MATHEMATICAL ENGINEERING
TECHNICAL REPORTS

Algorithm for Error-Controlled Simultaneous Block-Diagonalization of Matrices

Takanori MAEHARA and Kazuo MUROTA

METR 2009–53 December 2009

DEPARTMENT OF MATHEMATICAL INFORMATICS
GRADUATE SCHOOL OF INFORMATION SCIENCE AND TECHNOLOGY
THE UNIVERSITY OF TOKYO
BUNKYO-KU, TOKYO 113-8656, JAPAN

WWW page: http://www.keisu.t.u-tokyo.ac.jp/research/techrep/index.html

Algorithm for Error-Controlled Simultaneous Block-Diagonalization of Matrices

Takanori Maehara* and Kazuo Murota*

December 2009

Abstract

An algorithm is given for the problem of finding the finest simultaneous block-diagonalization of a given set of square matrices. This problem has been studied independently in the area of semidefinite programming and independent component analysis. The proposed algorithm considers the commutant algebra of the matrix $*$-algebra generated by the given matrices. It is simpler than other existing methods, and has the capability of controlling numerical errors. Some numerical examples are presented to demonstrate its merits.

Keywords: block-diagonalization, matrix $*$-algebra, commutant algebra, semidefinite programming, independent component analysis

*Department of Mathematical Informatics, Graduate School of Information Science and Technology, University of Tokyo, Tokyo 113-8656, Japan. maehara@misojiro.t.u-tokyo.ac.jp, murota@mist.i.u-tokyo.ac.jp

1 Introduction

In this paper, we consider the following problems:

> Problem [$\mathbb{R}$]: Given a set of $n \times n$ real matrices $A_1, \ldots, A_N$, find an $n \times n$ orthogonal matrix P such that $P^\top A_1 P, \ldots, P^\top A_N P$ are in a common block-diagonal form.
>
> Problem [$\mathbb{C}$]: Given a set of $n \times n$ complex matrices $A_1, \ldots, A_N$, find an $n \times n$ unitary matrix P such that $P^* A_1 P, \ldots, P^* A_N P$ are in a common block-diagonal form.

Recently, these problems have independently been studied in the area of mathematical programming, semidefinite programming (SDP) in particular, and in the area of signal processing, independent component analysis (ICA) in particular.

SDP is an optimization problem of the form:

$$\begin{array}{ll} \text{minimize} & \mathrm{tr}(A_0 X) \\ \text{subject to} & \mathrm{tr}(A_k X) = b_k \ (k = 1, \ldots, N), \\ & X \succeq O \end{array}$$

where X is the decision variable of an $n \times n$ symmetric matrix, and the problem data are the $N + 1$ symmetric matrices $A_0, A_1, \ldots, A_N$ and N real numbers $b_1, \ldots, b_N$. When the problem is endowed with geometrical or combinatorial symmetry, the data matrices $A_0, A_1, \ldots, A_N$ can often be simultaneously block-diagonalized and the associated SDP can be solved efficiently [2, 7, 8, 9].

In the literature of SDP, two numerical algorithms for simultaneous block-diagonalization are proposed. One is by Murota, Kanno, Kojima, and Kojima [12] and Maehara and Murota [11], to be called MKKKM algorithm in this paper, and the other is by de Klerk, Dobre, and Pasechnik [6]. The main idea of these algorithm is the following. Consider the matrix $*$-algebra generated by the given matrices and use the Artin-Wedderburn type structure theorem for matrix $*$-algebra (cf. Theorem 2.1). In theory, these algorithms always find the finest decomposition. In practice, however, the algorithms sometimes fail because of the numerical errors contained in the given matrices, such as observation errors and truncation errors as well as those incurred in the numerical computation. Other types of algorithms based primarily on symbolic methods are also proposed [3, 9].

ICA is an effective method for signal processing proposed independently by Ans, Hérault, and Jutten and by Barness, Carlin, and Steinberger in the early 1980's. Here we consider an extended framework of ICA, the multidimensional ICA proposed by Cardoso [4]. Let X be a (given) d-dimensional

signal. The multidimensional ICA is to decompose X into mutually independent signals by finding an invertible (constant) matrix W and mutually independent (possibly multidimensional) signals $Y_1, \ldots, Y_m$ such that $W^{-1}X = (Y_1, \ldots, Y_m)$. For this purpose, Cardoso [4] proposed the following method: First, normalize X to have zero mean and unit-matrix variance. Second, build the fourth order cumulant matrices C_{ij} $(i, j = 1, \ldots, d)$ of X, where each C_{ij} is a $d \times d$ matrix whose (k, l) entry is defined as

$$(C_{ij})_{kl} = \langle X_i X_j X_k X_l \rangle - \langle X_i X_j \rangle \langle X_k X_l \rangle - \langle X_i X_k \rangle \langle X_j X_l \rangle - \langle X_i X_l \rangle \langle X_j X_k \rangle \tag{1.1}$$

for $k, l = 1, \ldots, d$. Here $\langle \cdot \rangle$ means the expected value. If these matrices are brought to a common block-diagonal form with m diagonal blocks, then it is understood that X is decomposed into m independent components.

It should be mentioned in this context that there is another method that reduces the ICA problem to simultaneous block-diagonalization using the cross covariance matrices of given (temporally correlated) signals. A detailed discussion is found in Amari [1], where the estimating function theory for semiparametric statistical models is applied to the ICA problem.

