COMPLEX NON-ORTHOGONAL JOINT DIAGONALIZATION WITH SUCCESSIVE GIVENS AND HYPERBOLIC ROTATIONS

Xiao-Feng Gong, Ke Wang, Qiu-Hua Lin

Faculty of Electronic Information and Electrical Engineering
Dalian University of Technology, Dalian 116023, China
E-mail: xfgong@dlut.edu.cn

ABSTRACT

Complex blind source separation (BSS) received growing interests in many practical applications in the past decades, and non-orthogonal joint diagonalization (JD) of a set of complex matrices plays an instrumental role in solving these problems. In this paper, we propose a new complex non-orthogonal JD algorithm. This algorithm successively finds the optimal Givens and hyperbolic rotation matrices that constitute the elementary rotation matrix in each iteration in an alternating manner. It does not require the target matrices to be Hermitian, and thus could be well adapted to BSS problems that involve fourth-order cumulant slices or time-lagged covariance matrices. Simulations are provided to compare the proposed algorithm with other JD algorithms.

Index Terms — Blind source separation, Complex non-orthogonal joint diagonalization, Givens, Hyperbolic

1. INTRODUCTION

Blind source separation (BSS) aims at separating sources from their mixtures with no prior knowledge on the mixing process and sources other than some practical assumptions (e.g. the source independence). The early BSS works are mainly based on real-valued algorithms, while in the past decades, complex BSS [1, 8] has attracted growing interests as the need for complex signal processing increases.

Many BSS methods undergo a joint diagonalization (JD) step upon a set of matrices $\mathcal{M} = \{\boldsymbol{M}_1, ..., \boldsymbol{M}_K\}$, for the blind identification of the mixing matrix. More exactly, the target matrices are usually established as fourth-order cumulant slices [1] or time-lagged covariance matrices[2], that share the following JD structure by assuming source uncorrelation as well as non-stationarity, or source independence:

$$\boldsymbol{M}_k = \boldsymbol{A}\boldsymbol{D}_k\boldsymbol{A}^H \tag{1}$$

where $\boldsymbol{D}_k$ is the *kth* diagonal matrix and $\boldsymbol{A}$ is the mixing matrix. The superscript 'H' denotes conjugated transpose. As a result, fitting this JD structure yields the estimate of $\boldsymbol{A}$.

There are enormous JD algorithms in the open literature. The early ones, such as Cardoso's Jacobi-like method [1], are usually designed for unitary matrices, and thus prewhitening must be added to orthogonalize the target matrices. However, prewhitening is always approximate due to the estimation errors for the target matrices, and these errors could not be corrected in the JD stage that follows [3]. As a result, the non-orthogonal schemes without prewhitening have been widely advocated [4-12]. These algorithms are usually of iterative type, aiming at minimizing certain cost functions that measure the fitting of the JD structure. To name a few, the weighted least squares (WLS) criterion is used in [4-7] and JD is formulated as a set of subspace fitting problems. The work in [8] uses information theoretic criterion for JD of positive definite matrices. Minimization of the sum of off-diagonal squared norms is also widely adapted [1, 9-12].

Among the afore-mentioned JD algorithms, those using successive rotations are of a particular kind [1, 10-12]. These methods often use parameterized elementary rotation matrix in each iteration to update the estimate of the mixing matrix, and thus it is only needed to find several optimal parameters in each iteration. As such, these methods are often of closed-form in each iteration and could provide stable solutions that are insensitive to initializations. The orthogonal JD using Givens rotations is addressed in [1], while the works in [10-12] extend the idea of [1] to the non-orthogonal context by factorizing the elementary rotation matrix for each iteration into the product of Givens rotation matrix and one symmetric non-unitary matrix (hyperbolic rotation [10, 11] or triangular matrix [12]). However, [10-12] only considered the real-valued case, and could not be adopted in practical applications that involve complex non-orthogonal JD.

In this paper, we extend the methodology that adopts successive Givens and hyperbolic rotations [10, 11] to the complex domain.We note that this extension is not trivial as it involves optimization over more parameters than the real-valued case. In the rest of the paper, section 2 presents the proposed algorithm, which is compared with other methods via simulations in section 3. Section 4 concludes this paper.

