenius norm simply returns a sum of the squared entries of a matrix, we have according to (9)

$$\sum_{m=1}^{N} \left\|\tilde{\mathbf{R}}_m - \mathbf{A}\Delta'_m\mathbf{A}^H\right\|_F^2 = \sum_{n=1}^{N} \left\|\tilde{\mathbf{M}}_n - \mathbf{A}\mathtt{diag}(\mathbf{A}^*_{(n)})\mathbf{C}\right\|_F^2$$

and the $\mathcal{C}^{(2)}_{\text{DLS}}$ criterion can thus equivalently be rewritten as

$$\mathcal{C}^{(2)}_{\text{DLS}}(\mathbf{A}, \mathbf{C}) = \sum_{n=1}^{N} \left\|\tilde{\mathbf{M}}_n - \mathbf{A}\mathtt{diag}(\mathbf{A}^*_{(n)})\mathbf{C}\right\|_F^2. \tag{10}$$

This last result indicates that the AJD of the set of the N "more representative" matrices $\tilde{\mathbf{R}}_m, m \in \mathcal{I}_N$, using a congruence model can be replaced by the AJD of the set of matrices $\tilde{\mathbf{M}}_n,\ n \in \mathcal{I}_N$, using a constrained equivalence model where each matrix $\tilde{\mathbf{M}}_n$ must be sought as equivalent (in the LS sense) to a diagonal matrix $\mathtt{diag}(\mathbf{A}^*_{(n)})$.

A. Minimization w.r.t. $\mathbf{A}$ (*DIEM*)

The following derivations are highly inspired by the guidelines of [13] already evoked in Section II.

Using operator $\mathtt{vec}(.)$ in expression (10) yields

$$\mathcal{C}^{(2)}_{\text{DLS}}(\mathbf{A}, \mathbf{C}) = \sum_{n=1}^{N} \left\|\mathtt{vec}(\tilde{\mathbf{M}}_n) - (\mathbf{C}^T \circ \mathbf{A})a^*_n\right\|^2$$

where $a^*_n \triangleq \mathtt{diag}(\mathbf{A}^*_{(n)}),\ \forall n \in \mathcal{I}_N$.

With $\tilde{\mathbf{m}}_n \triangleq \mathtt{vec}(\tilde{\mathbf{M}}_n), \forall n \in \mathcal{I}_N$ and $\mathbf{H} \triangleq \mathbf{C}^T \circ \mathbf{A}$, we get

$$\mathcal{C}^{(2)}_{\text{DLS}}(\mathbf{A}, \mathbf{C}) = \sum_{n=1}^{N} \|\tilde{\mathbf{m}}_n - \mathbf{H}a^*_n\|^2.$$

We propose to relax the constraint on each vector a^*_n to be strictly the nth row of $\mathbf{A}^*$: we replace each a^*_n by its estimate $\hat{a}^*_n$ in the LS sense using $\hat{a}^*_n = \mathbf{H}^\dagger\tilde{\mathbf{m}}_n$. We then obtain the following new criterion we have to optimize w.r.t. $\mathbf{A}$ and $\mathbf{C}$

$$\mathcal{C}^{(3)}_{\text{DLS}}(\mathbf{A}, \mathbf{C}) = \sum_{n=1}^{N} \left\|\tilde{\mathbf{m}}_n - \mathbf{H}\mathbf{H}^\dagger\tilde{\mathbf{m}}_n\right\|^2.$$

Defining $\mathbf{P}^\perp(A, C) \triangleq \mathbf{I} - \mathbf{H}\mathbf{H}^\dagger$ and using the fact that it is Hermitian and idempotent, we get

$$\mathcal{C}^{(3)}_{\text{DLS}}(\mathbf{A}, \mathbf{C}) = \mathtt{trace}\left(\mathbf{P}^\perp(A, C)\sum_{n=1}^{N} \tilde{\mathbf{m}}_n\tilde{\mathbf{m}}_n^H\right). \tag{11}$$

Let us denote by $\tilde{\mathbf{M}}$ the rank-$N (N \times N)$ Hermitian semi-positive definite matrix $\sum_{n=1}^{N} \tilde{\mathbf{m}}_n\tilde{\mathbf{m}}_n^H$ and consider its eigendecomposition

$$\tilde{\mathbf{M}} = \sum_{n=1}^{N} \sigma_n\tilde{\mathbf{v}}_n\tilde{\mathbf{v}}_n^H, \tag{12}$$

where $\{\sigma_n\}_{n=1}^N$ are the eigenvalues and where $\{\tilde{\mathbf{v}}_n\}_{n=1}^N$ are the associated orthonormal eigenvectors. Substitution of (12) in (11) yields

$$\mathcal{C}_{\mathrm{DLS}}^{(3)}(\mathbf{A},\mathbf{C}) = \mathtt{trace}\left(\mathbf{P}^{\perp}(A,C)\sum_{n=1}^{N}\sigma_n\tilde{\mathbf{v}}_n\tilde{\mathbf{v}}_n^H\right) = \sum_{n=1}^{N}\sigma_n\left(\tilde{\mathbf{v}}_n^H\mathbf{P}^{\perp}(A,C)\tilde{\mathbf{v}}_n\right). \quad (13)$$

Projection matrix $\mathbf{P}^{\perp}(A,C)$ being semi-positive definite, complete optimization of $\mathcal{C}_{\mathrm{DLS}}^{(3)}$ is achieved *iff* each factor $\tilde{\mathbf{v}}_n^H\mathbf{P}^{\perp}(A,C)\tilde{\mathbf{v}}_n$ in the sum (13) is set to zero. This implies that there are two matrices $\mathbf{A}$ and $\mathbf{C}$ such that each vector $\tilde{\mathbf{v}}_n$ is in the range of $\mathbf{C}^T \circ \mathbf{A}$. Equivalently, the whole set of matrices $\tilde{\mathbf{V}}_n \triangleq \mathtt{vec}^{-1}(\tilde{\mathbf{v}}_n)$ must then satisfy the following joint decomposition:

$$\tilde{\mathbf{V}}_n = \mathbf{A}\Delta_n''\mathbf{C},\ \forall n \in \mathcal{I}_N \quad (14)$$

where the matrices Δ_n'' are diagonal.

Generally, such a joint decomposition does not exist for all matrices $\tilde{\mathbf{V}}_n$ but is however always possible for any two matrices. Therefore, a good enough closed-form solution for $\mathbf{A}$ can be found simply by nullifying the factors that weight the two largest eigenvalues, say σ_1 and σ_2.

When $\tilde{\mathbf{V}}_1$ (or $\tilde{\mathbf{V}}_2$) is invertible and when the diagonal entries of $\Delta_2''\Delta_1''^{-1}$ (or $\Delta_1''\Delta_2''^{-1}$) are all different (which will almost always be the case) then, according to (14), the estimate of $\mathbf{A}$ is given, up to a diagonal matrix and a permutation matrix, directly by computing the eigenvector matrix of $\tilde{\mathbf{V}}_2\tilde{\mathbf{V}}_1^{-1}$ (or $\tilde{\mathbf{V}}_1\tilde{\mathbf{V}}_2^{-1}$).