In the area of ICA, JADE of Cardoso and Souloumiac [5] is accepted as a standard algorithm to perform ICA via simultaneous block-diagonalization. The simultaneous block-diagonalization part of this algorithm is an extension of the Jacobi algorithm for eigenvalue decomposition. It applies successive Givens rotations to the given matrices until some diagonality criterion, such as the sum of square of off-diagonals, becomes minimal. This method is proposed originally for simultaneous diagonalization but it also works for simultaneous block-diagonalization, as pointed out recently by Theis [13]. This algorithm is not guaranteed to find the finest decomposition but it is robust against numerical errors.

The objective of this paper is to propose an algorithm for simultaneous error-controlled finest block-diagonalization. The algorithm is robust against numerical errors, and consequently suitable for a wide range of applications such as SDP and ICA.

Our algorithm, like the previous algorithms in the area of SDP, uses the algebraic structure of the matrix $*$-algebra $\mathcal{T}$ generated by the given matrices. The main difference is that our algorithm works with the commutant algebra of $\mathcal{T}$, whereas the algorithm of [11, 12] works with $\mathcal{T}$ itself and the algorithm of [6] uses the center of $\mathcal{T}$. More specifically, our algorithm applies the simple component decomposition procedure of the MKKKM algorithm to the commutant algebra of $\mathcal{T}$. This provides the algorithm with simplicity (see Section 3) and a remarkable capability of controlling numerical errors in terms of error-control parameter ϵ (see Section 4). Note that the MKKKM algorithm also has a parameter ϵ for coping with numerical round-off errors but this parameter is essentially different from the error-control parameter ϵ in the present paper. See Remark 1.1 below. Computational results show

that the proposed algorithm outperforms the MKKKM algorithm.

Compared with JADE our algorithm has the advantage that it always finds the finest decomposition. Computational results against some practical example problems from SDP and ICA show that the proposed algorithm competes favorably with JADE.

In this paper, we deal mainly with Problem [$\mathbb{C}$] because the theory of matrix $*$-algebras over $\mathbb{C}$ is simpler than that over $\mathbb{R}$ and hence the main features of the proposed algorithm can be explained more clearly in this case. The proposed algorithms can be adapted readily to Problem [$\mathbb{R}$], as will be explained in Remark 3.10.

This paper is organized as follows. Section 2 describes the theory of matrix $*$-algebras which our algorithm is based on. The basic idea of the algorithm is given in Section 3 and the robust version equipped with an error-controlling mechanism is in Section 4. Numerical examples are shown in Section 5.

Remark 1.1. Both the proposed algorithm and the MKKKM algorithm have a parameter ϵ but they are essentially different. To explain this difference, we describe here the role of the parameter ϵ of the MKKKM algorithm, whereas the role of the parameter ϵ of the proposed algorithm is to be described in Section 4. If the input matrices and the subsequent computations were free from numerical errors, the MKKKM algorithm would not need any parameter ϵ and would find the finest decomposition with probability one (note that the MKKKM algorithm is a randomized algorithm). However, to implement the MKKKM algorithm, we have to introduce an ad-hoc parameter ϵ that is used as a threshold between zero and non-zero. This makes the MKKKM algorithm behave differently from what is expected in theory, as is pointed out by de Klerk et al. [6]. ■

2 Matrix $*$-algebras

Let $\mathcal{M}_n = \mathcal{M}_n(\mathbb{C})$ be the set of $n \times n$ complex matrices. A subset $\mathcal{T}$ of $\mathcal{M}_n$ is said to be a $*$-subalgebra (or a matrix $*$-algebra) over $\mathbb{C}$ if $I_n \in \mathcal{T}$ and $[A, B \in \mathcal{T}; \alpha, \beta \in \mathbb{C} \implies \alpha A + \beta B, AB, A^* \in \mathcal{T}]$. We say that a matrix $*$-algebra $\mathcal{T}$ is simple if $\mathcal{T}$ has no ideal other than $\{O\}$ and $\mathcal{T}$ itself, where an ideal of $\mathcal{T}$ means a submodule $\mathcal{I}$ of $\mathcal{T}$ such that $[A \in \mathcal{T}, B \in \mathcal{I} \implies AB, BA \in \mathcal{I}]$. We say that $\mathcal{T}$ is irreducible if no $\mathcal{T}$-invariant subspace other than $\{\mathbf{0}\}$ and $\mathbb{C}^n$ exists, where a linear subspace W of $\mathbb{C}^n$ is said to be $\mathcal{T}$-invariant if $AW \subseteq W$ for every $A \in \mathcal{T}$.

The following is a standard result in $*$-algebra (e.g., [14, Chapter X]). Note that for a matrix $*$-algebra $\mathcal{T}$ and a unitary matrix P, the set $P^*\mathcal{T}P := \{P^*AP : A \in \mathcal{T}\}$ is another matrix $*$-algebra isomorphic to $\mathcal{T}$.

Theorem 2.1. Let $\mathcal{T}$ be a $*$-subalgebra of $\mathcal{M}_n(\mathbb{C})$.

(A) There exist a unitary matrix Q and simple $*$-subalgebras $\mathcal{T}_j$ of $\mathcal{M}_{\hat{n}_j}$ for some $\hat{n}_j$ $(j = 1, 2, \ldots, \ell)$ such that

$$Q^* \mathcal{T} Q = \bigoplus_{j=1}^{\ell} \mathcal{T}_j.$$

(B) If $\mathcal{T}$ is simple, there exist a unitary matrix P and an irreducible $*$-subalgebra $\mathcal{T}^\circ$ of $\mathcal{M}_{\bar{n}}$ for some $\bar{n}$ such that

$$P^* \mathcal{T} P = \mathcal{T}^\circ \otimes I_\mu,$$

where $n = \bar{n}\mu$.