This work is supported by Fundamental Research Fund for Central Universities of China, Doctoral Fund of Ministry of Education of China under grant 20110041120019, and National Natural Science Foundation of China under grants 60971097, 61072098, 61105008.

2. THE PROPOSED ALGORITHM

We assume a set of complex-valued square matrices $\mathcal{M}' = \{\boldsymbol{M}_1, \boldsymbol{M}_2, ..., \boldsymbol{M}_K\}$ share the JD structure as given in (1). The mixing matrix $\boldsymbol{A}$ is square and full rank, and $\boldsymbol{A}$, $\boldsymbol{M}_k$, $\boldsymbol{D}_k$ are of size $N \times N$. Instead of using $\mathcal{M}'$ for JD, we use a new matrix set formed by augmenting $\mathcal{M}'$ as: $\mathcal{M} = \{\boldsymbol{M}_1, ..., \boldsymbol{M}_K, \boldsymbol{M}_{K+1}, ..., \boldsymbol{M}_{2K}\}$, where $\boldsymbol{M}_{K+l} = \boldsymbol{M}_l^H$ for $l = 1, 2, ..., K$. Denoting $\mathrm{off}(\boldsymbol{P}) \triangleq \sum_{1 \le i \ne j \le N} |p_{i,j}|^2$ for $\boldsymbol{P} \in \mathrm{C}^{N \times N}$, we propose to tackle the JD problem by minimizing the sum of off-diagonal squared norms as follows:

$$\boldsymbol{A} = \arg\min_{\boldsymbol{A}} \sum_{k=1}^{2K} \mathrm{off}(\boldsymbol{A}^{-1} \boldsymbol{M}_k \boldsymbol{A}^{-H}) \tag{2}$$

In successive rotation based algorithms, the estimates for $\boldsymbol{A}$ and $\boldsymbol{M}_k$ are updated with elementary rotation matrix $\boldsymbol{T}_{(i,j)}$ for each index pair (i, j), $1 \le i < j \le N$, as follows:

$$\boldsymbol{M}_{k,new} = \boldsymbol{T}_{(i,j)}^H \boldsymbol{M}_{k,old} \boldsymbol{T}_{(i,j)}, \quad \boldsymbol{A}_{new} = \boldsymbol{A}_{old} \boldsymbol{T}_{(i,j)}^{-H} \tag{3}$$

where $\boldsymbol{M}_{k,new}$ and $\boldsymbol{A}_{new}$ denote the updates of $\boldsymbol{M}_k$ and $\boldsymbol{A}$ in the current iteration, and $\boldsymbol{M}_{k,old}$ and $\boldsymbol{A}_{old}$ denote the results obtained in the previous iteration, $k = 1, ..., 2K$. $\boldsymbol{T}_{(i,j)}$ equals the identity matrix except the entries indexed (i,i), (i, j), (j,i), and (j, j). The goal is then to find an optimal $\boldsymbol{T}_{(i,j)}$ to minimize $\sum_{k=1}^{2K} \mathrm{off}(\boldsymbol{M}_{k,new})$ for each iteration.

We note that any complex non-singular matrix could be factorized into the product of a Hermitian matrix and a unitary matrix, and thus $\boldsymbol{T}_{(i,j)}$ could be rewritten as $\boldsymbol{T}_{(i,j)} = \boldsymbol{H}_{\alpha_{ij},\beta_{ij}} \boldsymbol{G}_{\theta_{ij},\varphi_{ij}}$, with $\boldsymbol{G}_{\theta_{ij},\varphi_{ij}}$ being the unitary Givens rotation matrix, and $\boldsymbol{H}_{\alpha_{ij},\beta_{ij}}$ the Hermitian hyperbolic rotation matrix:

$$\boldsymbol{G}_{\theta_{ij},\varphi_{ij}} \triangleq \begin{bmatrix} \boldsymbol{I}_{i-1} & \vdots & 0 & \vdots & 0 \\ \cdots & \cos\theta_{ij} & \cdots & \mathrm{e}^{-\vec{\mathrm{i}}\varphi_{ij}} \sin\theta_{ij} & \cdots \\ 0 & \vdots & \boldsymbol{I}_{j-i-1} & \vdots & 0 \\ \cdots & -\mathrm{e}^{\vec{\mathrm{i}}\varphi_{ij}} \sin\theta_{ij} & \cdots & \cos\theta_{ij} & \cdots \\ 0 & \vdots & 0 & \vdots & \boldsymbol{I}_{N-j} \end{bmatrix}$$

