HAL
open science

Least square joint diagonalization of matrices under an intrinsic scale constraint

Dinh-Tuan Pham, Marco Congedo

To cite this version:

Dinh-Tuan Pham, Marco Congedo. Least square joint diagonalization of matrices under an intrinsic scale constraint. ICA 2009 - 8th International Conference on Independent Component Analysis and Signal Separation, Feb 2009, Paraty, Brazil. pp.298-305, 10.1007/978-3-642-00599-2_38 . hal-00371941

HAL Id: hal-00371941

https://hal.science/hal-00371941v1

Submitted on 31 Mar 2009

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Least square joint diagonalization of matrices under an intrinsic scale constraint

Dinh-Tuan Pham[1] and Marco Congedo[2]

[1] Laboratory Jean Kuntzmann, cnrs - Grenoble INP - UJF (Grenoble, France)
Dinh-Tuan.Pham@imag.fr
http://www-ljk.imag.fr/membres/Dinh-Tuan.Pham
[2] GIPSA-lab, cnrs - UJF - Univ. Stendhal - Grenoble INP (Grenoble, France).

Abstract. We present a new algorithm for approximate joint diagonalization of several symmetric matrices. While it is based on the classical least squares criterion, a novel intrinsic scale constraint leads to a simple and easily parallelizable algorithm, called LSDIC (Least squares Diagonalization under an Intrinsic Constraint). Numerical simulations show that the algorithm behaves well as compared to other approximate joint diagonalization algorithms.

1 Introduction

The diagonalization of a matrix and the joint diagonalization of two matrices are well-established concepts in linear algebra and their use in engineering is ubiquitous. Approximate joint diagonalization (AJD) refers to the problem of diagonalizing more then two matrices simultaneously. Considerable interest for AJD followed the discovery that it yields a solution for independent component analysis (e.g., JADE [2]) and second-order blind source separation (e.g., SOBI [1], see also [8]).

The classical linear instantaneous BSS problem assumes a mixing model of the form $\mathbf{x}(t) = \mathbf{A}\mathbf{s}(t)$, where $\mathbf{x}$ is a K-vector holding the sensor measurements, $\mathbf{A}$ is the $K \times K$ mixing matrix and $\mathbf{s}$ is a K-vector holding the source processes. The task is to recover the sources out of a scaling and permutation indeterminacies from the observation $\mathbf{x}$, assuming no knowledge of $\mathbf{A}$ and of the sources distribution. A simple approach to the source separation problem is to consider a set of matrices $\{\mathbf{C}_1, \ldots, \mathbf{C}_N\}$ consisting of statistics of the observations (of second order in most cases) which are estimates of matrices of the form $\mathbf{A}\mathbf{D}_n\mathbf{A}^T$, where $\mathbf{D}_n$ is a diagonal matrix with k-th diagonal element depending only on the distribution of the k-th source. The AJD then seeks a matrix $\mathbf{B}^T$ such that all N congruence transformations $\mathbf{B}^T\mathbf{C}_n\mathbf{B}$, $n = 1, \ldots, N$, are as diagonal as possible. Therefore it provides an estimate of the inverse of $\mathbf{A}$ (up to a scaling and a permutation) and the BSS problem is solved by $\mathbf{s}(t) = \mathbf{B}^T\mathbf{x}(t)$.

Several iterative algorithms have been developed to solve the AJD problem. A simple way to proceed is to minimize the criterion

$$\sum_{n=1}^{N} \|\mathbf{Off}(\mathbf{B}^T\mathbf{C}_n\mathbf{B})\|^2 \tag{1}$$

where **Off** denotes the operator which retains only the off diagonal of its matrix argument and $\|\mathbf{M}\| = [\mathrm{tr}(\mathbf{M}\mathbf{M}^T)]^{1/2}$ denotes the Frobenius norm of the matrix $\mathbf{M}$. However, without any restriction, one would end up with the trivial solution $\mathbf{B} = \mathbf{0}$. In [3], $\mathbf{B}$ is restricted to the orthogonal group and an efficient algorithm based on Givens rotations is presented. This restriction allows BSS only after whitening the sensor measurements, but whitening is known to affect adversely the quality of the separation forcing the data covariance to be exactly diagonal at the expenses of the input matrix set. In [9, 5] it is required instead that $\mathbf{Diag}(\mathbf{B}^T\mathbf{C}_0\mathbf{B}) = \mathbf{I}$, with $\mathbf{C}_0$ being some positive definite matrix, **Diag** denoting the operator which cancels the off diagonal elements of its matrix argument. This amounts to a normalization of the columns of $\mathbf{B}$ (a scaling). The algorithms are slow because for each iteration one needs to repeat K times the search for the eigenvector associated with the smallest eigenvalue of a $K \times K$ matrix. To avoid degeneracy, as well as the trivial solution, in [6] a term proportional to $-\log|\det(\mathbf{B})|$ is added to the criterion, albeit yielding an even slower algorithm.