(C) If $\mathcal{T}$ is irreducible, $\mathcal{T} = \mathcal{M}_n$. ■

Let $\mathcal{T}'$ denote the commutant algebra of a matrix $*$-subalgebra $\mathcal{T}$ of $\mathcal{M}_n$. This means that $\mathcal{T}'$ is the set of all matrices X that commute with every member of $\mathcal{T}$, i.e.,

$$\mathcal{T}' = \{X \in \mathcal{M}_n : [A, X] = O \ (A \in \mathcal{T})\},$$

where $[A, X] := AX - XA$. The set $\mathcal{T}'$ also forms a $*$-subalgebra of $\mathcal{M}_n$. The commutant operation $(\,')$ is compatible with unitary transformation, direct sum, and tensor product, i.e.,

$$\begin{aligned} (P^* \mathcal{T} P)' &= P^* \mathcal{T}' P, \\ (\mathcal{T}_1 \oplus \mathcal{T}_2)' &= \mathcal{T}_1' \oplus \mathcal{T}_2', \\ (\mathcal{T}_1 \otimes \mathcal{T}_2)' &= \mathcal{T}_1' \otimes \mathcal{T}_2'. \end{aligned}$$

We also have

$$\begin{aligned} \mathcal{M}_n(\mathbb{C})' &= \mathbb{C} I_n, \\ (\mathbb{C} I_n)' &= \mathcal{M}_n(\mathbb{C}). \end{aligned}$$

Using these identities, we can see $\mathcal{T}'' = \mathcal{T}$ as follows. By the structure theorem (Theorem 2.1), any matrix $*$-algebra $\mathcal{T}$ can be decomposed into the following form:

$$P^* \mathcal{T} P = \bigoplus_{j=1}^{\ell} (\mathcal{M}_{n_j} \otimes I_{\mu_j}) \tag{2.1}$$

with some unitary matrix P. Taking the commutant and using the above-mentioned properties of the commutant operation, we obtain

$$P^* \mathcal{T}' P = \bigoplus_{j=1}^{\ell} (I_{n_j} \otimes \mathcal{M}_{\mu_j}). \tag{2.2}$$

Taking again the commutant of this expression, we obtain

$$P^*\mathcal{T}''P = \bigoplus_{j=1}^{\ell}(\mathcal{M}_{n_j} \otimes I_{\mu_j}). \tag{2.3}$$

Comparing (2.1) and (2.3), we see $\mathcal{T}'' = \mathcal{T}$.

Remark 2.2. The relationship $\mathcal{T}'' = \mathcal{T}$ is called "double commutant property," which plays a fundamental and important role in the theory of algebra and operator theory. ■

3 Basic algorithm via the commutant

If a unitary matrix P simultaneously block-diagonalizes $A_1, \ldots, A_N$, then it also simultaneously block-diagonalizes all members A of $\mathcal{T}$, where $\mathcal{T}$ is the matrix $*$-algebra generated by $A_1, \ldots, A_N$. The structure theorem (Theorem 2.1) describes the structure of the finest block-diagonalization of a matrix $*$-algebra. Therefore, for the simultaneous block-diagonalization of $A_1, \ldots, A_N$, it is necessary and sufficient to find a unitary matrix P in the structure theorem.

Our algorithm is based on the following proposition for simple component decompositions, which is an immediate consequence of (2.2).

Proposition 3.1. If a unitary matrix P decomposes a matrix $*$-algebra $\mathcal{T}$ into the simple components, then P also decomposes the commutant algebra $\mathcal{T}'$ into the simple components. The converse is also true. ■

The converse part of this proposition implies that it is possible to construct an algorithm that decomposes $\mathcal{T}'$ to obtain the decomposition of $\mathcal{T}$. We will construct such an algorithm.

The simple component decomposition procedure of the MKKKM algorithm is based on the following propositions. Let us say that $A \in \mathcal{T}$ is generic if A does not have the same eigenvalue in distinct simple components of $\mathcal{T}$ and does not have repeated eigenvalues in any irreducible component of $\mathcal{T}$. To be more concrete, decompose A as $P^*AP = \bigoplus_{j=1}^{\ell} B_j \otimes I_{\mu_j}$ compatibly with (2.1), and call A generic if B_i and B_j with $i \neq j$ do not have a common eigenvalue and each B_i is free from multiple eigenvalues.

Proposition 3.2 (Proposition 3 of [12])**.** If we sample a Hermitian matrix $A \in \mathcal{T}$ randomly, then A is generic with probability one. ■

Proposition 3.3 (Proposition 4 of [12])**.** Let $A \in \mathcal{T}$ be a generic Hermitian matrix. Then a unitary matrix that diagonalizes A decomposes $\mathcal{T}$ into simple components. ■

Remark 3.4. The notion of genericity is defined in [12] and it is refined in [11] for Problem [$\mathbb{R}$]. However, since we now consider Problem [$\mathbb{C}$], we only need the original version of the definition introduced above. ■

These propositions imply that we can decompose $\mathcal{T}$ into simple components, once we know how to sample a generic Hermitian matrix in $\mathcal{T}$.

Our strategy here is to apply the above procedure not to $\mathcal{T}$ but to the commutant algebra $\mathcal{T}'$. Then the unitary matrix P constructed for $\mathcal{T}'$ is also good for $\mathcal{T}$ by Proposition 3.1. Thus our problem is reduced to sampling a generic Hermitian matrix, say X, in $\mathcal{T}'$.

