

Joint Eigenvalue Decomposition Using Polar Matrix Factorization

Xavier Luciani^{1,2} and Laurent Albera^{1,2}

¹ Inserm, UMR 642, Rennes, F-35000, France

² Université de Rennes 1, LTSI, Rennes, F-35000, France

<http://perso.univ-rennes1.fr/laurent.albera/>

Abstract. In this paper we propose a new algorithm for the joint eigenvalue decomposition of a set of real non-defective matrices. Our approach resorts to a Jacobi-like procedure based on polar matrix decomposition. We introduce a new criterion in this context for the optimization of the hyperbolic matrices, giving birth to an original algorithm called JD_TM. This algorithm is described in detail and a comparison study with reference algorithms is performed. Comparison results show that our approach provides quicker and more accurate results in all the considered situations.

Keywords: Joint diagonalization by similarity, joint eigenvalue decomposition, Jacobi method, polar matrix decomposition.

1 Introduction

In this study, we investigate the problem of Joint EigenValue Decomposition (JEVD) of a set of real matrices, which is encountered in different contexts such as 2-D DOA estimation [1], joint angle-delay estimation [2], multidimensional harmonic retrieval [3], Independent Component Analysis (ICA) [4] or Multi-way analysis [5]. JEVD consists in finding an eigenvector matrix \mathbf{A} from a set of non-defective matrices $\mathbf{M}^{(k)}$ verifying:

$$\forall k = 1 \dots K, \quad \mathbf{M}^{(k)} = \mathbf{A} \mathbf{D}^{(k)} \mathbf{A}^{-1}, \quad (1)$$

where the K diagonal matrices $\mathbf{D}^{(k)}$ are unknown.

This problem should not be confused with the classical problem of Joint Diagonalization by Congruence (JDC), for which \mathbf{A}^{-1} is replaced by \mathbf{A}^\top , except when \mathbf{A} is an orthogonal or unitary matrix [6]. JDC is the core of many ICA algorithms [7,8]. A large majority of these algorithms resorts to a suitable factorization of \mathbf{A} , performed by means of a Jacobi-like procedure. Such an approach has been naturally adapted to JEVD, although few papers have addressed this problem. Two main kinds of Jacobi-like algorithms have been developed in this context, based on different matrix factorizations. Originally, several authors had recourse to the QR factorization of \mathbf{A} in order to compute the different sets of eigenvalues [3,9]. Arguing that these QR-algorithms suffer from convergence

problems, Fu and Gao proposed an effective sh-rt algorithm [10] based on the polar decomposition. Indeed the polar decomposition has been used favourably for eigenvalue decomposition purpose since a long time [11, 12, 13] and also for JDC [14]. In a recent paper, Iferroudjene *et al.* introduced an alternative version of the sh-rt algorithm called JUST [15]. Our turn, we propose in this work an improvement of the sh-rt technique with significant numerical results.

2 Notations

In the following, scalars, vectors and matrices are denoted by lower case (a), lower case boldface (\mathbf{a}) and upper case boldface (\mathbf{A}) letters, respectively. The i -th entry of vector \mathbf{a} is denoted by a_i while A_{ij} is the (i, j) -th component of matrix \mathbf{A} . Diagonal matrices are denoted by \mathbf{D} , Givens and hyperbolic rotation matrices are denoted by \mathbf{G} and \mathbf{H} , respectively. For instance $\mathbf{G}(\theta_{ij})$ and $\mathbf{H}(\phi_{ij})$ are equal to the identity matrix, at the exception of the following components:

$$G(\theta_{ij})_{ii} = \cos(\theta_{ij}); \quad G(\theta_{ij})_{ij} = \sin(\theta_{ij}). \quad H(\phi_{ij})_{ii} = \cosh(\phi_{ij}); \quad H(\phi_{ij})_{ij} = \sinh(\phi_{ij}). \\ G(\theta_{ij})_{ji} = -\sin(\theta_{ij}); \quad G(\theta_{ij})_{jj} = \cos(\theta_{ij}). \quad H(\phi_{ij})_{ji} = \sinh(\phi_{ij}); \quad H(\phi_{ij})_{jj} = \cosh(\phi_{ij}).$$