Another line of research has focused on multiplicative algorithms with superlinear convergence. Such algorithms update the $\mathbf{C}_n$ matrices at each iteration and typically are faster than those minimizing the off-criterion. In [7] the introduction of a different criterion leads to a very efficient multiplicative Givens-like algorithm. However, it allows only positive definite input matrices and such requirement may be cumbersome in some BSS applications. The notion of criterion is dropped altogether in [10], where a multiplicative algorithm based on heuristics is described instead.

In this paper we will elaborate further upon the off criterion (1), expanding results presented in [4], where it has been shown that its minimizer under certain constraint satisfies a nested system of K generalized Rayleigh quotients. We propose a pseudo-Newton fixed point algorithm to solve such system which requires no eigenvalue-eigenvector decomposition and no $\mathbf{C}_n$ matrices updates. We discuss the local stability of the algorithm, i.e., its convergence near the solution. The algorithm, named LSDIC (Least Squares Diagonalization under an Intrinsic Constraint) has complexity per iteration similar to multiplicative algorithms. As in the case of other algorithms minimizing the off-criterion, the convergence is linear, thus overall execution time may be superior to multiplicative algorithms. However LSDIC is naturally parallelizable not only with respect to N, like multiplicative algorithms, but also with respect to K, allowing computational super-efficiency in massive parallel computing architectures. Furthermore, unlike multiplicative algorithms (with the exception of [7]), it accommodates complex mixing and/or sources.

2 The criterion

We consider the joint approximation diagonalization of a set of N symmetric $K \times K$ matrices $\mathbf{C}_1, \ldots, \mathbf{C}_N$. We consider the least square criterion (1) subjected to the constraint: $\sum_{n=1}^{N}(\mathbf{b}_k^T\mathbf{C}_k\mathbf{b}_k)^2 = 1,\ k = 1, \ldots, K$ ($\mathbf{b}_k$ denoting the k-th column of $\mathbf{B}$). Unlike other constraints, this is a new intrinsic constraint that

does not favor a particular matrix in the above set or make use of another matrix outside this set.

Since $\sum_{n=1}^{N}(\mathbf{b}_k^T\mathbf{C}_n\mathbf{b}_k)^2 = \sum_{n=1}^{N}\mathbf{b}_k^T\mathbf{C}_n\mathbf{b}_k\mathbf{b}_k^T\mathbf{C}_n\mathbf{b}_k$ the constraint can be rewritten as $\mathbf{b}_k^T\mathbf{M}(\mathbf{b}_k)\mathbf{b}_k = 1,\ k = 1,\ldots,K$, where

$$\mathbf{M}(\mathbf{b}_k) = \sum_{n=1}^{N}\mathbf{C}_n\mathbf{b}_k\mathbf{b}_k^T\mathbf{C}_n. \tag{2}$$

Further, $\|\mathbf{Off}(\mathbf{B}^T\mathbf{C}_n\mathbf{B})\|^2 = \|\mathbf{B}^T\mathbf{C}_n\mathbf{B}\|^2 - \sum_{k=1}^{K}(\mathbf{b}_k^T\mathbf{C}_n\mathbf{b}_k)^2$ and $\|\mathbf{B}^T\mathbf{C}_n\mathbf{B}\|^2 = \mathrm{tr}(\mathbf{B}^T\mathbf{C}_n\mathbf{B}\mathbf{B}^T\mathbf{C}_n\mathbf{B})$, therefore, the criterion (1) under the above constraint reduces to $\mathrm{tr}[\mathbf{B}^T\mathbf{M}(\mathbf{B})\mathbf{B}]-K$ where $\mathbf{M}(\mathbf{B})$ is defined as in (2) but with $\mathbf{b}_k$ replaced by $\mathbf{B}$. Note that $\mathbf{M}(\mathbf{B})$ can also be computed as $\mathbf{M}(\mathbf{B}) = \sum_{k=1}^{K}\mathbf{M}(\mathbf{b}_k)$.