Assume that $\mathcal{T}$ is generated by $A_1, \ldots, A_N$. Then a Hermitian matrix X belongs to $\mathcal{T}'$ if and only if X satisfies

$$[A_k, X] = O \quad (k = 1, \ldots, N). \tag{3.1}$$

This is a system of linear equations in X, and a random solution to it serves as the X above.

We may summarize our ideas to the following algorithm.

Algorithm 3.5 (Basic form)**.**

1: Sample a generic Hermitian matrix X as a random solution of (3.1).
2: Output a unitary matrix P that diagonalizes X. ■

Remark 3.6. A concrete procedure for a random solution of (3.1) will be given in Section 4 for the robust version of this algorithm. ■

By Propositions 3.1 and 3.2, and the converse part of Proposition 3.3, a unitary matrix P computed by this algorithm obviously decomposes $\mathcal{T}$ into simple components. Furthermore, it turns out that this matrix P decomposes $\mathcal{T}$ into irreducible components. To prove this claim, we need the next lemma.

Lemma 3.7. Let A be an $n \times n$ complex matrix and X be an $n \times n$ Hermitian matrix. If $[A, X] = O$ then, for any unitary matrix P that diagonalizes X as $P^*XP = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$, we have

$$(P^*AP)_{ij} = 0 \quad \text{if} \quad \lambda_i \neq \lambda_j. \tag{3.2}$$

Proof. If $[A, X] = O$, then $[P^*AP, P^*XP] = O$. The (i, j) component of this identity reads

$$(P^*AP)_{ij}(\lambda_i - \lambda_j) = 0.$$

Hence $(P^*AP)_{ij} = 0$ if $\lambda_i \neq \lambda_j$. □

This lemma implies that if a Hermitian matrix X satisfies (3.1) and a unitary matrix P diagonalizes X, then P is a matrix that block-diagonalizes $A_1, \ldots, A_N$. Moreover, this gives the finest decomposition, as stated below.

Proposition 3.8. Algorithm 3.5 gives the finest block-diagonal decomposition of $A_1, \ldots, A_N$.

Proof. Let X be a random Hermitian solution of (3.1) and P be a unitary matrix that diagonalizes X. Then X is generic in $\mathcal{T}'$ by Proposition 3.2.

Let us count the multiplicity of an eigenvalue of X. By equation (2.2) and genericity of X, the multiplicity of an eigenvalue of X is n_j for some $j = 1, \ldots, \ell$. Therefore, the diagonal blocks of $P^* A_k P$ $(k = 1, \ldots, N)$ are of size n_j $(j = 1, \ldots, \ell)$, which are fine as in the decomposition in Theorem 2.1. This shows that the obtained decomposition $P^* A_k P$ $(k = 1, \ldots, N)$ is the finest decomposition. □

We now give an example to illustrate how the proposed algorithm works.

Example 3.1. Let

$$A_1 = \begin{bmatrix} 2 & 1 & 0 & 0 \\ 1 & 2 & 0 & 0 \\ 0 & 0 & 1 & 2 \\ 0 & 0 & 2 & 1 \end{bmatrix}, \; A_2 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}, \; A_3 = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}.$$

The matrix

$$X = \begin{bmatrix} 0.2057 & 0.1525 & -0.3036 & -0.3036 \\ 0.1525 & 0.2057 & -0.3036 & -0.3036 \\ -0.3036 & -0.3036 & 0.2057 & 0.1525 \\ -0.3036 & -0.3036 & 0.1525 & 0.2057 \end{bmatrix}$$

is a generic Hermitian solution of (3.1), which we can obtain by the method to be described in Section 4. The eigenvalue decomposition of X is

$$P^* X P = \mathrm{diag}(-0.2491, 0.9655, 0.0533, 0.0533),$$

where

$$P = \begin{bmatrix} -0.5000 & -0.5000 & -0.4330 & -0.5590 \\ -0.5000 & -0.5000 & 0.4330 & 0.5590 \\ -0.5000 & 0.5000 & -0.5590 & 0.4330 \\ -0.5000 & 0.5000 & 0.5590 & -0.4330 \end{bmatrix}.$$

Since X has three distinct eigenvalues with multiplicities 1, 1, and 2, the given matrices A_1, A_2, and A_3 can be decomposed into three diagonal blocks

with sizes 1, 1, and 2. Indeed, we have

$$P^*A_1P = \left[\begin{array}{c|c|cc} 3.0000 & 0.0000 & 0.0000 & 0.0000 \\ \hline 0.0000 & 3.0000 & 0.0000 & 0.0000 \\ \hline 0.0000 & 0.0000 & 0.2499 & -0.9683 \\ 0.0000 & 0.0000 & -0.9683 & -0.2499 \end{array}\right],$$

$$P^*A_2P = \left[\begin{array}{c|c|cc} 1.0000 & 0.0000 & 0.0000 & 0.0000 \\ \hline 0.0000 & -1.0000 & 0.0000 & 0.0000 \\ \hline 0.0000 & 0.0000 & 0.9683 & 0.2499 \\ 0.0000 & 0.0000 & 0.2499 & -0.9683 \end{array}\right],$$

$$P^*A_3P = \left[\begin{array}{c|c|cc} 1.0000 & 0.0000 & 0.0000 & 0.0000 \\ \hline 0.0000 & -1.0000 & 0.0000 & 0.0000 \\ \hline 0.0000 & 0.0000 & -0.9683 & -0.2499 \\ 0.0000 & 0.0000 & -0.2499 & 0.9683 \end{array}\right].$$