$$\boldsymbol{H}_{\alpha_{ij},\beta_{ij}} \triangleq \begin{bmatrix} \boldsymbol{I}_{i-1} & \vdots & 0 & \vdots & 0 \\ \cdots & \cosh\alpha_{ij} & \cdots & \mathrm{e}^{-\vec{\mathrm{i}}\beta_{ij}} \sinh\alpha_{ij} & \cdots \\ 0 & \vdots & \boldsymbol{I}_{j-i-1} & \vdots & 0 \\ \cdots & \mathrm{e}^{\vec{\mathrm{i}}\beta_{ij}} \sinh\alpha_{ij} & \cdots & \cosh\alpha_{ij} & \cdots \\ 0 & \vdots & 0 & \vdots & \boldsymbol{I}_{N-j} \end{bmatrix} \tag{4}$$

where '$\vec{\mathrm{i}}$' is the imaginary unit, '$\cosh$' denotes hyperbolic cosine and '$\sinh$' denotes hyperbolic sine. As a result, the problem now amounts to finding optimal parameters $\tilde{\theta}_{ij}$, $\tilde{\varphi}_{ij}$, $\tilde{\alpha}_{ij}$, $\tilde{\beta}_{ij}$ in each iteration to minimize $\sum_{k=1}^{2K} \mathrm{off}(\boldsymbol{M}_{k,new})$. Noting that it might be difficult to find all the parameters simultaneously, we adopt the following alternating scheme in each iteration:

$$\{\tilde{\theta}_{ij}, \tilde{\varphi}_{ij}\} = \arg\min_{\theta_{ij},\varphi_{ij}} \sum_{k=1}^{2K} \mathrm{off}(\boldsymbol{G}_{\theta_{ij},\varphi_{ij}}^H \boldsymbol{M}_k \boldsymbol{G}_{\theta_{ij},\varphi_{ij}}) \tag{5.a}$$

$$\{\tilde{\alpha}_{ij}, \tilde{\beta}_{ij}\} = \arg\min_{\alpha_{ij},\beta_{ij}} \sum_{k=1}^{2K} \mathrm{off}(\boldsymbol{H}_{\alpha_{ij},\beta_{ij}}^H \boldsymbol{N}_{k,new} \boldsymbol{H}_{\alpha_{ij},\beta_{ij}}) \tag{5.b}$$

where $\boldsymbol{N}_{k,new} \triangleq \boldsymbol{G}_{\tilde{\theta}_{ij},\tilde{\varphi}_{ij}}^H \boldsymbol{M}_k \boldsymbol{G}_{\tilde{\theta}_{ij},\tilde{\varphi}_{ij}}$. Moreover, as (5.a) could be solved via Cardoso's Jacobi-like algorithm [1], the problem now amounts to solving (5.b).

As indicated in (5.b), all the off-diagonal elements are involved in finding the optimal hyperbolic rotation matrix. However, considering all the off-diagonal elements would result in a difficult minimization problem and thus we propose to approximate $\mathrm{off}(\boldsymbol{H}_{\alpha_{ij},\beta_{ij}}^H \boldsymbol{N}_{k,new} \boldsymbol{H}_{\alpha_{ij},\beta_{ij}})$ in (5.b) by considering only two specific off-diagonal elements: the elements indexed (i, j) and (j,i) in $\boldsymbol{H}_{\alpha_{ij},\beta_{ij}}^H \boldsymbol{N}_{k,new} \boldsymbol{H}_{\alpha_{ij},\beta_{ij}}$. The reason for doing so is that the (*i*,*j*)th and (*j*,*i*)th elements are twice affected by $\boldsymbol{H}_{\alpha_{ij},\beta_{ij}}$, and thus contribute the most to the changes on the sum of off-diagonal squared norms [10, 11]. Hence, (5.b) is approximated as:

$$\begin{aligned} \{\tilde{\alpha}_{ij}, \tilde{\beta}_{ij}\} = \arg\min_{\alpha_{ij},\beta_{ij}} \sum_{k=1}^{2K} & |(\boldsymbol{H}_{\alpha_{ij},\beta_{ij}}^H \boldsymbol{N}_{k,new} \boldsymbol{H}_{\alpha_{ij},\beta_{ij}})_{i,j}|^2 \\ & + |(\boldsymbol{H}_{\alpha_{ij},\beta_{ij}}^H \boldsymbol{N}_{k,new} \boldsymbol{H}_{\alpha_{ij},\beta_{ij}})_{j,i}|^2 \end{aligned} \tag{6}$$