Matrices $\tilde{\mathbf{V}}_1$ and $\tilde{\mathbf{V}}_2$ being linear combinations of the whole set of matrices $\tilde{\mathbf{M}}_n$ and each $\tilde{\mathbf{M}}_n$ itself involving the whole set of matrices $\tilde{\mathbf{R}}_n$, we expect here to get a better solution than that obtained from EJD with only $\tilde{\mathbf{R}}_1$ and $\tilde{\mathbf{R}}_2$. This will be later confirmed by numerical simulations.

To summarize the discussion, we propose the following so-called DIEM (DIagonalization using Equivalent Matrices) algorithm performing AJD of a set of target matrices $\tilde{\mathbf{T}}_k, k \in \mathcal{I}_K$. Steps for DIEM are described in Table I.

TABLE I
STEPS FOR THE DIEM ALGORITHM

DIEM algorithm

Step 1: *From the K target ($N \times N$) matrices $\tilde{\mathbf{T}}_k$, build $\tilde{\mathbf{T}} \triangleq \sum_{k=1}^{K}\mathtt{vec}(\tilde{\mathbf{T}}_k)\mathtt{vec}^H(\tilde{\mathbf{T}}_k)$.*

Step 2: *Compute the eigenvectors $\tilde{\mathbf{r}}_1,\ldots,\tilde{\mathbf{r}}_N$ of $\tilde{\mathbf{T}}$ associated with the N largest eigenvalues.*

Step 3: *Build $\tilde{\mathbf{R}}_1 \triangleq \mathtt{vec}^{-1}(\tilde{\mathbf{r}}_1),\ldots,\tilde{\mathbf{R}}_N \triangleq \mathtt{vec}^{-1}(\tilde{\mathbf{r}}_N)$.*

Step 4: *Build a set of N matrices $\tilde{\mathbf{M}}_n$ defined as: $\tilde{\mathbf{M}}_n \triangleq [\tilde{\mathbf{R}}_1^{(n)},\ldots,\tilde{\mathbf{R}}_N^{(n)}], \forall n \in \mathcal{I}_N$, $\tilde{\mathbf{R}}_m^{(n)}$ being a column n of $\tilde{\mathbf{R}}_m$.*

Step 5: *From the N matrices $\tilde{\mathbf{M}}_n$, build $\tilde{\mathbf{M}} \triangleq \sum_{n=1}^{N}\mathtt{vec}(\tilde{\mathbf{M}}_n)\mathtt{vec}^H(\tilde{\mathbf{M}}_n)$.*

Step 6: *Compute the eigenvectors $\tilde{\mathbf{v}}_1$ and $\tilde{\mathbf{v}}_2$ of $\tilde{\mathbf{M}}$ associated with the two largest eigenvalues.*

Step 7: *Build $\tilde{\mathbf{V}}_1 \triangleq \mathtt{vec}^{-1}(\tilde{\mathbf{v}}_1)$ and $\tilde{\mathbf{V}}_2 \triangleq \mathtt{vec}^{-1}(\tilde{\mathbf{v}}_2)$.*

Step 8: *Compute $\mathbf{A}$ as the eigenvector matrix of $\tilde{\mathbf{V}}_2\tilde{\mathbf{V}}_1^{-1}$.*

TABLE II
STEPS FOR THE IDIEM ALGORITHM

IDIEM algorithm

Step 1: *Perform Steps 1 to 5 as described in the* DIEM *algorithm.*

Step 2: *Compute the eigenvectors $\tilde{\mathbf{v}}_1,\tilde{\mathbf{v}}_2,\ldots,\tilde{\mathbf{v}}_N$ of $\tilde{\mathbf{M}}$ associated with the N largest eigenvalues.*

Step 3: *Build $\tilde{\mathbf{V}}_1 \triangleq \mathtt{vec}^{-1}(\tilde{\mathbf{v}}_1),\ldots,\tilde{\mathbf{V}}_N \triangleq \mathtt{vec}^{-1}(\tilde{\mathbf{v}}_N)$.*

Step 4: *Compute $\mathbf{E} \triangleq \mathbf{C}^{-1}$ as the eigenvector matrix of $\tilde{\mathbf{V}}_1^{-1}\tilde{\mathbf{V}}_2$.*

Step 5: *Proceed for $m = 1,2,\ldots,N$*
Build $\mathbf{\Gamma} \triangleq [\tilde{\mathbf{V}}_1\mathbf{E}^{(m)},\ldots\tilde{\mathbf{V}}_N\mathbf{E}^{(m)}]$.
Compute $\mathbf{A}^{(m)}$ as the eigenvector associated with the largest eigenvalue of matrix $\mathbf{\Gamma}\mathbf{\Gamma}^H$.

B. *Minimization w.r.t.* $\mathbf{C}$ *Then* $\mathbf{A}$ (*IDIEM*)

One advantage of $\mathcal{C}_{\mathrm{DLS}}^{(3)}(\mathbf{A},\mathbf{C})$ compared to $\mathcal{C}_{\mathrm{DLS}}^{(1)}(\mathbf{A})$ is that it exhibits more degrees of freedom through the matrix $\mathbf{C}$. Another major advantage is that the equations in (14) now depend linearly on $\mathbf{A}$. This allows for a given $\mathbf{C}$, to find column-wise a direct solution for $\mathbf{A}$ satisfying (14) in the LS sense.

As for matrix $\mathbf{A}$ in DIEM, we first choose a matrix $\mathbf{C}$ that satisfies (14) only for $n = 1, 2$. Such a matrix can almost always be found from the eigendecomposition of $\tilde{\mathbf{V}}_1^{-1}\tilde{\mathbf{V}}_2$. Now, let us denote by $\mathbf{E}^{(m)}$ a column m of $\mathbf{C}^{-1}$. From (14), we obtain directly

$$\tilde{\mathbf{V}}_n\mathbf{E}^{(m)} = \delta_{nm}''^{m}\mathbf{A}^{(m)},\ \forall n \in \mathcal{I}_N \quad (15)$$

where $\delta_{nm}''^{m}$ is the (m,m) element of Δ_n'' and where $\mathbf{A}^{(m)}$ is the mth column of the matrix $\mathbf{A}$ we seek.

In practice, there is no vector $\mathbf{A}^{(m)}$ satisfying strictly (15). However, a solution in the LS sense can always be found and it is well known[1] that it is given by the eigenvector associated with the largest eigenvalue of the matrix $\Gamma\Gamma^H$, where

$$\Gamma \triangleq \left[\tilde{\mathbf{V}}_1\mathbf{E}^{(m)},\ldots\tilde{\mathbf{V}}_N\mathbf{E}^{(m)}\right].$$

Estimation of the complete matrix $\mathbf{A}$ is obtained column-wise using this procedure with $m = 1, 2, \ldots, N$. Steps for the corresponding algorithm termed IDIEM (Improved DIEM) are summarized in the following Table II.