One can turn the minimization problem of $\mathrm{tr}[\mathbf{B}^T\mathbf{M}(\mathbf{B})\mathbf{B}]$ under the constraint $\mathbf{b}_k^T\mathbf{M}(\mathbf{b}_k)\mathbf{b}_k = 1,\ k = 1,\ldots,K$, into one without constraint, as follows. Note that for an arbitrary matrix $\mathbf{B}$, the matrix $\mathbf{B}\mathbf{D}^{-1/4}(\mathbf{B})$, where $\mathbf{D}(\mathbf{B})$ is the diagonal matrix with diagonal elements

$$d(\mathbf{b}_k) = \mathbf{b}_k^T\mathbf{M}(\mathbf{b}_k)\mathbf{b}_k, \quad k = 1,\ldots,K, \tag{3}$$

will satisfy the constraint. Replacing $\mathbf{B}$ in the criterion $\mathrm{tr}[\mathbf{B}^T\mathbf{M}(\mathbf{B})\mathbf{B}]$ by $\mathbf{B}\mathbf{D}^{-1/4}(\mathbf{B})$ then yields the criterion

$$\mathcal{C}(\mathbf{B}) = \mathrm{tr}[\mathbf{D}^{-1/2}(\mathbf{B})\mathbf{B}^T\tilde{\mathbf{M}}(\mathbf{B})\mathbf{B}] = \sum_{n=1}^{N}\mathrm{tr}\{[\mathbf{D}^{-1/2}(\mathbf{B})\mathbf{B}^T\mathbf{C}_n\mathbf{B}]^2\}, \tag{4}$$

where

$$\tilde{\mathbf{M}}(\mathbf{B}) = \sum_{n=1}^{N}\mathbf{C}_n\mathbf{B}\mathbf{D}^{1/2}(\mathbf{B})\mathbf{B}^T\mathbf{C}_n = \sum_{k=1}^{K}\mathbf{M}(\mathbf{b}_k)/d^{1/2}(\mathbf{b}_k).$$

Therefore one may simply minimize $C(\mathbf{B})$ given in (4) *without any constraint*.

3 Gradient of the criterion

To compute the gradient of the criterion (4) at the point $\mathbf{B}$ we shall perform its Taylor expansion around $\mathbf{B}$ up to first order. Let $\mathbf{\Delta}$ be a small increment of $\mathbf{B}$ then for any symmetric matrix $\mathbf{E}$, one has

$$\begin{aligned}
&\mathrm{tr}\{[\mathbf{E}(\mathbf{B}+\mathbf{\Delta})^T\mathbf{C}_n(\mathbf{B}+\mathbf{\Delta})]^2\}\\
&\qquad = \mathrm{tr}[(\mathbf{E}\mathbf{B}^T\mathbf{C}_n\mathbf{B} + \mathbf{E}\mathbf{B}^T\mathbf{C}_n\mathbf{\Delta} + \mathbf{E}\mathbf{\Delta}^T\mathbf{C}_n\mathbf{B} + \mathbf{E}\mathbf{\Delta}^T\mathbf{C}_n\mathbf{\Delta})^2]\\
&\qquad = \mathrm{tr}[(\mathbf{E}\mathbf{B}^T\mathbf{C}_n\mathbf{B})^2] + 4\mathrm{tr}(\mathbf{E}\mathbf{B}^T\mathbf{C}_n\mathbf{B}\mathbf{E}\mathbf{B}^T\mathbf{C}_n\mathbf{\Delta}) + O(\|\mathbf{\Delta}\|^2),
\end{aligned} \tag{5}$$

where $O(\|\mathbf{\Delta}\|^2)$ denotes a term of the same order as $\|\mathbf{\Delta}\|^2$ as $\mathbf{\Delta} \to 0$. By the same calculation with $\mathbf{E}$ being the identity matrix and $\mathbf{B}$ and $\mathbf{\Delta}$ replaced by their k-th rows $\mathbf{b}_k$ and $\boldsymbol{\delta}_k$, one gets, after summing up with respect to n:

$$d(\mathbf{b}_k + \boldsymbol{\delta}_k) = d(\mathbf{b}_k) + 4\mathbf{b}_k^T\mathbf{M}(\mathbf{b}_k)\boldsymbol{\delta}_k + O(\|\boldsymbol{\delta}_k\|^2).$$