We denote the elements indexed (i, j) in $\boldsymbol{H}_{\alpha_{ij},\beta_{ij}}^H \boldsymbol{N}_{k,new} \boldsymbol{H}_{\alpha_{ij},\beta_{ij}}$ and $\boldsymbol{N}_{k,new}$ as $\tilde{m}_{k,(i,j)}$ and $n_{k,(i,j)}$, respectively, and $\xi \triangleq 2\sum_{k=1}^{2K}(|\tilde{m}_{k,(i,j)}|^2 + |\tilde{m}_{k,(j,i)}|^2 + |n_{k,(i,j)}|^2 + |n_{k,(j,i)}|^2)$, then the minimization problem in (6) is equal to minimizing ξ as $|n_{k,(i,j)}|^2 + |n_{k,(j,i)}|^2$ could be considered as a constant. In addition, we further rewrite ξ as:

$$\begin{aligned} \xi = -2\sum_{k=1}^{2K}\psi_k + 2\sum_{k=1}^{2K} [& (\tilde{m}_{k,(i,j)} - n_{k,(i,j)})(\tilde{m}_{k,(i,j)}^* - n_{k,(i,j)}^*) \\ & + (\tilde{m}_{k,(j,i)} + n_{k,(j,i)})(\tilde{m}_{k,(j,i)}^* + n_{k,(j,i)}^*)] \end{aligned} \tag{7}$$

where $\psi_k \triangleq \tilde{m}_{k,(i,j)} n_{k,(i,j)}^* + n_{k,(i,j)} \tilde{m}_{k,(i,j)}^* - \tilde{m}_{k,(j,i)} n_{k,(j,i)}^* - n_{k,(j,i)} \tilde{m}_{k,(j,i)}^*$. We note that $\tilde{m}_{k,(i,j)} = \tilde{m}_{k+K,(j,i)}^*$, and $n_{k,(i,j)} = n_{k+K,(j,i)}^*$. As a result, $\psi_k + \psi_{k+K} = 0$ and the first term of (7) vanishes. By further denoting $\tilde{m}_k \triangleq \tilde{m}_{k,(i,j)} = \tilde{m}_{k+K,(j,i)}^*$, and $n_k \triangleq n_{k,(i,j)} = n_{k+K,(j,i)}^*$, (7) could be rewritten as:

$$\begin{aligned} \xi = \sum_{k=1}^{K} \{ & [\tilde{m}_k^* + n_k^* + \mathrm{e}^{2\vec{\mathrm{i}}\beta_{ij}}(\tilde{m}_k - n_k)][\tilde{m}_k + n_k + \mathrm{e}^{-2\vec{\mathrm{i}}\beta_{ij}}(\tilde{m}_k^* - n_k^*)] \\ & + [\tilde{m}_k^* + n_k^* - \mathrm{e}^{2\vec{\mathrm{i}}\beta_{ij}}(\tilde{m}_k - n_k)][\tilde{m}_k + n_k - \mathrm{e}^{-2\vec{\mathrm{i}}\beta_{ij}}(\tilde{m}_k^* - n_k^*)]\} \end{aligned} \tag{8}$$