C. *Concluding Remarks*

The two previous joint diagonalization algorithms seek to minimize the direct fitting criterion $C_{\mathrm{DLS}}^{(1)}$ through a sequence of successive approximations:

- the minimization of $C_{\mathrm{DLS}}^{(2)}$ as a substitute to minimizing $C_{\mathrm{DLS}}^{(1)}$ is almost always an approximation;

[1]This approach is often referred to as principal component analysis (PCA).

- the minimization of $C_{\mathrm{DLS}}^{(3)}$ as a substitute to minimizing $C_{\mathrm{DLS}}^{(2)}$ is almost always an approximation, too (due to the relaxation of the structural constraint);
- the proposed solutions–either by applying (14) only to the two leading matrices (DIEM), or by solving a LS problem in fitting these relations (IDIEM)—almost always serve merely as approximate minimizers of $C_{\mathrm{DLS}}^{(3)}$.

Although the proposed solutions are only approximate, they have the advantage of being closed-form, fast solutions. As it will be shown in the following section, these algorithms also yield competitive performance compared to other (non-approximate) methods.

IV. Numerical Simulations

In this section, we investigate the accuracy and the CPU time cost of several AJD algorithms versus (I)DIEM (for both DIEM and IDIEM) via computer simulations (Matlab® code). Results are provided, firstly, for the generic AJD problem then in a BSS context.

In order to allow comparison between different simulation scenarios, the resulting off-diagonality error $\mathcal{C}_{\mathrm{ILS}}(\mathbf{B})(3)$, the resulting LS fit error $\mathcal{C}_{\mathrm{DLS}}(\mathbf{A},\Delta_k)(4)$ and the lower bound $\mathcal{B}(6)$ have to be scaled by the total number of measurements in the target set. We thus define the normalized fit-error by $\bar{\mathcal{C}}_{\mathrm{ILS}} \triangleq \frac{\mathcal{C}_{\mathrm{ILS}}}{KN^2}$, the normalized off-diagonality error by $\bar{\mathcal{C}}_{\mathrm{DLS}} \triangleq \frac{\mathcal{C}_{\mathrm{DLS}}}{KN^2}$ and the normalized lower bound by $\bar{\mathcal{B}} \triangleq \frac{\mathcal{B}}{KN^2}$.

In addition, the final value of $\bar{\mathcal{C}}_{\mathrm{ILS}}(\mathbf{B})$ depending on the norm of $\mathbf{B}$, we have to scale $\mathbf{B}$ to allow comparison between the different $\mathcal{C}_{\mathrm{ILS}}$-based algorithms. We choose to impose $\mathtt{diag}(\mathtt{diag}(\mathbf{B}\tilde{\mathbf{T}}_1\mathbf{B}^H)) = \mathbf{I}$, where $\mathbf{I}$ is the identity matrix and $\tilde{\mathbf{T}}_1$ is the first target matrix assumed to be positive definite.

A. Performance Comparison for Generic AJD

We randomly build the K target matrices $\tilde{\mathbf{T}}_k$ of size $N(N \leq K)$ as

$$\tilde{\mathbf{T}}_k = \tilde{\mathbf{A}}\mathbf{D}_k\tilde{\mathbf{A}}^H + \beta_k \Pi_k.$$

The entries of the mixing matrix $\tilde{\mathbf{A}}$ are complex, independent and normally distributed, $\mathcal{N}(0,1)$, for all the experiments except the last one where they are drawn from a uniform distribution, $\mathcal{U}(0,1)$. The entries of diagonal matrices $\mathbf{D}_k$ are independent and normally distributed but, it will depend on individual experiment, they are taken as real or complex. The perturbation matrices Π_k are Hermitian positive-definite complex matrices. In such a scenario, a sufficiently large K assures that the dimension of the signal part of $\mathcal{L}_{\tilde{T}}$ (the subspace spanned by the matrices $\tilde{\mathbf{A}}\mathbf{D}_k\tilde{\mathbf{A}}^H$) is N. Let us recall that this property is essential for the two (I)DIEM algorithms.

The coefficients β_k are chosen such that, $\forall k \in \mathcal{I}_K$,

$$10\log_{10}\frac{\|\tilde{\mathbf{A}}\mathbf{D}_k\tilde{\mathbf{A}}^H\|_F^2}{\|\beta_k\Pi_k\|_F^2} = \rho,$$

where ρ defines a signal-to-noise ratio (SNR) expressed in dB.

In a first experiment, we compare the performance in terms of subspace fitting of the only three noniterative existing AJD algorithms, namely DIEM, IDIEM and the EJD solution proposed in [13]. We take $N = 5$ and $K = 20$. The matrices $\{D_k\}_{k\in\mathcal{I}_K}$ are complex. For each Monte Carlo simulation, the 2-norm condition number of $\tilde{\mathbf{A}}$ (the ratio of the largest singular value of $\tilde{\mathbf{A}}$ to the smallest) is less than or equal to 5 ($\mathtt{cond}(\tilde{\mathbf{A}}) \leq 5$).

Mean ($\bar{\mathrm{m}}$) and standard deviation (std) of the normalized fit-error over 300 independent trials are summarized in Table III for different noise levels. For each Monte Carlo simulation, mean and variance of the normalized fitting lower bound $\bar{\mathcal{B}}(6)$ are stated in the first column.

TABLE III
Fitting Performance Versus the Noise Level $N = 5, K = 15$, $\mathtt{cond}(\tilde{\mathbf{A}}) \leq 5$

	$\bar{\mathcal{C}}_{DLS}$ (dB)		
	EJD	DIEM	IDIEM
$\rho = 20$; $\bar{\mathcal{B}}$ { $\bar{\mathrm{m}}$: -38.3 dB, std: 1.1 dB	$\bar{\mathrm{m}}$: -26.2 std: 7.2	$\bar{\mathrm{m}}$: -34.4 std: 4.2	$\bar{\mathrm{m}}$: -36.1 std: 1.4
$\rho = 30$; $\bar{\mathcal{B}}$ { $\bar{\mathrm{m}}$: -48.2 dB, std: 1.0 dB	$\bar{\mathrm{m}}$: -32.9 std: 12.2	$\bar{\mathrm{m}}$: -44.2 std: 5.9	$\bar{\mathrm{m}}$: -45.9 std: 3.5
$\rho = 40$; $\bar{\mathcal{B}}$ { $\bar{\mathrm{m}}$: -58.2 dB, std: 1.0 dB	$\bar{\mathrm{m}}$: -40.9 std: 16.7	$\bar{\mathrm{m}}$: -54.1 std: 6.5	$\bar{\mathrm{m}}$: -55.9 std: 3.9
$\rho = 60$; $\bar{\mathcal{B}}$ { $\bar{\mathrm{m}}$: -78.2 dB, std: 1.0 dB	$\bar{\mathrm{m}}$: -60.2 std: 19.8	$\bar{\mathrm{m}}$: -74.2 std: 5.6	$\bar{\mathrm{m}}$: -76.0 std: 2.0

TABLE IV
Fitting Performance Versus the Number K of Target Matrices. $N = 5, \rho = 20$ dB, $\mathtt{cond}(\tilde{\mathbf{A}}) \leq 5$