■

Remark 3.9. Note that there is a slight difference between the decomposition given by Algorithm 3.5 and the decomposition described in Theorem 2.1. Theorem 2.1 says that if $\mathcal{T}$ is simple then $\mathcal{T}$ can be decomposed into the form with identical diagonal blocks, i.e., $\mathcal{T} = \{\mathrm{diag}(B, \dots, B) : B \in \mathcal{T}^\circ\}$. However, Algorithm 3.5 decomposes $\mathcal{T}$ into the form $\mathcal{T} = \{\mathrm{diag}(Q_1^*BQ_1, \dots, Q_\mu^*BQ_\mu) : B \in \mathcal{T}^\circ\}$ for some (unknown) unitary matrices $Q_1, \dots, Q_\mu$. ■

Remark 3.10. The proposed algorithm can be adapted to a matrix $*$-algebra over $\mathbb{R}$: Replace complex conjugate "$*$" by transpose "$^\top$" and "unitary" by "orthogonal." The correctness (the $\mathbb{R}$-variant of Proposition 3.8) can also be proved in the same way. The only complication is that there are three types of irreducible components in a matrix $*$-algebra over $\mathbb{R}$ (cf., e.g., [12, 14]). But the eigenvalue counting argument in the proof of Proposition 3.8 still works for all types of irreducible components. ■

4 Algorithm for error-controlled decomposition

In many applications, the input matrices $A_1, \dots, A_N$ are subject to numerical noises, such as observation errors or truncation errors. In such cases, it is not likely that the given matrices can be decomposed in a nontrivial way in the strict algebraic sense. In practice, however, some reasonable or plausible decomposition is wanted.

In this section, we consider an error-controlled simultaneous block-diagonalization, in which the entries in off-diagonal blocks are controlled by a parameter ϵ, and we propose an algorithm to find such a decomposition. The algorithm can be seen as a parametrized version of Algorithm 3.5. If the error-control parameter ϵ is zero, the algorithm is reduced to Algorithm 3.5. If

the parameter $\epsilon > 0$, the algorithm outputs a unitary matrix P such that $P^*A_1P, \ldots, P^*A_NP$ are close to a block-diagonal form in the sense that the entries in the off-diagonal blocks are of the order of ϵ.

A key observation is the following lemma, which is a parametrized extension of Lemma 3.7. Let $\|A\|$ denote the Frobenius norm of A, i.e., $\|A\| := \sqrt{\mathrm{tr}(A^*A)}$.

Lemma 4.1. Let A be an $n \times n$ complex matrix and X be an $n \times n$ Hermitian matrix. If $\|[A, X]\| \leq \epsilon$, then, for any unitary matrix P that diagonalizes X as $P^*XP = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$, we have

$$|(P^*AP)_{ij}| \cdot |\lambda_i - \lambda_j| \leq \epsilon.$$

Proof. The proof is a straightforward extension of the proof of Lemma 3.7. If $\|[A, X]\| \leq \epsilon$ then $\|[P^*AP, P^*XP]\| \leq \epsilon$. Therefore

$$\sum_{i,j} |(P^*AP)_{ij}(\lambda_i - \lambda_j)|^2 \leq \epsilon^2.$$

This implies the desired inequality. □

This lemma implies that if a Hermitian matrix X satisfies

$$\|[A_k, X]\| \leq \epsilon \quad (k = 1, \ldots, N), \tag{4.1}$$

then the (i, j) component of P^*A_kP is of the order of ϵ, as long as λ_i and λ_j are not too close. Hence, by replacing the equation (3.1) by (4.1) in Step 1 in Algorithm 3.5, we obtain a parametrized version of the algorithm that is equipped with an error-control mechanism.

Algorithm 4.2 (Error-controlling version of the algorithm)**.**

1: Sample a generic Hermitian matrix X as a random solution of (4.1).
2: Output a unitary matrix P that diagonalizes X. ■

In the rest of this section, we describe how to obtain a random Hermitian solution of (4.1) needed in Step 1 of Algorithm 4.2.

If we have a (non-Hermitian) X that satisfies

$$\|[A_k, X]\| \leq \epsilon, \quad \|[A_k^*, X]\| \leq \epsilon \quad (k = 1, \ldots, N), \tag{4.2}$$

then $(X + X^*)/2$ is a Hermitian solution of (4.1). Therefore we only have to consider how to obtain a random (non-Hermitian) solution of (4.2).

Let $\mathrm{vec}(X) := (x_{11}, \ldots, x_{nn})$ be the n^2-dimensional vector associated with an $n \times n$ matrix $X = (x_{ij})$. Since $[A, X]$ is linear in x_{ij}, there exists an $n^2 \times n^2$ matrix T, depending on A, such that

$$\mathrm{vec}([A, X]) = T(\mathrm{vec}(X)). \tag{4.3}$$

Note that

$$\|[A, X]\| = \|\mathrm{vec}([A, X])\| = \|T\mathrm{vec}(X)\|. \tag{4.4}$$

Let $T_1, \ldots, T_N$ and $\hat{T}_1, \ldots, \hat{T}_N$ be the matrices that correspond to $A_1, \ldots, A_N$ and $A_1^*, \ldots, A_N^*$. Then finding a matrix X in (4.2) is equivalent to finding an n^2-dimensional vector u such that

$$\|T_k u\| \leq \epsilon, \quad \|\hat{T}_k u\| \leq \epsilon \quad (k = 1, \ldots, N). \tag{4.5}$$

To find such u, we consider the $n^2 \times n^2$ Hermitian matrix S defined as

$$S = \sum_{k=1}^{N} \left(T_k^* T_k + \hat{T}_k^* \hat{T}_k \right). \tag{4.6}$$

Let $v_1, \ldots, v_r$ be the normalized (mutually orthogonal) eigenvectors of S with the corresponding eigenvalues smaller than ϵ^2. Then any vector $u = c_1 v_1 + \cdots + c_r v_r$ with $|c_1|^2 + \cdots + |c_r|^2 = 1$ satisfies (4.5).