As $\tilde{m}_k = \cosh^2\alpha_{ij} n_k - \mathrm{e}^{-\vec{\mathrm{i}}\beta_{ij}} \sinh\alpha_{ij} \cosh\alpha_{ij} (n_{k,(i,i)} + n_{k,(j,j)}) + \mathrm{e}^{-2\vec{\mathrm{i}}\beta_{ij}} \sinh^2\alpha_{ij} n_k^*$, where $n_{k,(i,i)}$ and $n_{k,(j,j)}$ are the *i*th and *j*th diagonal elements of $\boldsymbol{N}_{k,new}$, respectively, the second term on the right side of (8) is equal to $4|n_k|^2$. Therefore, the problem given in (6) is reduced to the minimization of $\chi \triangleq \sum_{k=1}^{K} [\tilde{m}_k^* + n_k^* + \mathrm{e}^{2\vec{\mathrm{i}}\beta_{ij}}(\tilde{m}_k - n_k)][\tilde{m}_k + n_k + \mathrm{e}^{-2\vec{\mathrm{i}}\beta_{ij}}(\tilde{m}_k^* - n_k^*)]$. In addition, forming a vector $\boldsymbol{y} \in \mathrm{C}^K$ with its *k*th entry: $y_k \triangleq [\tilde{m}_k^* + n_k^* + \mathrm{e}^{2\vec{\mathrm{i}}\beta_{ij}}(\tilde{m}_k - n_k)]$, we have $\chi = \boldsymbol{y}^H \boldsymbol{y}$. As a result, substituting $\tilde{m}_k$ into $\boldsymbol{y}$ yields the following:

$$\boldsymbol{y} = \underbrace{\begin{bmatrix} -n_{1,(i,i)} - n_{1,(j,j)} & n_1 & n_1^* \\ -n_{2,(i,i)} - n_{2,(j,j)} & n_2 & n_2^* \\ \vdots & \vdots & \vdots \\ -n_{K,(i,i)} - n_{K,(j,j)} & n_K & n_K^* \end{bmatrix}}_{\boldsymbol{W}} \underbrace{\begin{bmatrix} \mathrm{e}^{\vec{\mathrm{i}}\beta_{ij}} \sinh 2\alpha_{ij} \\ \mathrm{e}^{2\vec{\mathrm{i}}\beta_{ij}} (\cosh 2\alpha_{ij} - 1) \\ \cosh 2\alpha_{ij} + 1 \end{bmatrix}}_{\boldsymbol{x}} \tag{9}$$

Further noting that $\boldsymbol{x}^H \boldsymbol{J} \boldsymbol{x} = 1$, with $\boldsymbol{J} = \mathrm{diag}(-\tfrac{1}{2}, \tfrac{1}{4}, \tfrac{1}{4})$, where 'diag' makes a diagonal matrix with given inputs, we could obtain the optimal vector $\tilde{\boldsymbol{x}}$ by solving the following constrained minimization problem:

$$\begin{cases} \tilde{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}} \boldsymbol{x}^H \boldsymbol{W}^H \boldsymbol{W}\boldsymbol{x} \\ s.t. \quad \boldsymbol{x}^H \boldsymbol{J}\boldsymbol{x} = 1 \end{cases} \tag{10}$$

which can be solved using the Lagrange multipliers strategy. More exactly, by constructing a cost function incorporating the equality constraint as $h(\boldsymbol{x}) = \boldsymbol{x}^H\boldsymbol{W}^H\boldsymbol{W}\boldsymbol{x} + \lambda(\boldsymbol{x}^H\boldsymbol{J}\boldsymbol{x} - 1)$, and letting its derivative with regards to $\boldsymbol{x}^*$ equal zero, we have $\boldsymbol{W}^H\boldsymbol{W}\boldsymbol{x} = -\lambda\boldsymbol{J}\boldsymbol{x}$, which indicates that $\boldsymbol{x}$ could be obtained as the generalized eigenvector of matrix pencil $(\boldsymbol{W}^H\boldsymbol{W}, -\boldsymbol{J})$ associated with the smallest eigenvalue. Hence, the optimal hyperbolic rotation angles could be obtained as:

$$\begin{aligned} \cosh\alpha_{ij} &= \sqrt{\tilde{\boldsymbol{x}}(3)/2} \\ e^{i\beta_{ij}}\sinh\alpha_{ij} &= \tilde{\boldsymbol{x}}(1)/2\cosh\alpha_{ij} \end{aligned} \tag{11}$$

We note that the proposed complex non-orthogonal JD (CNJD) does not require the target matrices to be Hermitian. Moreover, we stop the iterations of CNJD when $\min(\theta, \varphi, \alpha_{ij})$ is smaller than threshold τ. CNJD is summarized as follows:

- **Input**: A set of square $N \times N$ matrices $\mathcal{M} = \{\boldsymbol{M}_1,...,\boldsymbol{M}_K, \boldsymbol{M}_{K+1},...,\boldsymbol{M}_{2K}\}$, $\boldsymbol{M}_{K+l} = \boldsymbol{M}_l^H$, $l = 1,...,K$, and a threshold τ
- **Output**: The mixing matrix estimate $\boldsymbol{A}$
- **Implementation:**