	$\bar{\mathcal{C}}_{DLS}$ (dB)		
	EJD	DIEM	IDIEM
$K = 6$; $\bar{\mathcal{B}}$ { $\bar{\mathrm{m}}$: -45.0 dB, std: 3.5 dB	$\bar{\mathrm{m}}$: -27.5 std: 7.6	$\bar{\mathrm{m}}$: -29.6 std: 3.7	$\bar{\mathrm{m}}$: -32.14 std: 7.6
$K = 10$; $\bar{\mathcal{B}}$ { $\bar{\mathrm{m}}$: -39.6 dB, std: 1.6 dB	$\bar{\mathrm{m}}$: -26.7 std: 7.6	$\bar{\mathrm{m}}$: -33.5 std: 4.7	$\bar{\mathrm{m}}$: -36.6 std: 3.0
$K = 20$; $\bar{\mathcal{B}}$ { $\bar{\mathrm{m}}$: -37.8 dB, std: 0.92 dB	$\bar{\mathrm{m}}$: -26.5 std: 7.1	$\bar{\mathrm{m}}$: -34.8 std: 3.7	$\bar{\mathrm{m}}$: -36.2 std: 1.4
$K = 30$; $\bar{\mathcal{B}}$ { $\bar{\mathrm{m}}$: -37.3 dB, std: 0.7 dB	$\bar{\mathrm{m}}$: -26.7 std: 7.56	$\bar{\mathrm{m}}$: -35.3 std: 2.8	$\bar{\mathrm{m}}$: -36.3 std: 1.4
$K = 50$; $\bar{\mathcal{B}}$ { $\bar{\mathrm{m}}$: -36.9 dB, std: 0.5 dB	$\bar{\mathrm{m}}$: -26.7 std: 7.4	$\bar{\mathrm{m}}$: -35.9 std: 2.0	$\bar{\mathrm{m}}$: -36.4 std: 0.56

As expected, we observe that a lower fitting error is provided by DIEM and IDIEM than with the EJD solution. Moreover, results obtained with IDIEM remain close to the normalized lower bound $\bar{\mathcal{B}}$.

In the following second experiment, we still take $N = 5$, fix the noise level ρ to 20 dB and force $\mathtt{cond}(\tilde{\mathbf{A}}) \leq 5$. Table IV gives the fitting performance of the three algorithms over 300 independent trials as a function of the number K of target matrices. These results have to be compared to the normalized lower bound $\bar{\mathcal{B}}(6)$ given in the first column.

We find that, instead of EJD, DIEM and more particularly IDIEM reach the normalized fitting lower bound $\bar{\mathcal{B}}$ in terms of mean and variance as the number of available target matrices increases.

In a third experiment DIEM and IDIEM are compared to different existing iterative algorithms based on either the direct criterion (AC–DC [11]), or the indirect one (QDIAG

TABLE V
PERFORMANCE AND CPU TIME COST. $N = 5, K = 15, \rho = 20$ dB, $\texttt{cond}(\tilde{\mathbf{A}}) \leq 5$. $\bar{\mathcal{B}}(\bar{m} : -38.3$ dB, $\text{std} : 1.1$ dB$)$

	$\overline{\mathcal{C}}_{DLS}$ (dB)	$\overline{\mathcal{C}}_{ILS}$ (dB)	PI (dB)	CPU time
EJD	$\overline{\text{m}}$: -26.2 std: 7.2		$\overline{\text{m}}$: -11.9 std: 9.5	1
DIEM	$\overline{\text{m}}$: -34.4 std: 4.2		$\overline{\text{m}}$: -25.8 std: 6.7	1.24
IDIEM	$\overline{\text{m}}$: -36.1 std: 1.38		$\overline{\text{m}}$: -27.0 std: 5.3	1.9
AC-DC	$\overline{\text{m}}$: -37.0 std: 0.77		$\overline{\text{m}}$: -31.9 std: 3.4	291
SDIAG		$\overline{\text{m}}$: -20.3 std: 3.21	$\overline{\text{m}}$: -24.82 std: 5.12	224
QDIAG		$\overline{\text{m}}$: -20.2 std: 3.2	$\overline{\text{m}}$: -24.8 std: 5.25	60
UWEDGE		$\overline{\text{m}}$: -20.02 std: 3.31	$\overline{\text{m}}$: -26.1 std: 5.1	8.0
OJD		$\overline{\text{m}}$: -18.8 std: 3.05	$\overline{\text{m}}$: -21.93 std: 4.4	32

TABLE VI
PERFORMANCE AND CPU TIME COST. $N = 5, K = 15, \rho = 20$ dB. $\bar{\mathcal{B}}(\bar{m} : -38.1$ dB, $\text{std} : 1.2$ dB$)$

	$\overline{\mathcal{C}}_{DLS}$ (dB)	$\overline{\mathcal{C}}_{ILS}$ (dB)	PI (dB)	CPU time
EJD	$\overline{\text{m}}$: -24.7 std: 5.9		$\overline{\text{m}}$: -7.9 std: 8.0	1
DIEM	$\overline{\text{m}}$: -32.2 std: 6.6		$\overline{\text{m}}$: -20.5 std: 10.9	1.29
IDIEM	$\overline{\text{m}}$: -33.7 std: 4.9		$\overline{\text{m}}$: -17.0 std: 14.3	2.0
AC-DC	$\overline{\text{m}}$: -36.7 std: 1.6		$\overline{\text{m}}$: -26.3 std: 11.0	791
SDIAG		$\overline{\text{m}}$: -15.7 std: 4.7	$\overline{\text{m}}$: -14.7 std: 10.7	440
QDIAG		$\overline{\text{m}}$: -15.8 std: 4.0	$\overline{\text{m}}$: -14.3 std: 11.1	115
UWEDGE		$\overline{\text{m}}$: -15.4 std: 4.4	$\overline{\text{m}}$: -15.3 std: 11.8	16
OJD		$\overline{\text{m}}$: -14.3 std: 4.3	$\overline{\text{m}}$: -13.6 std: 10.3	51

[8], [14]–[16], SDIAG [18], UWEDGE [19], OJD[2] [5]). Since AC–DC and SDIAG cannot accommodate non-Hermitian matrices, $\mathbf{D}_k$ are taken as real. Additionally, in order to assure a proper hard-whitening for OJD and a proper scaling for $\mathcal{C}_{\text{ILS}}$-based algorithms, we force $\tilde{\mathbf{T}}_1$ to be a positive-definite matrix choosing $\mathbf{D}_1$ with positive diagonal entries.