Summarizing the above argument, we obtain the following algorithm to sample a random Hermitian matrix X of $\mathcal{T}'$.

Algorithm 4.3 (Generating a random Hermitian X)**.**

1: Construct the matrix S in (4.6) from $A_1, \ldots, A_N$.
2: Find normalized eigenvectors, say, $v_1, \ldots, v_r$ of S such that corresponding eigenvalues are smaller than ϵ^2.
3: Sample $c_1, \ldots, c_r$ randomly with $|c_1|^2 + \cdots + |c_r|^2 = 1$.
4: Put $u = c_1 v_1 + \cdots + c_r v_r$ and output the matrix X such that $\mathrm{vec}(X) = u$.

■

Remark 4.4. In some cases a suitable value of parameter ϵ must be determined from the input matrices $A_1, \ldots, A_N$ alone. Since the algorithm computes the eigenvalues of S explicitly, it is reasonable to set ϵ to a value that separates relatively small eigenvalues of S. See Example 5.1 in Section 5.

In other cases we may have a priori knowledge about the order of magnitude of numerical errors in the input and computation. Let $A_1, \ldots, A_N$ be the nominal values of the input matrices, and $\tilde{A}_1, \ldots, \tilde{A}_N$ the actual inputs containing noises and errors. Let $N_k = \tilde{A}_k - A_k$ $(k = 1, \ldots, N)$ and assume that we know a number δ such that $\delta \geq \max_{1 \leq k \leq N} \|N_k\|$. If X satisfies

$$[A_k, X] = O, \tag{4.7}$$

then

$$\|[\tilde{A}_k, X]\| = \|[A_k, X] + [N_k, X]\| \leq 2\|N_k\|\|X\| \leq 2\delta\|X\|.$$

Therefore we can reasonably set $\epsilon = 2\delta\|X\|$ to make the solution X of (4.7) feasible to (4.1). Conversely, if δ is sufficiently small, any feasible solution of (4.1) is close to some solution of (4.7) by continuity. ■

5 Numerical examples

Here we compare three algorithms by four numerical examples.

The algorithms compared here are the following: (1) the proposed algorithm ($\mathbb{R}$ version of Algorithm 4.3), (2) the algorithm due to Murota–Kanno–Kojima–Kojima–Maehara [12, 11] (MKKKM in short), and (3) simultaneous block-diagonalization part of JADE algorithm [5] (JADE in short). We have implemented the proposed algorithm and MKKKM, and used the implementation of JADE by Cardoso[1].

Example problems considered here are the following: Examples 5.1 and 5.2 are illustrative examples and Examples 5.3 and 5.4 are practical examples from SDP and ICA. Note that all examples contain small numerical noises. For all numerical examples, the proposed algorithm competes favorably with MKKKM and JADE.

Remark 5.1. We have excluded the algorithm by de Klerk–Dobre–Pasechnik [6] from our comparison. This is because their algorithm needs a basis of $*$-algebra but it is hard to generate a basis from a set of matrices that are contaminated with numerical noises. ■

Example 5.1 (Noisy version of Example 3.1)**.** Let

$$A_1 = \begin{bmatrix} 2 & 1 & 0 & 0 \\ 1 & 2 & 0 & 0 \\ 0 & 0 & 1 & 2 \\ 0 & 0 & 2 & 1 \end{bmatrix}, \; A_2 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}, \; A_3 = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

be the same matrices as in Example 3.1. Let $\tilde{A}_1, \tilde{A}_2$, and $\tilde{A}_3$ be input matrices such that $\tilde{A}_k = A_k + N(0, 10^{-2})$, where $N(0, 10^{-2})$ is a matrix whose entries independently follow the normal distribution of mean zero and variance 10^{-2}.

The eigenvalues of the matrix S in Algorithm 4.3 are

$$\begin{aligned} &\{0.000, 0.002, 0.004; \quad 8.007; \\ &\; 15.782, 15.955, 15.976, 16.014, 16.094, 16.123, 16.170; \\ &\; 23.927; \quad 39.976, 39.978, 40.054, 40.056\}. \end{aligned}$$

We can see that there are three relatively small eigenvalues, which serve as a clue to an appropriate value of the control-parameter ϵ. Note that if A_1, A_2, and A_2 were input to Algorithm 4.3, the corresponding S would have the following eigenvalues:

$$\begin{aligned} &\{0.000, 0.000, 0.000; \quad 8.000; \\ &\; 16.000, 16.000, 16.000, 16.000, 16.000, 16.000, 16.000; \\ &\; 24.000; \quad 40.000, 40.000, 40.000, 40.000\}, \end{aligned}$$

[1] `http://www.tsi.enst.fr/~cardoso/Algo/Joint_Diag/joint_diag_r.m`

which are very close to the eigenvalues for $\tilde{A}_1, \tilde{A}_2$ and $\tilde{A}_3$. In particular, the three small eigenvalues of S for $\tilde{A}_1, \tilde{A}_2$ and $\tilde{A}_3$ correspond to the zero eigenvalues of S for A_1, A_2 and A_3.