  $\boldsymbol{A} \leftarrow \boldsymbol{I}_N$

  while $\max(|\sinh\alpha_{ij}|, |\sin\theta_{ij}|) \geq \tau$ **do**

  for all $1 \leq i < j \leq N$ **do**

  - Obtain optimal Givens angles $\theta_{ij}, \varphi_{ij}$ via the Jacobi-like algorithm [1] and the Givens rotation matrix $\boldsymbol{G}_{\theta_{ij},\varphi_{ij}}$ by (4)

    for $k = 1$ to $2K$ **do** $\boldsymbol{N}_k \leftarrow \boldsymbol{G}^H_{\theta_{ij},\varphi_{ij}}\boldsymbol{M}_k\boldsymbol{G}_{\theta_{ij},\varphi_{ij}}$ **end for**
  - Compute the eigenvector $\boldsymbol{x}$ of $(\boldsymbol{W}^H\boldsymbol{W}, \mathrm{diag}(\tfrac{1}{2}, -\tfrac{1}{4}, -\tfrac{1}{4}))$ of the smallest eigenvalue, where $\boldsymbol{W}$ is obtained by (9).

    if $\boldsymbol{x}(3) < 0$ **then** $\boldsymbol{x} \leftarrow -\boldsymbol{x}$ **end if**

    $\cosh\alpha_{ij} \leftarrow \sqrt{\tilde{\boldsymbol{x}}(3)/2}$, $e^{i\beta_{ij}}\sinh\alpha_{ij} \leftarrow \tilde{\boldsymbol{x}}(1)/2\cosh\alpha_{ij}$. Obtain the hyperbolic rotation matrix $\boldsymbol{H}_{\alpha_{ij},\beta_{ij}}$ by (4)
  - **for** $k = 1$ to K **do**

    $\boldsymbol{M}_k \leftarrow \boldsymbol{H}^H_{\alpha_{ij},\beta_{ij}}\boldsymbol{N}_k\boldsymbol{H}_{\alpha_{ij},\beta_{ij}}, \boldsymbol{T}_{(i,j)} \leftarrow \boldsymbol{H}_{\alpha_{ij},\beta_{ij}}\boldsymbol{G}_{\theta_{ij},\varphi_{ij}}, \boldsymbol{A} \leftarrow \boldsymbol{A}\boldsymbol{T}^{-H}_{(i,j)}$

    end for

  end for

  end while

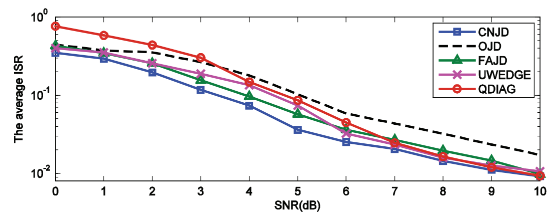
3. SIMULATIONS

We assume three far-field narrow band signals impinge upon a uniform linear array of three sensors with spacing equaling half the wavelength of incidences. The sensor signals can be modeled as: $\boldsymbol{x}(t) = \sum_{m=1}^{3}\boldsymbol{a}(\theta_m)s_m(t) + \boldsymbol{n}(t)$, where $\boldsymbol{a}(\theta_m) \triangleq [1, e^{i\pi\sin\theta_m}, e^{i2\pi\sin\theta_m}]$ is the mth mixing vector, $s_m(t)$ is the mth source, and $\boldsymbol{n}(t)$ is the noise [2]. The impinging angles of sources are $\theta_1 = 40°$, $\theta_2 = 80°$ and $\theta_3 = 12°$, respectively. As a result, JD over fourth-order or second-order cumulant matrices could be used to identify the mixing vectors blindly.