Therefore

$$\mathbf{D}(\mathbf{B}+\mathbf{\Delta}) = \mathbf{D}(\mathbf{B}) + 4\mathbf{Diag}[\mathbf{P}^T(\mathbf{B})\mathbf{\Delta}] + O(\|\mathbf{\Delta}_k\|^2)$$

where

$$\mathbf{P}(\mathbf{B}) = [\,\mathbf{M}(\mathbf{b}_1)\mathbf{b}_1 \quad \cdots \quad \mathbf{M}(\mathbf{b}_K)\mathbf{b}_K\,].$$

Then from the first order Taylor expansion $(x+\epsilon)^{-1/2} = x^{-1/2} - \frac{1}{2}x^{-3/2}\epsilon + O(|\epsilon|^2)$, one gets

$$\mathbf{D}^{-1/2}(\mathbf{B}+\mathbf{\Delta}) = \mathbf{D}^{-1/2}(\mathbf{B}) - 2\mathbf{D}^{-3/2}(\mathbf{B})\mathbf{Diag}[\mathbf{P}^T(\mathbf{B})\mathbf{\Delta}] + O(\|\mathbf{\Delta}\|^2) \qquad (6)$$

Applying now (5) with $\mathbf{E} = \mathbf{D}^{-1/2}(\mathbf{B}+\mathbf{\Delta}) = \mathbf{D}^{-1/2}(\mathbf{B}) + O(\|\mathbf{D}\|)$ yields

$$\begin{aligned}\mathrm{tr}\{[\mathbf{D}^{-1/2}(\mathbf{B}+\mathbf{\Delta})(\mathbf{B}+\mathbf{\Delta})^T\mathbf{C}_n(\mathbf{B}+\mathbf{\Delta})]^2\} = \\ \mathrm{tr}\{[\mathbf{D}^{-1/2}(\mathbf{B}+\mathbf{\Delta})\mathbf{B}^T\mathbf{C}_n\mathbf{B}]^2\} + 4\mathrm{tr}[\mathbf{D}^{-1/2}(\mathbf{B})\mathbf{B}^T\mathbf{C}_n\mathbf{B}\mathbf{D}^{-1/2}(\mathbf{B})\mathbf{B}^T\mathbf{C}_n\mathbf{\Delta}] \\ + O(\|\mathbf{\Delta}\|^2).\end{aligned}$$

But by (6)

$$\begin{aligned}\mathrm{tr}\{[\mathbf{D}^{-1/2}(\mathbf{B}+\mathbf{\Delta})\mathbf{B}^T\mathbf{C}_n\mathbf{B}]^2\} = \mathrm{tr}\{[\mathbf{D}^{-1/2}(\mathbf{B})\mathbf{B}^T\mathbf{C}_n\mathbf{B}]^2\} \\ - 4\mathrm{tr}\{\mathbf{B}^T\mathbf{C}_n\mathbf{B}\mathbf{D}^{-1/2}(\mathbf{B})\mathbf{B}^T\mathbf{C}_n\mathbf{B}\mathbf{D}^{-3/2}(\mathbf{B})\mathbf{Diag}[\mathbf{P}(\mathbf{B})\mathbf{\Delta}]\} + O(\|\mathbf{\Delta}\|^2).\end{aligned}$$

Therefore combining the above results and summing up with respect to n, one gets:

$$\begin{aligned}\mathcal{C}(\mathbf{B}+\mathbf{\Delta}) = \mathcal{C}(\mathbf{B}) + 4\mathrm{tr}[\mathbf{D}^{-1/2}(\mathbf{B})\mathbf{B}^T\tilde{\mathbf{M}}(\mathbf{B})\mathbf{\Delta}] \\ - 4\mathrm{tr}\{\mathbf{B}^T\tilde{\mathbf{M}}(\mathbf{B})\mathbf{B}\mathbf{D}^{-3/2}(\mathbf{B})\mathbf{Diag}[\mathbf{P}^T(\mathbf{B})\mathbf{\Delta}]\} + O(\|\mathbf{\Delta}\|^2).\end{aligned}$$