We set $\epsilon = 0.070$ to separate the three small eigenvalues and then we obtain

$$P = \begin{bmatrix} -0.4968 & 0.4998 & 0.6101 & -0.3623 \\ -0.5005 & 0.5028 & -0.6038 & 0.3634 \\ 0.5006 & 0.4950 & 0.3642 & 0.6097 \\ 0.5020 & 0.5023 & -0.3615 & -0.6041 \end{bmatrix},$$

$$P^\top X P = \mathrm{diag}(-0.0013, 0.2246, 0.6890, 0.6891).$$

Since X has four distinct eigenvalues with one pair close enough to be identified. The matrices $\tilde{A}_1$, $\tilde{A}_2$, and $\tilde{A}_3$ can be decomposed into three diagonal blocks with small off-diagonal errors of magnitude of $\mathrm{O}(\epsilon)$ as follows:

$$P^\top \tilde{A}_1 P = \left[\begin{array}{r|r|rr} 2.9974 & 0.0095 & -0.0011 & -0.0000 \\ \hline -0.0053 & 2.9919 & 0.0047 & -0.0073 \\ \hline 0.0043 & -0.0074 & -0.4822 & 0.8684 \\ 0.0142 & 0.0112 & 0.8907 & 0.4710 \end{array}\right],$$

$$P^\top \tilde{A}_2 P = \left[\begin{array}{r|r|rr} -0.9942 & -0.0062 & 0.0131 & -0.0042 \\ \hline 0.0049 & 1.0049 & -0.0036 & -0.0065 \\ \hline -0.0083 & 0.0032 & 0.8759 & 0.4638 \\ -0.0025 & -0.0015 & 0.4662 & -0.8762 \end{array}\right],$$

$$P^\top \tilde{A}_3 P = \left[\begin{array}{r|r|rr} -0.9960 & 0.0072 & -0.0152 & 0.0124 \\ \hline -0.0065 & 1.0157 & 0.0085 & 0.0121 \\ \hline -0.0214 & -0.0072 & -0.8841 & -0.4556 \\ -0.0047 & 0.0032 & -0.4778 & 0.8921 \end{array}\right].$$

In this example, MKKKM and JADE also output the same block-diagonal structure but there are differences in off-diagonal values. The mean and the variance of the maximum absolute value of off-diagonals for 100 samples are shown in Table 1. This shows that the proposed algorithm and JADE are superior to MKKKM in this example.

Table 1: Mean and variance of the maximum absolute value of off-diagonals of 100 samples.

Algorithm	Proposed	MKKKM	JADE
Mean	0.0218	0.0373	0.0206
Variance	2.4482×10^{-5}	1.5218×10^{-4}	1.876×10^{-5}

Example 5.2 (High-multiplicity version of Example 5.1)**.** Let A_1, A_2, and A_3 be the same matrices as in Example 5.1. Let B_1, B_2, and B_3 be matrices such that $B_k = \mathrm{diag}(A_k, A_k, A_k) + N(0, 10^{-2})$, which are used as the input to the algorithms.

In this example, MKKKM and JADE never find the finest block-diagonalization; the outputs of MKKKM have large off-block-diagonal values and the outputs of JADE have non-finest block-diagonal structure. In contrast, the proposed algorithm successfully finds the finest block-diagonalization for all problem instances.

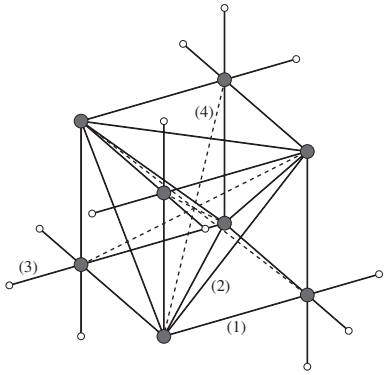


Figure 1: A cubic truss [12].

Example 5.3 (Example from semidefinite programming)**.** Here we work with a problem of SDP in [12]. Consider the cubic truss in Figure 1 (Fig. 1 in [12]), where the dotted members should be ignored here. The truss consists of 30 members and 8 free nodes (therefore the degree of freedom is 24). The members can be divided into three classes, so that we obtain three 24×24 matrices A_1, A_2, A_3.

Remark 5.2. The truss including the dotted members and the truss excluding the dotted members have the same geometrical symmetry ($\mathrm{T_d}$ symmetry) but the truss excluding dotted members has additional algebraic symmetry due to sparsity. Therefore the matrices of the truss excluding dotted members has finer decomposition than the matrices of the truss including dotted members. See [12] for more details. ■

The matrix S in the proposed algorithm is of size 576×576 and have 32 eigenvalues of order 10^{-7}, 18 eigenvalues of order 10^{-1}, and 526 eigenvalues greater than 10^{7}. Set ϵ to separate the eigenvalues of order 10^{-7}, and then we obtain the finest block-diagonal decomposition. (It may be noted that when we set ϵ to separate the eigenvalues smaller than 10^{-1}, we also obtain the finest block-diagonal decomposition).

In this example, the MKKKM algorithm sometimes finds the finest decomposition and sometimes not. (Recall that MKKKM is a randomized algorithm, see Remark 1.1). More precisely, among the 20 runs, we have obtained the finest decomposition 14 times. JADE algorithm never finds the finest decomposition.

Example 5.4 (Example from independent component analysis)**.** Here we work with a problem of ICA, which is a standard setting in the area of ICA. Let Y_1 and Y_2 be 2-dimensional signals of length $T = 10000$ shown in Figure 2(a), representing a sequence of (x, y)-coordinates of $T = 10000$ points. Note that Y_1 corresponds to the character "J" and Y_2 corresponds to the character "A". Let Y_3 and Y_4 be 1-dimensional Gaussian noises of length $T = 10000$. It is assumed that Y_1, Y_2, Y_3, and Y_4 are mutually independent. Let W be a fixed random matrix and put $X = W(Y_1, Y_2, Y_3, Y_4)$, which is 6-dimensional signal of length 10000. The ICA problem is to obtain Y_1 and Y_2 from X using the numerical information of X. An accepted method for this is to find a unitary matrix P that simultaneously block-diagonalizes the fourth order cumulant matrices $A_1, \ldots, A_N$ defined as (1.1). Here each A_k is of size 36×36 and $N = 36$. We use Cardoso's implementation[2] for building these matrices.