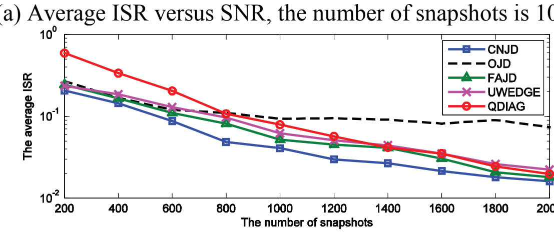
The proposed CNJD method is compared with Cardoso's Jacobi-like orthogonal JD (OJD) [1], Li's fast approximate JD (FAJD) [5], Tichavsky and Yeredor's uniformly weighted exhaustive diagonalization by Gaussian iteration (UWEDGE) [6], and Vollgraf's quadratic optimization for approximate matrix diagonalization (QDIAG) [7]. In addition, we perform prewhitening for OJD due to its orthogonal nature, and use uniform weights for FAJD, UWEDGE, and QDIAG. The positive definite matrix demanded by QDIAG is the identity matrix. We note that using properly designed non-uniform weights may improve the performance of these algorithms, which is out of the scope of this paper. In addition, OJD and CNJD are initialized with identity matrix, while UWEDGE and FAJD use conventional OJD outputs as initial guesses. The average interference-to-signal ratio [4] (ISR) is used to measure the accuracy of the estimates of the mixing matrix[2]. The signal-to-noise ratio (SNR) is $SNR \triangleq 10\log_{10}(p_\varepsilon^{-1}p_s)$, where p_s is the signal power and p_ε is the noise power.

In the first simulation, we consider BSS based on JD of fourth-order cumulant slices. The incidences are assumed to be random phase signals, and the noise is Gaussian white. The entire set of fourth-order cumulant matrices are used for CNJD, FAJD, UWEDGE, and QDIAG. Yet the scheme to extract principle eigenmatrices [1] is adopted for OJD.

We fix the number of snapshots to 1000, let SNR vary from 0~10dB, and plot the average ISR curves against SNR in Figure 1.(a). Then we fix SNR to 5dB and let the number of snapshots vary from 200 ~ 2000. The average ISR curves against the number of snapshots are plotted in Figure 1.(b). The results are obtained from 100 Monte-Carlo runs.



(a) Average ISR versus SNR, the number of snapshots is 1000



(b) Average ISR versus the number of snapshots, SNR is 5dB

Figure 1. Comparison of CNJD, OJD, FAJD, UWEDGE, and QDIAG in BSS based on fourth-order cumulants

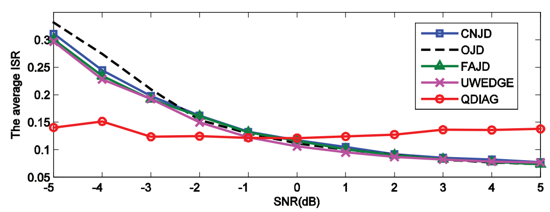
Figure 1 demonstrates that OJD generally underperforms non-orthogonal JD. Moreover, the proposed CNJD method yields the most precise estimates of the mixing matrix, and

[2] Define $\boldsymbol{P} \triangleq \tilde{\boldsymbol{A}}^{-1}\boldsymbol{A}$, where $\tilde{\boldsymbol{A}}$ and $\boldsymbol{A}$ are the estimated and true mixing matrices, respectively, then ISR for each row of $\boldsymbol{P}$ is the sum of squares of all the elements except the largest, divided by the square of the largest one in that row. The average ISR is the average of ISR's for all rows.

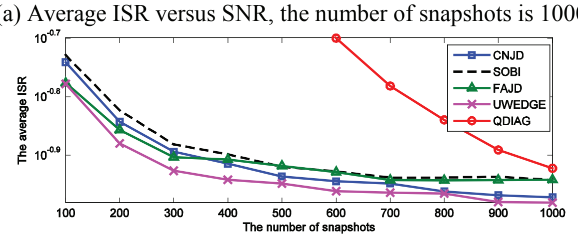
thus is able to provide improved robustness to the errors caused by noise and finite sampling in the context of fourth-order BSS than other JD algorithms.

In the second simulation, we consider second-order BSS that fulfills blind identification of the mixing matrix by JD of time-lagged covariance matrices: $\boldsymbol{R}_{k,l} = F^{-1}\boldsymbol{X}_k\boldsymbol{X}_l^H$, where $\boldsymbol{X}_k$ is the kth time-lagged frame of the array signal: $\boldsymbol{X}_k \triangleq [\boldsymbol{x}((k-1)(1-\xi)F+1),\ldots,\boldsymbol{x}((k-1)(1-\xi)F+F)]$, F is the frame length, and $0 < \xi \leq 1$ is the overlapping factor [8].