Besides the two previous estimation errors (fitting and off-diagonality) corresponding to each algorithm, the BSS classical performance index (PI) for the obtained overall mixing-unmixing matrix $\mathbf{G} \triangleq \mathbf{B}\tilde{\mathbf{A}}$ will be given for all the methods. This index is classically defined as

$$\text{PI}(\mathbf{G}) \triangleq \frac{1}{N(N-1)} \left[\sum_{i \in \mathcal{I}_N} \left(\sum_{j \in \mathcal{I}_N} \frac{|G_i^j|^2}{\max_\ell |G_i^\ell|^2} - 1 \right) + \sum_{j \in \mathcal{I}_N} \left(\sum_{i \in \mathcal{I}_N} \frac{|G_i^j|^2}{\max_\ell |G_\ell^j|^2} - 1 \right) \right].$$

This non-negative index is zero if $\mathbf{G}$ satisfies (5) and a small value indicates proximity to a separating solution.

Tables V and VI provide performance results for $N = 5$, $K = 15$ and $\rho = 20$ dB over 300 independent trials.

[2]Pre-whitening followed by the Givens rotations based orthogonal joint diagonalization algorithm used in the BSS JADE algorithm [3].

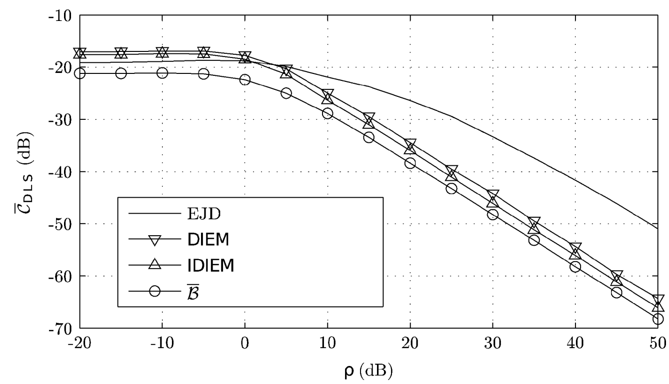

Fig. 1. $\bar{\mathcal{C}}_{\text{DLS}}$ as a function of SNR. $N = 5, K = 15, \texttt{cond}(\tilde{\mathbf{A}}) \leq 5$.

In Table V, matrix $\tilde{\mathbf{A}}$ is randomly chosen such that $\texttt{cond}(\tilde{\mathbf{A}}) \leq 5$ and in Table VI, $\texttt{cond}(\tilde{\mathbf{A}})$ is left free. The average relative CPU time cost of each algorithm is shown in the last column, the CPU load of EJD being taken as a reference. In order to avoid convergence problems and decrease the number of iterations, all iterative algorithms are initialized by the EJD solution.

Both DIEM and IDIEM algorithms behave particularly well in these two simulation scenarios since they have undoubtedly the best accuracy-speed tradeoff. In terms of BSS performance (PI), the best results are for AC–DC with the heaviest CPU load. The orthogonal joint diagonalization algorithm OJD provides the second lowest performance. This illustrates the fact that the hard-whitening phase imposes a limit on the reachable performance [4].

In the better conditioned case (Table V), DIEM has a comparable accuracy than UWEDGE, SDIAG and QDIAG while being faster. IDIEM exhibits a performance slightly better (about 1 or 2 dB) than its competitors (except for AC–DC) with still a lower CPU load.

When the condition number of $\tilde{\mathbf{A}}$ is left free (Table VI), DIEM and AC–DC maintain a reasonable accuracy in losing only 5 dB while the others lose 10 dB. The CPU load increases for all the iterative algorithms while it remains obviously unchanged with the proposed direct ones.

Then, we investigate the performance of OJD and the fastest algorithms EJD, DIEM, IDIEM, UWEDGE for a wide range of signal-to-noise ratios. We still take $N = 5$, $K = 15$ and we initially consider the favorable case where $\texttt{cond}(\tilde{\mathbf{A}}) \leq 5$. For each SNR's value, the resulting LS fit error $\bar{\mathcal{C}}_{\text{DLS}}$ is shown on Fig. 1 over 300 Monte Carlo experiments. It has to be compared to the corresponding theoretical lower bound $\bar{\mathcal{B}}$. Fig. 2 presents the obtained performance index PI and the averaged relative (to EJD) CPU time cost can be found in the legend.

In Fig. 1, we see that for SNRs higher than about 0 dB, the fit errors obtained with DIEM and IDIEM are lower than those obtained with EJD. For SNRs higher than 10 dB, the proposed algorithms provide a performance close to the theoretical lower bound (2 dB for IDIEM and 4 dB for DIEM). In Fig. 2, we observe that for SNRs lower than 10 dB all the methods perform really bad. For higher SNRs, the PIs obtained with DIEM, IDIEM and UWEDGE are very close and always better than with OJD.

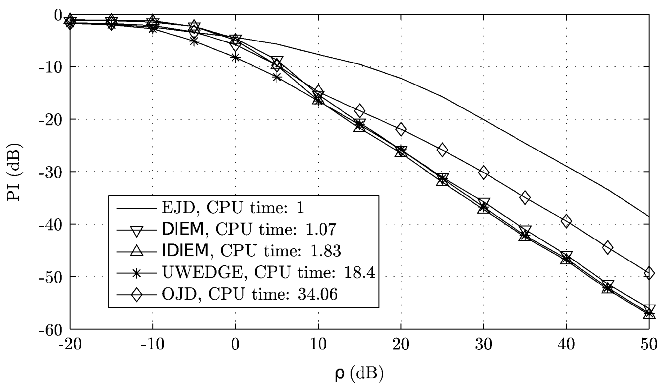

Fig. 2. PI as a function of SNR. $N = 5, K = 15, \texttt{cond}(\tilde{\mathbf{A}}) \leq 5$.

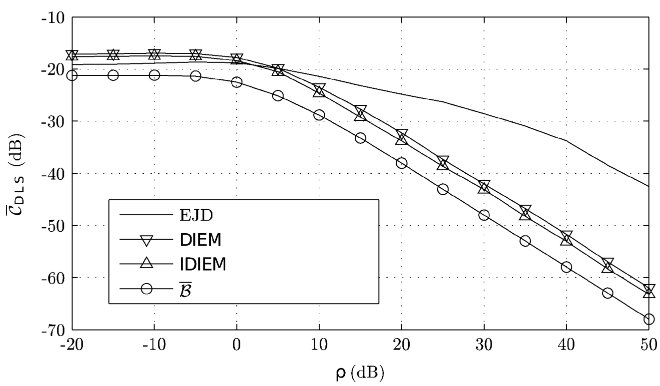

Fig. 3. $\bar{\mathcal{C}}_{\mathrm{DLS}}$ as a function of SNR. $N = 5, K = 15$, free $\texttt{cond}(\tilde{\mathbf{A}})$.

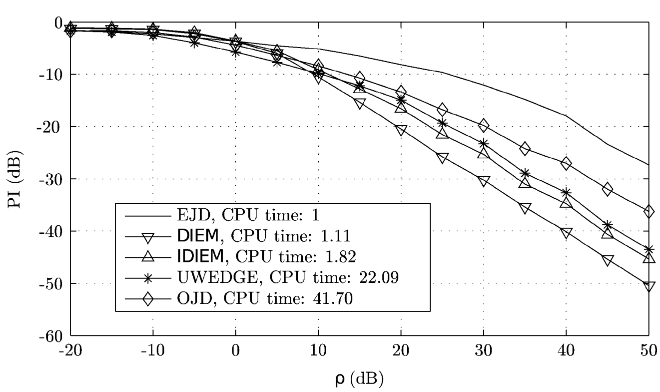

Fig. 4. PI as a function of SNR. $N = 5, K = 15$, free $\texttt{cond}(\tilde{\mathbf{A}})$.