Finally, noting that $\mathrm{tr}[\mathbf{U}\mathbf{Diag}(\mathbf{V})] = \mathrm{tr}[\mathbf{Diag}(\mathbf{U})\mathbf{Diag}(\mathbf{V})] = \mathrm{tr}[\mathbf{Diag}(\mathbf{U})\mathbf{V}]$, one gets

$$\mathcal{C}(\mathbf{B}+\mathbf{\Delta}) = \mathcal{C}(\mathbf{B}) + 4\mathrm{tr}\{\mathbf{D}^{-1/2}(\mathbf{B})[\mathbf{B}^T\tilde{\mathbf{M}}(\mathbf{B}) - \mathbf{\Gamma}(\mathbf{B})\mathbf{P}^T(\mathbf{B})]\mathbf{\Delta}\} + O(\|\mathbf{\Delta}\|^2).$$

where

$$\mathbf{\Gamma}(\mathbf{B}) = \mathbf{D}^{-1}(\mathbf{B})\mathbf{Diag}[\mathbf{B}^T\tilde{\mathbf{M}}(\mathbf{B})\mathbf{B}].$$

The above relation shows that the gradient of the criterion $\mathcal{C}$ is

$$\mathcal{C}'(\mathbf{B}) = 4[\tilde{\mathbf{M}}(\mathbf{B})\mathbf{B} - \mathbf{P}(\mathbf{B})\mathbf{\Gamma}(\mathbf{B})]\mathbf{D}^{-1/2}(\mathbf{B})].$$

Setting the gradient to 0 yields the equations

$$\mathbf{M}(\mathbf{b}_k)\mathbf{b}_k = \frac{\mathbf{b}_k^T\mathbf{M}(\mathbf{b}_k)\mathbf{b}_k}{\mathbf{b}_k^T\tilde{\mathbf{M}}(\mathbf{B})\mathbf{b}_k}\tilde{\mathbf{M}}(\mathbf{B})\mathbf{b}_k, \qquad k = 1,\ldots,K$$

which means that $\mathbf{b}_k$ is a generalized eigenvector of $\mathbf{M}(\mathbf{b}_k)$ relative to $\tilde{\mathbf{M}}(\mathbf{B})$, with eigenvalue $[\mathbf{b}_k^T\mathbf{M}(\mathbf{b}_k)\mathbf{b}_k]/[\mathbf{b}_k^T\tilde{\mathbf{M}}(\mathbf{B})\mathbf{b}_k]$.

4 Pseudo Newton algorithm

We have thought of implementing quasi Newton algorithm to minimize the criterion (4), which would require the calculation of an approximate Hessian. Although such calculation is theoretically possible, the result is quite involved so that the corresponding algorithm is very complex and costly computationally. Therefore we shall replace the true Hessian with some reasonable positive definite matrix, which we call pseudo Hessian. Let $\mathrm{vec}(\mathbf{B})$ denotes the vector formed by stacking the columns of $\mathbf{B}$, then the pseudo Newton algorithm can be expressed as $\mathrm{vec}(\mathbf{B}) \leftarrow \mathrm{vec}(\mathbf{B}) - \mathcal{H}^{-1}(\mathbf{B})\mathrm{vec}[\mathcal{C}'(\mathbf{B})]$ where $\mathcal{H}(\mathbf{B})$ is the pseudo Hessian. We take

$$\mathcal{H}(\mathbf{B}) = 4 \begin{bmatrix} \dfrac{\tilde{\mathbf{M}}(\mathbf{B})}{d^{1/2}(\mathbf{b}_1)} & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \dfrac{\tilde{\mathbf{M}}(\mathbf{B})}{d^{1/2}(\mathbf{b}_K)} \end{bmatrix}.$$

This is a reasonable choice as it is positive definite and is of about the same order of magnitude as the Hessian (the expansion of $\mathcal{C}(\mathbf{B}+\mathbf{\Delta})$ up to second order in $\mathbf{\Delta}$ contains actually the positive term $4\mathrm{tr}[\mathbf{\Delta}^T\mathbf{D}^{-1/2}(\mathbf{B})\tilde{\mathbf{M}}(\mathbf{B})\mathbf{\Delta}]$). But the main reason for making the above choice is that the corresponding algorithm reduces to a very simple fixed point iteration:

$$\mathbf{B} \leftarrow \tilde{\mathbf{M}}^{-1}(\mathbf{B})\mathbf{P}(\mathbf{B})\mathbf{\Gamma}(\mathbf{B})\mathbf{D}^{-1/2}(\mathbf{B}),$$

or explicitly

$$\mathbf{b}_k \leftarrow \frac{\mathbf{b}_k^T\tilde{\mathbf{M}}(\mathbf{B})\mathbf{b}_k}{d(\mathbf{b}_k)}\tilde{\mathbf{M}}^{-1}(\mathbf{B})\mathbf{M}(\mathbf{b}_k)\mathbf{b}_k, \quad k = 1, \ldots, K.$$