3 A Novel Algorithm for JEVD

3.1 A Jacobi-Like Computation of the Polar Matrix Decomposition

In this subsection, all matrices are square matrices of dimensions N . Polar matrix decomposition states that any non-singular real matrix can be factorized into the product of an orthogonal matrix \mathbf{Q} and a positive symmetric matrix \mathbf{S} . It is well known that \mathbf{Q} can be decomposed into a product of Givens rotation matrices and a unitary diagonal matrix. In the same way, it has been shown that \mathbf{S} can be decomposed into a product of hyperbolic rotation matrices and diagonal matrices [14]. Since (1) admits a scaling indeterminacy, the eigenvector matrix can only be estimated up to a diagonal scaling matrix. Therefore, taking into account that diagonal, hyperbolic and Givens matrices commute, one can reasonably assume that a solution matrix \mathbf{A} can be found as a product of Givens and hyperbolic rotation matrices:

$$\mathbf{A} = \prod_{i=1}^{N-1} \prod_{j=i+1}^N \mathbf{G}(\theta_{ij}) \mathbf{H}(\phi_{ij}). \tag{2}$$

The main point is that any Givens or hyperbolic matrix is defined by only one parameter (angle). Therefore we have now to find a set of $M = N(N - 1)/2$ couple of parameters $\{\theta_{ij}, \phi_{ij}\}_{1 \leq i < j \leq N}$ in order to get (1). Jacobi-like procedures achieve this problem by optimizing a suitable criterion with respect to each parameter, one by one. Consequently, inserting (2) into (1) we get:

$$\forall k = 1 \dots K, \quad \mathbf{D}^{(k)} = \left(\prod_{m=1}^M \mathbf{H}(\phi_m)^{-1} \mathbf{G}(\theta_m)^T \right) \mathbf{M}^{(k)} \left(\prod_{m=1}^M \mathbf{G}(\theta_m) \mathbf{H}(\phi_m) \right), \tag{3}$$

where each index m ($m = 1 \cdots M$) corresponds to a couple (i, j) ($1 \leq i < j \leq N$). Then, the algorithm scheme is simple. It consists in iteratively diagonalizing the $\mathbf{M}^{(k)}$ matrices by a successive optimization with respect to $\mathbf{G}(\theta_m)$ and $\mathbf{H}(\phi_m)$:

$$\forall k = 1 \cdots K, \quad \mathbf{M}^{(k,0)} = \mathbf{M}^{(k)}, \tag{4}$$

$$\forall k = 1 \cdots K, \quad \forall m = 1 \cdots M, \quad \mathbf{N}^{(k,m)} = \mathbf{G}(\theta_m)^\top \mathbf{M}^{(k,m-1)} \mathbf{G}(\theta_m), \tag{5}$$

$$\forall k = 1 \cdots K, \quad \forall m = 1 \cdots M, \quad \mathbf{M}^{(k,m)} = \mathbf{H}(\phi_m)^{-1} \mathbf{N}^{(k,m)} \mathbf{H}(\phi_m). \tag{6}$$

Thereby, at each stage m , the optimal corresponding Givens and hyperbolic matrices are successively computed in order to get K diagonal matrices $\mathbf{M}^{(k,M)}$ at the end of the process.

3.2 Optimization Step

A natural criterion to compute the optimal m -th Givens angle θ_m is thus to minimize the sum of the euclidean norms of the off-diagonal terms of the K matrices $\mathbf{N}^{(k,m)}$:

$$\zeta_G(\theta_m) = \sum_k^K \sum_{\substack{p,q \\ p \neq q}}^{N,N} \left(N_{pq}^{(k,m)} \right)^2. \tag{7}$$

This criterion is the generalization of the original Jacobi criterion to the JD context. Since Givens matrices are orthogonal, the same definition of $\mathbf{N}^{(k,m)}$ holds in both the JDC and JEVD cases and thus the same optimization algorithms can be used. For instance, our proposed algorithm resorts to the same approach than the JAD algorithm described in [16] whereas the sh-rt and JUST algorithms use an other minimization scheme.

Once the optimal Givens matrix $\mathbf{G}(\theta_m)$ is computed, different criteria can be used for the optimal computation of $\mathbf{H}(\phi_m)$. This is the main difference between the three JEVD algorithms. The JUST algorithm resorts to criterion (7) by replacing $\mathbf{N}^{(k,m)}$ by $\mathbf{M}^{(k,m)}$, whereas the sh-rt method aims at minimizing the Frobenius norm of $\mathbf{M}^{(k,m)}$.