Figs. 3 and 4 present the fitting error and the BSS performance obtained when the condition number of $\tilde{\mathbf{A}}$ is now left free.

Fig. 3 still shows that for SNRs higher than about 0 dB, a best fit is obtained with DIEM and IDIEM than with EJD. For SNRs higher than 10 dB and compared to the case where $\texttt{cond}(\tilde{\mathbf{A}}) \leq 5$ (Fig. 1), we notice a performance loss of about 3 dB with (I)DIEM and a performance loss of about 7 dB with EJD. In terms of PI (Fig. 4), we notice a performance loss for all the methods versus Fig. 2: a loss of more than 10 dB for EJD, OJD, and UWEGDE, only 5 dB (respectively, 7 dB) for DIEM (respectively, IDIEM). This confirms that IDIEM and especially DIEM are robust to ill-conditioned cases.

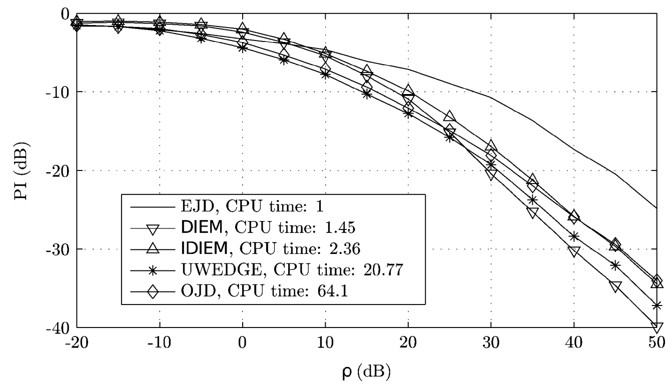

Fig. 5. PI as a function of SNR. $N = 5, K = 6$, free $\texttt{cond}(\tilde{\mathbf{A}})$.

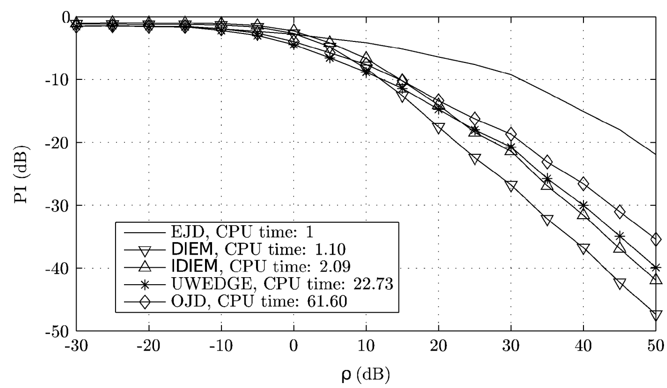

Fig. 6. PI versus SNR. $N = 5, K = 6$, free $\texttt{cond}(\tilde{\mathbf{A}})$, low $\texttt{cond}(\mathbf{C})$.

Fig. 5 presents the performance index PI obtained when the number of available target matrices is reduced to $K = 6$. The condition number of $\tilde{\mathbf{A}}$ is still left free.

Fig. 5 shows that for SNRs lower than 25 dB, all the methods provide a bad BSS performance. For higher SNRs, the best performance is here obtained with DIEM. IDIEM only reaches the performance of OJD for SNRs higher than about 40 dB. This behavior is due to the sensitivity of DIEM and especially IDIEM to the bad condition number of matrix $\mathbf{C}$ (see (9)). This occurs when at least two columns of $\mathbf{C}$ are close and we will say that here the signal subspace diversity is not sufficient. The probability to obtain a bad condition number for $\mathbf{C}$ (a low subspace diversity) increases as the number of target matrices decreases.

To illustrate our discussion, we propose an new experiment (see Fig. 6) where the condition number of matrix $\mathbf{C}$ is constrained to be low. For that, we take $\Delta_1 = \mathbf{I}$ and force the vectors $\delta_k \triangleq \texttt{diag}(\Delta_k),\ k = \{2\ldots,6\}$, to be random orthonormal bases of $\mathbb{R}^5$.

We can see in Fig. 6 the performance improvement obtained with our two algorithms compared to the previous case. Now, in terms of CPU load, (I)DIEM algorithms remain the most competitive for all the previous experiments.

The algorithms OJD, EJD, DIEM, IDIEM, and UWEDGE are now compared for problems of larger dimension. The target matrices are generated as in the previous experiments. The condition number of the mixing matrix $\tilde{\mathbf{A}}$ is left free. UWEDGE and OJD only use the N "more representative" matrices $\tilde{\mathbf{R}}_n$ and the positive definite target matrix $\tilde{\mathbf{T}}_1$ for scaling and whitening.

TABLE VII

PERFORMANCE AND CPU TIME COST. $\rho = 40$ dB, $N = 20, K = 30$, $\bar{\mathcal{B}}(\bar{m}: -70.4\text{ dB}, \mathrm{std}: 1.16\text{ dB})$

	$\bar{\mathcal{C}}_{DLS}$ (dB)	$\bar{\mathcal{C}}_{ILS}$ (dB)	PI (dB)	CPU time
EJD	$\bar{m}$: -32.0 std: 2.4		$\bar{m}$: -11.8 std: 3.2	1
DIEM	$\bar{m}$: -43.9 std: 8.2		$\bar{m}$: -27.7 std: 9.4	2.1
IDIEM	$\bar{m}$: -55.7 std: 9.1		$\bar{m}$: -26.7 std: 11.9	5.3
UWEDGE		$\bar{m}$: -23.5 std: 6.0	$\bar{m}$: -26.7 std: 10.3	26.1
OJD		$\bar{m}$: -21.6 std: 2.6	$\bar{m}$: -23.1 std: 4.1	397

TABLE VIII

PERFORMANCE AND CPU TIME COST. $\rho = 40$ dB, $N = 100, K = 160$, $\bar{\mathcal{B}}(\bar{m}: -84.2\text{ dB}, \mathrm{std}: 0.22\text{ dB})$

	$\bar{\mathcal{C}}_{DLS}$ (dB)	$\bar{\mathcal{C}}_{ILS}$ (dB)	PI (dB)	CPU time
EJD	$\bar{m}$: -42.0 std: 0.4		$\bar{m}$: -14.8 std: 1.1	1
DIEM	$\bar{m}$: -50.4 std: 6.2		$\bar{m}$: -27.7 std: 8.8	14.1
IDIEM	$\bar{m}$: -65.3 std: 18.0		$\bar{m}$: -31.6 std: 17.0	25.0
UWEDGE		$\bar{m}$: -27.5 std: 8.7	$\bar{m}$: -32.0 std: 10.1	272.1
OJD		$\bar{m}$: -26.4 std: 0.4	$\bar{m}$: -27.9 std: 2.1	3676