Further, since the criterion (4) is scale invariant, we may drop the factor $\mathbf{b}_k^T\tilde{\mathbf{M}}(\mathbf{B})\mathbf{b}_k/d(\mathbf{b}_k)$ and renormalize $\mathbf{b}_k$ so that $d(\mathbf{b}_k) = 1$. This leads to the algorithm:

$$\mathbf{b}_k \leftarrow \mathbf{M}^{-1}(\mathbf{B})\mathbf{M}(\mathbf{b}_k)\mathbf{b}_k, \quad \mathbf{b}_k \leftarrow \mathbf{b}_k/[\mathbf{b}_k^T\mathbf{M}(\mathbf{b}_k)\mathbf{b}_k]^{1/4}, \quad k = 1, \ldots, K,$$

assuming that $\mathbf{B}$ has been previously normalized so that $\tilde{\mathbf{M}}(\mathbf{B}) = \mathbf{M}(\mathbf{B})$.

It is worthwhile to note that the normalization of $\mathbf{b}_k$ already requires the calculation of $\mathbf{M}(\mathbf{b}_k)\mathbf{b}_k$, hence the computation of the unnormalized new $\mathbf{b}_k$ requires only a pre-multiplication with $\tilde{\mathbf{M}}^{-1}(\mathbf{B})$. Thus our algorithm can be implemented as follows.

1. Initialization: Start with some unnormalized $\mathbf{B}$. For $k = 1, \ldots, K$, compute (in parallel): $\mathbf{M}_k = \sum_{n=1}^N \mathbf{C}_n\mathbf{b}_k\mathbf{b}_k^T\mathbf{C}_n$, $\mathbf{p}_k = \mathbf{M}_k\mathbf{b}_k$, $s_k = (\mathbf{b}_k^T\mathbf{p}_k)^{1/2}$ and normalize $\mathbf{b}_k \leftarrow \mathbf{b}_k/s_k^{1/2}$, $\mathbf{p}_k \leftarrow \mathbf{p}_k/s_k^{3/2}$.
2. Iteration: while not converge do

- Compute $\mathbf{M} = \sum_{k=1}^{K} \mathbf{M}_k / s_k$, then make the Cholesky decomposition $\mathbf{M} = \mathbf{R}^T\mathbf{R}$ ($\mathbf{R}$ upper triangular).
- For $k = 1, \ldots, K$, compute (in parallel): $\mathbf{b}_k = \mathbf{R}^{-1}(\mathbf{R}^{T\,-1}\mathbf{p}_k)$, $\mathbf{M}_k = \sum_{n=1}^{N} \mathbf{C}_n \mathbf{b}_k \mathbf{b}_k^T \mathbf{C}_n$, $\mathbf{p}_k = \mathbf{M}_k \mathbf{b}_k$, $s_k = (\mathbf{b}_k^T \mathbf{p}_k)^{1/2}$ and normalize $\mathbf{b}_k \leftarrow \mathbf{b}_k / s_k^{1/2}$, $\mathbf{p}_k \leftarrow \mathbf{p}_k / s_k^{3/2}$. (Note that multiplication by the inverse of a triangular matrix is done by solving a linear triangular system.)

The above fixed point algorithm is locally stable (i.e. convergent when started near a solution) if the matrix $\mathbf{I} - \mathcal{H}^{-1}(\mathbf{B})\mathcal{C}''(\mathbf{B})$, where $\tilde{\mathcal{C}}''(\mathbf{B})$ denotes the true Hessian, has all its eigenvalues of modulus strictly less than 1. The convergence speed is controlled by the maximum modulus of the eigenvalues. For the (quasi-) Newton algorithm, $\mathcal{H}$ is (nearly) equal to $\mathcal{C}''(\mathbf{B})$, hence the matrix $I - \mathcal{H}^{-1}(\mathbf{B})\mathcal{C}''(\mathbf{B})$ is (nearly) zero and the algorithm has (almost) quadratic convergence. For the pseudo Newton algorithm, if the chosen $\mathcal{H}(\mathbf{B})$ is not too far from $\mathcal{C}''(\mathbf{B})$, one may still expect convergence although the speed is only linear. Simulations indeed show that our pseudo Newton algorithm converges generally well. Note that (global) convergence is not guaranteed even in the quasi-Newton algorithm, as the starting point may be too far from the solution. But since $\mathcal{H}(\mathbf{B})$ is positive definite, one can always reduce the step size, i.e. by taking the new $\mathrm{vec}(\mathbf{B})$ as $\mathrm{vec}(\mathbf{B}) - \lambda\mathcal{H}^{-1}(\mathbf{B})\mathcal{C}'(\mathbf{B})$, with $\lambda \in (0, 1]$, so that the criterion is decreased at each iteration. The algorithm would then converge at least to a local minimum point.