Instead of minimizing all the (off-diagonal) entries, we propose to target two particular off-diagonal entries of $\mathbf{M}^{(k,m)}$: if m corresponds to the $(i, j)_{i < j}$ couple, we simply aim at computing the optimal $M_{ij}^{(k,m)}$ and $M_{ji}^{(k,m)}$ components by using a "targeting" hyperbolic matrix. It is noteworthy that the transformation (6) affects the i -th and j -th rows and the i -th and j -th columns of $\mathbf{M}^{k,m}$ but only the (i, j) and the (j, i) components are twice affected by the hyperbolic matrix and its inverse. Hence our choice to focus on the latter. Therefore, we use the following alternative criterion ζ_H^{JDTM} for the computation of the hyperbolic matrix:

$$\zeta_H^{JDTM}(\phi_m) = \sum_k^K \left(M_{ij}^{(k,m)} \right)^2 + \left(M_{ji}^{(k,m)} \right)^2, \tag{8}$$

giving birth to our Joint Diagonalization algorithm based on Targeting hyperbolic Matrices (JDTM). A similar approach has been used with success in a

different context [14]. Note that in the case of Givens matrices we showed that the optimizations of criteria (7) and (8) were mathematically equivalent.

Now, let us look at the components of $\mathbf{M}^{k,m}$. As previously mentioned, we only consider the (i, j) -th and (j, i) -th components which are given by:

$$M_{ij}^{(k,m)} = \left(N_{ii}^{(k,m)} - N_{jj}^{(k,m)} \right) \frac{\sinh(2\phi_m)}{2} + N_{ij}^{(k,m)} \cosh(\phi_m)^2 - N_{ji}^{(k,m)} \sinh(\phi_m)^2, \tag{9}$$

$$M_{ji}^{(k,m)} = \left(N_{jj}^{k,m} - N_{ii}^{(k,m)} \right) \frac{\sinh(2\phi_m)}{2} - N_{ij}^{(k,m)} \sinh(\phi_m)^2 + N_{ji}^{(k,m)} \cosh(\phi_m)^2. \tag{10}$$

Furthermore we can write that:

$$\left(M_{ij}^{(k,m)} \right)^2 + \left(M_{ji}^{(k,m)} \right)^2 = \frac{\left(M_{ij}^{(k,m)} + M_{ji}^{(k,m)} \right)^2}{2} + \frac{\left(M_{ij}^{(k,m)} - M_{ji}^{(k,m)} \right)^2}{2} \tag{11}$$

where the first term of the right-hand side is constant. Indeed, we derive from (9) and (10) the following equality:

$$\frac{\left(M_{ij}^{(k,m)} + M_{ji}^{(k,m)} \right)^2}{2} = \frac{\left(N_{ij}^{(k,m)} + N_{ji}^{(k,m)} \right)^2}{2} \tag{12}$$

where the right-hand side clearly does not depend on ϕ_m . Thereby minimizing ζ_H^{JTDM} is equivalent to minimize the λ function defined by:

$$\lambda(\phi_m) = \sum_k^K \left(M_{ij}^{(k,m)} - M_{ji}^{(k,m)} \right)^2. \tag{13}$$

We denote by $\mathbf{y}^{(m)}$ the column vector of \mathbb{R}^K defined by $y_k^{(m)} = M_{ij}^{(k,m)} - M_{ji}^{(k,m)}$, so that $\lambda(\phi_m) = \mathbf{y}^{(m)\top} \mathbf{y}^{(m)}$. From (9) and (10) we obtain:

$$\mathbf{y}^{(m)} = \mathbf{W}^{(m)} \mathbf{x}(\phi_m), \tag{14}$$

with:

$$\mathbf{W}^{(m)} = \begin{bmatrix} N_{ii}^{1,m} - N_{jj}^{1,m} & N_{ij}^{1,m} - N_{ji}^{1,m} \\ \vdots & \vdots \\ N_{ii}^{K,m} - N_{jj}^{K,m} & N_{ij}^{K,m} - N_{ji}^{K,m} \end{bmatrix}; \mathbf{x}(\phi_m) = \begin{bmatrix} \sinh(2\phi_m) \\ \cosh(2\phi_m) \end{bmatrix}.$$