TABLE IX

PERFORMANCE AND CPU TIME COST. $\tilde{\mathbf{A}} \sim \mathcal{U}(0, 1)$, $\rho = 40$ dB, $N = 100$, $K = 160$, $\bar{\mathcal{B}}(\bar{m}: -88.3\text{ dB}, \mathrm{std}: 0.45\text{ dB})$

	$\bar{\mathcal{C}}_{DLS}$ (dB)	$\bar{\mathcal{C}}_{ILS}$ (dB)	PI (dB)	CPU time
EJD	$\bar{m}$: -46.1 std: 0.55		$\bar{m}$: -13.6 std: 0.77	1
DIEM	$\bar{m}$: -56.0 std: 6.38		$\bar{m}$: -29.0 std: 7.8	14.1
IDIEM	$\bar{m}$: -68.5 std: 11.7		$\bar{m}$: -31.0 std: 11.1	25.0
UWEDGE		$\bar{m}$: -26.5 std: 9.6	$\bar{m}$: -28.0 std: 9.1	365
OJD		$\bar{m}$: -11.1 std: 0.35	$\bar{m}$: -23.1 std: 0.45	3750

The matrices $\tilde{\mathbf{R}}_n$ are computed, from the initial target set, using steps 1 to 3 in Table I. Table VII (respectively, Table VIII) provides performance results for $N = 20$ (respectively, $N = 100$), $\rho = 40$ dB and $K = 30$ (respectively, $K = 160$) over 300 independent trials (respectively, 100 independent trials).

One can see that, even for problems of larger dimension, DIEM and IDIEM remain faster than its competitors while having a interesting accuracy. One can note here that the orthogonal joint diagonalization algorithm, OJD, provides surprisingly a good performance. This behavior is due to the fact that the mixing matrix $\tilde{\mathbf{A}}$ tends to be an orthogonal matrix as N increases since its entries are drawn independently from a zero-mean unit-variance distribution. This is confirmed by the following last experiment where the entries of the mixing matrix $\tilde{\mathbf{A}}$ are now drawn independently from a uniform distribution between 0 and 1, $\mathcal{U}(0, 1)$. In this case, OJD returns a lower performance while (I)DIEM and UWEDGE still maintain good results (see Table IX).

TABLE X

BSS PERFORMANCE AND CPU TIME COST. $\rho = 30$ dB, $N = 4, K = 12$, $\bar{\mathcal{B}}(\bar{m}: -42.6\text{ dB}, \mathrm{std}: 2.7\text{ dB})$

	$\bar{\mathcal{C}}_{DLS}$ (dB)	$\bar{\mathcal{C}}_{ILS}$ (dB)	PI (dB)	CPU time
EJD	$\bar{m}$: -33.0 std: 11.3		$\bar{m}$: -19.3 std: 17.6	1
DIEM	$\bar{m}$: -38.6 std: 7.0		$\bar{m}$: -28.2 std: 10.0	1.01
IDIEM	$\bar{m}$: -40.9 std: 4.1		$\bar{m}$: -29.1 std: 11.7	1.3
AC-DC	$\bar{m}$: -41.9 std: 2.9		$\bar{m}$: -30.03 std: 12.3	323
SDIAG		$\bar{m}$: -25.8 std: 9.1	$\bar{m}$: -24.8 std: 17.2	125
QDIAG		$\bar{m}$: -30.5 std: 6.3	$\bar{m}$: -25.7 std: 20.4	61
UWEDGE		$\bar{m}$: -25.6 std: 9.9	$\bar{m}$: -25.15 std: 17.1	8.7
OJD		$\bar{m}$: -18.5 std: 12.2	$\bar{m}$: -15.2 std: 17.1	13

B. (I)DIEM Performance for Second-Order BSS Applications

In the following simulation, four RF (10 cm wavelength) carrier waves modulated in amplitude by audio signals (speech and music) impinge on a four-element uniform linear array with half-wavelength sensor spacing in the presence of stationary complex white Gaussian noise. The audio sources are naturally differently colored i.e., they each have a different power spectral density. This property is essential in order to assure proper dimension of the signal part of $\mathcal{L}_{\tilde{T}}$ (here equal to 4).

For each Monte Carlo simulation, the direction-of-arrival (DOA) of each wave is randomly chosen in the range $[-60^\circ, 60^\circ]$. The difference between any two DOA is chosen as greater than or equal to 5°. The sample rate is set to 3000 Hz and we consider snapshots of size 50 000. The target matrix set is composed of $K = 12$ estimated spatial covariance matrices taken at lags $k = \{0, 5, 10 \ldots, 55\}$ (see (2)). For all the algorithms, the mean overall rejection level (PI) and the resulting errors for the two criteria are estimated by averaging 300 independent trials. Results can be found in Table VI for a SNR $\rho = 30$ dB. One again, the mean CPU time cost of each algorithm is given in the last column (the CPU time cost of EJD still being taken as reference) and the EJD solution is used to initialize each iterative algorithm.

This last simulation validates our approach in a BSS context, since we find that (I)DIEM exhibits better results both in terms of performance and CPU load than its competitors. In particular, it has fit error comparable with AC–DC while outperforming it in terms of CPU load.

V. CONCLUSION

In this paper, we addressed the problem of nonorthogonal AJD of a set of N-by-N matrices. We considered the particular case where the target matrix set spans a subspace of size at least equal to N which is usually not a restrictive requirement in a BSS context, except maybe when a large number of sources is considered. In this case, we propose two approximate but closed-form solutions for the direct least-squares criterion optimization.

The two corresponding direct algorithms DIEM and IDIEM outperform all the tested iterative AJD algorithms in terms of computational load while maintaining a good accuracy even for ill-conditioned cases regarding the mixing matrix. When the number of available matrices is sufficient, a better performance is even obtained than with most of the $\mathcal{C}_{\mathrm{ILS}}$-based algorithms we tested. Moreover, the proposed algorithms are not subject to any structural constraint on neither the mixing matrix (orthogonal, unitary), nor the target matrices (symmetric, Hermitian, positive definite, etc.). The main drawback of our approach is that the performance of the two proposed algorithms decreases when the subspace diversity provided by the target matrices is not sufficient. This may occur for a bad SNR, when the number of available target matrices is too low.

APPENDIX

Consider an $N \times N$ matrix $\mathbf{T}_m$ satisfying the following decomposition:

$$\mathbf{T}_m = \mathbf{A}\boldsymbol{\Delta}_m\mathbf{A}^H$$

where $\boldsymbol{\Delta}_m$ is a diagonal matrix.

The (i,n) element of $\mathbf{T}_m$ is expressed, $\forall (i,n) \in \mathcal{I}_N^2$ as

$$t_{m}{}_{i}^{n} = \sum_{k=1}^{N} a_i^k a^{*}{}_{n}^{k} \delta_{m}{}_{k}^{k}$$

where $a^{*}{}_{i}^{k}$ and $\delta_{m}{}_{k}^{k}$ denote the (i,k) element of $\mathbf{A}^*$ and the (k,k) element of $\boldsymbol{\Delta}_m$ respectively.