5 Simulation

We compare our LSDIC algorithm to the well-established FFDIAG algorithm of [10] and QDIAG of [9]. We plan to perform a more comprehensive comparison and to publish that in a longer article elsewhere.

Square diagonal matrices with each diagonal entry distributed as a χ-square random variable with one degree of freedom are generated. Each of these matrices, named $\mathbf{D}_n$, may represent the error-free covariance matrix of independent standard Gaussian processes. The noisy input matrices are obtained as $\mathbf{C}_n = \mathbf{A}\mathbf{D}_n\mathbf{A}^T + \mathbf{N}_n$ with symmetric noise matrix $\mathbf{N}_n$ having entries randomly distributed as a Gaussian with zero mean and standard deviation σ. Furthermore, the diagonal elements of $\mathbf{N}_n$ are taken unsigned so to obtain (in general) positive definitive input matrices $\mathbf{C}_n$. The parameter σ controls the overall signal to noise ratio of the input matrices. Two different values will be considered, of which one ($\sigma = 0.01$) represents a small amount of noise closely simulating the exact joint diagonalization (JD) case and the other ($\sigma = 0.05$) simulating the approximate JD case. Two kinds of mixing matrix $\mathbf{A}$ are considered. In the general case mixing matrix $\mathbf{A}$ is obtained as the pseudo-inverse of a matrix with unit norm row vectors which entries are randomly distributed as a standard Gaussian; in this case the mixing matrix may be badly conditioned and we can evaluate the robustness of the AJD algorithms with respect to the conditioning of the mixing matrix. We also consider the case in which $\mathbf{A}$ is a random orthogonal matrix; in this case we can evaluate their robustness with respect to noise.

For QDIAG a normalization matrix $\mathbf{C}_0$ such that $\mathbf{Diag}(\mathbf{B}^T\mathbf{C}_0\mathbf{B}) = \mathbf{I}$ needs to be specified; we use the sum of the input matrix for $\mathbf{C}_0$. As it is well known given true mixing $\mathbf{A}$, each AJD algorithm estimates demixing matrix $\mathbf{B}^T$, which should approximate the inverse of actual $\mathbf{A}$ out of row scaling and permutation. Then, the global matrix $\mathbf{G} = \mathbf{B}^T\mathbf{A}$ should equal a scaled permutation matrix. At each (simulation) repetition we compute the performance index as

$$\text{Performance Index} = \frac{2(K-1)\sum_{i=1}^{K}\sum_{j=1}^{K} G_{ij}^2}{\sum_{i=1}^{K}\max_{j=1,\ldots,K} G_{ij}^2 + \sum_{j=1}^{K}\max_{i=1,\ldots,K} G_{ij}^2}. \quad (7)$$

This index is positive and reaches its maximum 1 if and only if $\mathbf{G}$ has only one non-null elements in each row and column, i.e., if $\mathbf{B}^T$ has been estimated exactly out of the usual row scaling and permutation ambiguities. We computed the means and standard deviations obtained across 250 repetitions for 30 input matrices of dimension 15×15. For each simulation set we then computed all pair-wise bi-directional student-t tests for the null hypothesis $\mu_j = \mu_k, j \neq k$, where μ_j $(j = 1, 2, 3)$ denotes the performance means of the j-th AJD methods. We corrected the resulting p-values for the number of comparisons (3 method pairs) using Bonferroni's method. Results are presented in table 1.

	Orthogonal Mixing (good conditioning)		Non-Orthogonal Mixing (variable conditioning)	
	$\sigma = 0.01$	$\sigma = 0.05$	$\sigma = 0.01$	$\sigma = 0.05$
QDIAG	0.99976054 (0.00015259)	$0.93320283^{<}$ (0.05647088)	0.99976047 (0.00015269)	0.93436055 (0.05463265)
FFDIAG	0.99975514 (0.00015743)	0.94904938 (0.04617571)	$0.98244331^{<}$ (0.05888834)	$0.88311444^{<}$ (0.13230755)
LSDIC	0.99976258 (0.00014988)	0.95775684 (0.03721794)	0.99976252 (0.00014992)	$0.95796237^{>}$ (0.03702060)

Table 1. Mean and standard deviation (in parentheses) of the performance index (7) attained by QDIAG [9], FFDIAG [10] and our LSDIC algorithm across 250 repetitions of the simulation with $K = 15$ and $N = 30$. The higher the mean and the lower the standard deviation, the better the performance. Legend: $<$ ($>$) indicates that the mean performance of the AJD method is significantly worse (better) as compared to the other two methods as seen with a student-t test with 248 degrees of freedom and setting the type I (false positive) error rate to 0.05 after Bonferroni correction (bi-directional $p(t) < 0.017$).