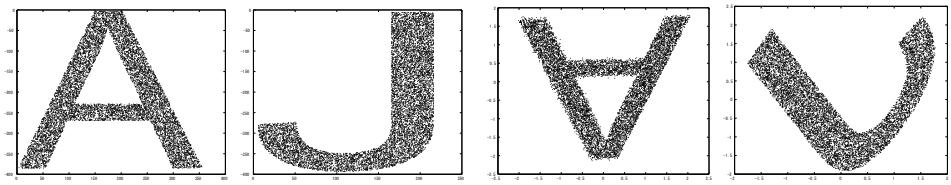
In this example, the proposed algorithm and JADE always found a source signals Y_1 and Y_2 successfully, but MKKKM sometimes failed. Typical outputs are shown in Figure 2 (b)–(d). The outputs of the proposed algorithm and JADE algorithm are sharper than the outputs of the MKKKM algorithm. Note that it is legitimate that the obtained signals are rotated or reflected.

For this example, the proposed algorithm is more useful than MKKKM. JADE is comparable with the proposed algorithm, but the proposed algorithm is recommend for practical use since it has theoretical guarantees for finest-ness of the block-diagonal decomposition and boundness of the off-diagonal errors.

Acknowledgments

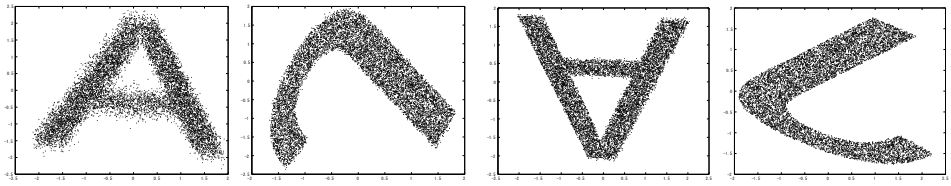
The authors thank Yoshihiro Kanno for the data of Example 5.3. This work is supported by a Grant-in-Aid for Scientific Research from the Ministry of Education, Culture, Sports, Science and Technology of Japan and by the Global COE "The Research and Training Center for New Development in Mathematics."

[2] `http://www.tsi.enst.fr/~cardoso/Algo/Jade/jadeR.m`



(a) Sources S_1 and S_2.

(b) Proposed Algorithm



(c) MKKKM Algorithm

(d) JADE Algorithm

Figure 2: The scatter plots of the signals for the example from ICA.

References

[1] S. Amari: Estimating functions of independent component analysis for temporally correlated signals, Neural Computation, Vol. 12, pp. 2083-2107, 2000.

[2] Y. Bai, E. de Klerk, D. V. Pasechnik and R. Sotirov: Exploiting group symmetry in truss topology optimization, Optimization and Engineering, Vol. 10, No. 3, 2009.

[3] S. Boyd, P. Diaconis, P. Parrilo, L. Xiao: Fastest mixing Markov chain on graphs with symmetries, SIAM Journal on Optimization, Vol. 20, No. 2 (2009), pp. 792–819.

[4] J.-F. Cardoso: Multidimensional independent component analysis, In Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing 1998, Seattle. pp. 1941–1944.

[5] J.-F. Cardoso and A. Souloumiac: Blind beamforming for non Gaussian signals, In IEE Proceedings-F, Vol. 140 (1993), pp. 362–370.

[6] E. de Klerk, C. Dobre, and D. V. Pasechnik: Numerical block diagonalization of matrix $*$-algebras with application to semidefinite programming, Optimization Online, 2009. (E-Print ID : 2009-02-2244) `http://www.optimization-online.org/DB_HTML/2009/02/2244.html`

[7] E. de Klerk, D.V. Pasechnik and A. Schrijver: Reduction of symmetric semidefinite programs using the regular $*$-representation, Mathematical Programming, Series B, Vol. 109 (2007), pp. 613–624.

[8] E. de Klerk and R. Sotirov: Exploiting group symmetry in semidefinite programming relaxations of the quadratic assignment, Mathematical Programming, Series A, 2008.

[9] K. Gatermann and P.A. Parrilo: Symmetry groups, semidefinite programs, and sums of squares, Journal of Pure and Applied Algebra, Vol. 192 (2004), pp. 95–128.

[10] M. Kojima, S. Kojima and S. Hara: Linear algebra for semidefinite programming, Research Report B-290, Tokyo Institute of Technology, October 1994; also in RIMS Kokyuroku 1004, Kyoto University, pp. 1–23, 1997.

[11] T. Maehara and K. Murota: A numerical algorithm for block-diagonal decomposition of matrix $*$-algebras with general irreducible components, Japan Journal of Industrial and Applied Mathematics, to appear.

[12] K. Murota, Y. Kanno, M. Kojima and S. Kojima: A numerical algorithm for block-diagonal decomposition of matrix $*$-algebras with application to semidefinite programming, Japan Journal of Industrial and Applied Mathematics, to appear.

[13] F.J. Theis: Toward a general independent subspace analysis, In Proceedings of Neural Information Processing Systems 2006. pp. 1361–1368.

[14] J.H.M. Wedderburn: Lectures on Matrices, American Mathematical Society, New York, 1934; Dover, Mineola, N.Y., 2005.