This can be rewritten in vector-matrix notation as

$$t_{m}{}_{i}^{n} = [a_i^1, \ldots, a_i^N] \begin{bmatrix} a^{*}{}_{n}^{1} & 0 & \cdots & 0 \\ 0 & a^{*}{}_{n}^{2} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & a^{*}{}_{n}^{N} \end{bmatrix} \begin{bmatrix} \delta_{m}{}_{1}^{1} \\ \vdots \\ \vdots \\ \delta_{m}{}_{N}^{N} \end{bmatrix}.$$

Consequently, a column n of $\mathbf{T}_m$ yields

$$\begin{bmatrix} t_{m}{}_{1}^{n} \\ \vdots \\ \vdots \\ t_{m}{}_{N}^{n} \end{bmatrix} = \mathbf{A} \begin{bmatrix} a^{*}{}_{n}^{1} & 0 & \cdots & 0 \\ 0 & a^{*}{}_{n}^{2} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & a^{*}{}_{n}^{N} \end{bmatrix} \begin{bmatrix} \delta_{m}{}_{1}^{1} \\ \vdots \\ \vdots \\ \delta_{m}{}_{N}^{N} \end{bmatrix}$$
$$= \mathbf{A}\mathtt{diag}(\mathbf{A}^{*}_{(n)})\mathtt{diag}(\Delta_m).$$

■

REFERENCES

[1] J. W. Brewer, "Kronecker products and matrix calculus in system theory," *IEEE Trans. Circuits Syst.*, vol. 25, pp. 772–781, Sep. 1978.

[2] R. Horn and C.-R. Johnson, *Matrix Analysis*. Cambridge, U.K.: Cambridge Univ. Press, 1993.

[3] J.-F. Cardoso and A. Souloumiac, "Blind beamforming for non-Gaussian signals," *Proc. Inst. Elect. Eng.—F, Radar Signal Process.*, vol. 140, no. 6, pp. 362–370, Dec. 1993.

[4] J.-F. Cardoso, "On the performance of orthogonal source separation algorithms," in *Proc. EUSIPCO*, 1994, pp. 776–779.

[5] J.-F. Cardoso and A. Souloumiac, "Jacobi angles for simultaneous diagonalization," *SIAM J. Matrix Anal. Appl.*, vol. 17, no. 1, pp. 161–164, Jan. 1996.

[6] A. Belouchrani, K. Abed-Meraïm, J.-F. Cardoso, and E. Moulines, "A blind source separation technique using second order statistics," *IEEE Trans. Signal Process.*, vol. 45, pp. 434–444, Feb. 1997.

[7] M. Wax and J. Sheinvald, "A least-square approach to joint diagonalization," *IEEE Signals Process. Lett.*, vol. 4, no. 2, pp. 52–53, Feb. 1997.

[8] S. Dégerine, "Sur la diagonalisation conjointe approchée par un critère des moindres carrés," in *Proc. 18ème Colloque GRETSI*, Toulouse, France, Sep. 2001, pp. 311–314.

[9] A. J. Van Der Veen, "Joint diagonalization via subspace fitting techniques," in *Proc. Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, Salt Lake City, UT, May 2001, pp. 2773–2776.

[10] M. Joho and K. Rahbar, "Joint diagonalization of correlation matrices by using newton methods with application to blind signal separation," in *Proc. Sensor Array Multichannel Signal Process. Workshop*, Aug. 2002, pp. 403–407.

[11] A. Yeredor, "Non-orthogonal joint diagonalization in the least square sense with application in blind source separation," *IEEE Trans. Signal Process.*, vol. 50, no. 7, pp. 1545–1553, Jul. 2002.

[12] A. Yeredor, A. Ziehe, and K.-R. Müller, "Approximate joint diagonalization using natural gradient approach," in *Lecture Notes in Computer Science (LNCS 3195): Independent Component Analysis and Blind Sources Separation, in Proc. ICA*, Granada, Spain, Sep. 2004, pp. 89–96.

[13] A. Yeredor, "On using exact joint-diagonalization for non-iterative approximate joint-diagonalization," *IEEE Signal Process. Lett.*, vol. 12, no. 9, pp. 645–648, Sep. 2005.

[14] R. Vollgraf and K. Obermayer, "Quadratic optimization for simultaneous matrix diagonalization," *IEEE Trans. Signal Process.*, vol. 54, no. 9, pp. 3270–3278, Sep. 2006.

[15] E.-M. Fadaili, N. Thirion-Moreau, and E. Moreau, "Non orthogonal joint diagonalization/zero-diagonalization for source separation based on time-frequency distributions," *IEEE Trans. Signal Process.*, vol. 55, no. 5, pp. 1673–1687, May 2007.

[16] S. Dégerine and E. Kane, "A comparative study of approximate joint diagonalization algorithms for blind source separation in presence of additive noise," *IEEE, Trans. Signal Process.*, vol. 55, no. 6, pp. 3022–3031, Jun. 2007.

[17] X.-L. Li and X.-D. Zhang, "Nonorthogonal joint diagonalization free of degenerate solution," *IEEE Trans. Signal Process.*, vol. 55, no. 5, pp. 1803–1814, May 2007.

[18] D. T. Pham and M. Congedo, "Least square joint diagonalization of matrices under an intrinsic scale constraint," in *Proc. 8th Int. Conf. Independent Component Anal. Signal Separation*, Paraty, Brazil, Mar. 2009, pp. 298–305.

[19] P. Tichavsky and A. Yeredor, "Fast approximate joint diagonalization incorporating weight matrices," *IEEE Trans. Signal Process.*, vol. 57, no. 3, pp. 878–891, Mar. 2009.

Gilles Chabriel received the Microelectronic and Automatic Engineer Diploma from the Institut des Sciences de l'Ingénieur de Montpellier (currently Polytech'Montpellier), Montpellier, France, in 1992 and the Ph.D. degree in engineering sciences from the Université du Sud Toulon-Var (USTV), La Garde, France, in 1997.

He is currently an Assistant Professor in the Sciences and Applied Sciences Department of the USTV, where he teaches courses in electronics and signal processing. In the Signals and Systems Team of the Institut Matériaux, Microélectronique, Nanosciences de Provence (IM2NP, UMR CNRS 6242), his research is in the areas of signal processing, telecommunications, and electronic systems.

Jean Barrère was born in Bayonne, France, in 1961. He received the Ph.D. degree in mechanics from the University of Bordeaux I, France, in 1990.

From 1990 to 1994, he worked for ELF-Aquitaine (currently Total-Fina-Elf), CERFACS (European Centre for Research and Advanced Training in Scientific Computation), and Aérospatiale (currently EADS), all in Toulouse, France. His work focused on filtration laws in porous media studies and then plane structure design. He is currently a Research Engineer in the Signals and Systems Team of the Institut Matériaux, Microélectronique, Nanosciences de Provence (IM2NP, UMR CNRS 6242) of the Université du Sud Toulon-Var, La Garde, France, where his research interests include wave separation and array processing.