We assume the three sources are uncorrelated signals of different spectra obtained by filtering random phase signals by AR models of order 1 with coefficients $\rho_1 = 0.85e^{\vec{i}\pi/3}$, $\rho_2 = 0.85e^{\vec{i}\pi/6}$, and $\rho_3 = 0.85e^{\vec{i}2\pi/3}$. The noise is Gaussian white. The principle eigenmatrices [1] obtained from the entire set of time-lagged covariance matrices are used for all the compared algorithms as target matrices. We fix the number of snapshots to $T = 1000$, set the overlapping factor $\xi = 0.75$, the frame length $F = 100$, and let SNR vary from -5dB ~ 5dB. The average ISR curves versus SNR are plotted in Figure 2.(a). Then we fix SNR to 0dB, set the frame length $F = 10^{-1}T$, let T vary from 100 ~ 1000, and plot the average ISR curves versus the number of snapshots in Figure 2.(b). The results are obtained from 100 Monte-Carlo runs.



(a) Average ISR versus SNR, the number of snapshots is 1000



(b) Average ISR versus the number of snapshots, SNR is 0dB

Figure 2. Comparison of CNJD, JADE, FAJD, UWEDGE, and QDIAG in BSS based on fourth-order cumulants

Figure 2.(a) shows that CNJD provides close accuracy to FAJD and UWEDGE, which is superior to OJD for low SNR (-5dB ~ -3dB) and to QDIAG when SNR exceeds 0dB. For fixed SNR, Figure 2.(b) shows that CNJD performs closely to FAJD and OJD for short data, while approaches UWEDGE as the number of snapshots increases. Moreover, CNJD, OJD, FAJD, and UWEDGE outperforms QDIAG in the presence of the errors introduced by short data length.

4. CONCLUSION

This paper presents a new complex non-orthogonal joint diagonalization algorithm (CNJD) using successive rotations, where the elementary rotation matrix in each iteration is obtained from alternatively updated Givens and hyperbolic rotations. This algorithm does not require the target matrices be Hermitian and thus could be used in complex BSS applications that involve time-lagged covariance matrices or fourth-order cumulant slices. Simulations show that CNJD is able to provide more accurate estimates of the mixing matrix for fourth-order BSS, and almost equally accurate estimates of the mixing matrix for second-order BSS, compared with other complex joint diagonalization algorithms.

5. REFERENCES

[1] J. -F. Cardoso and A. Souloumiac, “Blind beamforming for non-Gaussian signals,” *Proc. IEE.-F,* vol. 140, no. 6, pp. 362-370, 1993.

[2] A. Belouchrani, K. Abed-Meraim, J. -F. Cardoso, and E. Moulines, “A blind source separation technique using second-order statistics,” *IEEE Trans. Signal Process.*, vol. 45, no. 2, pp. 434-444, Feb. 1997.

[3] A. Souloumiac, “Joint diagonalization: is non-orthogonal always preferable to orthogonal?,” *Proc. CAMSAP’2009*, Radisson Aruba Resort, Casino & Spa Aruba, Dutch Antilles, pp. 305-308, Dec. 13-16, 2009.

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

[5] 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.

[6] 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.

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

[8] D-T. Pham, C. Serviére, and H. Boumaraf, “Blind separation of speech mixtures based on nonstationarity,” *Proc. ISSPA’2003*, Paris, France, pp. 73-76, Jul. 1-4, 2003.

[9] G. Hori, “Joint diagonalization and matrix differential equations,” *Proc. NOLTA’99*, Hawaii, USA, pp. 675-678, Nov. 28 - Dec. 2, 1999.

[10] A. Souloumiac, “Nonorthogonal joint diagonalization by combining givens and hyperbolic rotations,” *IEEE Trans. Signal Process.*, vol. 57, no. 6, pp. 2222-2231, Jun. 2009.

[11] X. Luciani and L. Albera, “Joint eigenvalue decomposition using polar matrix factorization,” *Proc. LVA/ICA’2010*, St.Malo, France, pp. 555-562, Sep. 27-30, 2010.

[12] B. Afsari, “Simple LU and QR based non-orthogonal matrix joint diagonalization,” *Proc. ICA’2006*, Charleston, USA, pp. 1-7, Mar. 5-8, 2006.