To see how badly conditioned are our (non orthogonal) mixing matrices, we have computed their condition number for matrix inversion with respect to the Frobenius lscjadiag-ica-final.pdfnorm, which for a square matrix $\mathbf{A}$ is given by $\|\mathbf{A}\|\|\mathbf{A}^{-1}\|$. This number is always not less than 1 and high values indicate bad conditioning. In our 250 repetitions of the simulation, we found that the logarithms base 10 of the condition numbers of our mixing matrices have mean 2.13, standard deviation 0.43, minimum 1.50 and maximum 3.92, which means that some mixing matrices used in the simulation are pretty badly conditioned.

In the case of orthogonal mixing the three algorithms display nearly identical results in the low-noise simulation ($\sigma = 0.01$), whereas in the noisy simulation ($\sigma = 0.05$) QDIAG performs significantly worse then both FFDIAG and LSDIC. In the case of non-orthogonal mixing matrices (general case) FFDIAG performs significantly worse than both QDIAG and LSDIC regardless the noise level, whereas LSDIC significantly outperformed both QDIAG and FFDIAG in the high-noise ($\sigma = 0.05$) case. The lower mean and larger standard deviation displayed by FFDIAG in the non-orthogonal case reflects occasional failing in estimating correctly the demixing matrix $\mathbf{B}$ due to the ill-conditioning of the mixing matrix. On the other hand QDIAG appears little robust with respect to noise regardless the conditioning of the mixing matrix. These simulations show the good behavior of LSDIC both in low and high noise conditions and its robustness with respect of the ill-conditioning of the mixing matrix.

6 Conclusion

We have proposed a new algorithm for joint approximated diagonalization of a set of matrices, based on the common least squares criterion but with a novel intrinsic scale constraint. The algorithm is simple, generally fast and is easy to parallelize.

References

1. A. Belouchrani, K. Abed-Meraim, J.-F. Cardoso, and R. Moulines. A blind source separation technique using second-order statistics. *IEEE Trans. Signal Process.*, 45(2):434–444, 1997.
2. J.-F Cardoso and A. Souloumiac. Blind beamforming for non-gaussian signals. *IEE Proc-F (Radar and Signal Process)*, 140(6):362–370, 1993.
3. J.-F. Cardoso and A. Souloumiac. Jacobi angles for simultaneous diagonalization. *SIAM Journal on Matrix Analysis and Applications*, 17(1):161–164, 1996.
4. M. Congedo and D.-T. Pham. Least-squares joint diagonalization of a matrix set by a congruence transformation. In *Singaporean-French IPAL (Image Processing and Application Laboratory) Symposium*, February 2009.
5. S. Dégerine and E. Kane. A comparative study of approximate joint diagonalization algorithms for blind source separation in presence of additive noise. *IEEE Trans. on Signal Process.*, 55(6):3022–3031, 2007.
6. X.-L. Li and X.D. Zhang. Nonorthogonal joint diagonalization free of degenerate solution. *IEEE Transactions on Signal Processing*, 55(5):1803–1814, 2007.
7. D. T. Pham. Joint approximate diagonalization of positive definite matrices. *SIAM Journal on Matrix Analysis and Applications*, 22(4):1136–1152, 2001.
8. D.-T. Pham and J.-F. Cardoso. Blind separation of instantaneous mixtures of non stationary sources. *IEEE Trans. on Signal Process.*, 42(9):1837–1848, 2001.
9. R. Vollgraf and K. Obermayer. Quadratic optimization for simultaneous matrix diagonalization. *IEEE Trans. on Signal Process.*, 54(9):3270–3278, 2006.
10. A. Ziehe, P. Laskov, G. Nolte, and K.-R. Mller. A fast algorithm for joint diagonalization with non-orthogonal transformations and its application to blind source separation. *Journal of Machine Learning Research*, 5:801–818, 2004.