Now defining the diagonal (2×2) matrix \mathbf{J} such that $J_{11} = -J_{22} = -1$ and observing that $\mathbf{x}(\phi_m)^\top \mathbf{J} \mathbf{x}(\phi_m) = 1$, we have thus to minimize the quantity $\mathbf{x}(\phi_m)^\top \mathbf{W}^{(m)\top} \mathbf{W}^{(m)} \mathbf{x}(\phi_m)$ under the constraint that $\mathbf{x}(\phi_m)^\top \mathbf{J} \mathbf{x}(\phi_m) = 1$. This can be done using the Lagrange multipliers strategy. Thereby, we have to minimize the L function given by:

$$L(\mathbf{x}(\phi_m), \mu(\phi_m)) = \mathbf{x}(\phi_m)^\top \mathbf{W}^{(m)\top} \mathbf{W}^{(m)} \mathbf{x}(\phi_m) - \mu(\phi_m) \mathbf{x}(\phi_m)^\top \mathbf{J} \mathbf{x}(\phi_m). \tag{15}$$

This leads to:

$$\mathbf{J}\mathbf{W}^{(m)\top}\mathbf{W}^{(m)}\mathbf{x}(\phi_m) = \mu(\phi_m)\mathbf{x}(\phi_m). \tag{16}$$

Thus, $\mu(\phi_m)$ and $\mathbf{x}(\phi_m)$ are associated eigenvalue and eigenvector of matrix $\mathbf{J}\mathbf{W}^{(m)\top}\mathbf{W}^{(m)}$ and it is easily shown that $\lambda(\phi_m) = \mu(\phi_m)$. $\mathbf{J}\mathbf{W}^{(m)\top}\mathbf{W}^{(m)}$ is diagonalizable by construction and it can be shown that it has two non-null eigenvalues of opposite sign (the proof is not given here due to lack of space). Since the Gram matrix $\mathbf{W}^{(m)\top}\mathbf{W}^{(m)}$ is a positive semi-definite matrix, $\mathbf{x}(\phi_m)^\top\mathbf{W}^{(m)\top}\mathbf{W}^{(m)}\mathbf{x}(\phi_m)$ is positive and hence $\lambda(\phi_m)$. As a consequence $\mathbf{x}(\phi_m)$ is the eigenvector associated to the positive eigenvalue of $\mathbf{J}\mathbf{W}^{(m)\top}\mathbf{W}^{(m)}$. Finally we have:

$$\phi(m) = \frac{1}{2} \operatorname{atanh} \left(\frac{x(\phi_m)_1}{x(\phi_m)_2} \right). \tag{17}$$

We have just shown how the Givens and hyperbolic matrices are computed by the JD TM algorithm for the M couples (i, j) . Actually, since the previous procedure follows an alternate optimization scheme, it has to be repeated several times before convergence. Each repetition is called a "sweep". In other words, if N_s is the number of sweeps, \mathbf{A} is actually estimated by:

$$\widehat{\mathbf{A}} = \prod_{n_s} \prod_{i=1}^{N_s-1} \prod_{j=i+1}^N \mathbf{G}(\theta_{ij}^{n_s}) \mathbf{H}(\phi_{ij}^{n_s}). \tag{18}$$

This iterative approach is common to all Jacobi-like algorithms. Finally, the K matrices, $\widehat{\mathbf{D}}^{(k)} = \widehat{\mathbf{A}}^{-1} \mathbf{M}^{(k)} \widehat{\mathbf{A}}$ are approximately diagonalized. One can measure the diagonalization achievement by using the following criterion:

$$r_D = \sum_k^K \sum_{\substack{p,q \\ p \neq q}}^{N,N} \left(\widehat{\mathbf{D}}_{pq}^{(k)} \right)^2. \tag{19}$$

4 Simulations

The proposed algorithm has been validated and compared to the sh-rt and JUST algorithms by varying values of i) the Signal-to-Noise Ratio (SNR), ii) the matrix dimension N and iii) the number K of matrices to be diagonalized. Three kinds of simulation were conducted in order to quantify the relative effects of these quantities. The matrix set to be diagonalized is built according to the following model:

$$\forall k = 1 \dots K, \mathbf{M}^{(k)} = \frac{\mathbf{X}^{(k)}}{\|\mathbf{X}^{(k)}\|_F} + \sigma \frac{\mathbf{E}^{(k)}}{\|\mathbf{E}^{(k)}\|_F} \text{ with } \mathbf{X}^{(k)} = \mathbf{A}\mathbf{D}^{(k)}\mathbf{A}^{-1}. \tag{20}$$

\mathbf{A} , $\mathbf{D}^{(k)}$ and $\mathbf{E}^{(k)}$ entries are drawn randomly according to a standard normal distribution. $\mathbf{E}^{(k)}$ simulates a Gaussian additive noise. We define the SNR as $-20 \log_{10}(\sigma)$. Hence, σ is chosen in order to obtain the desired value of SNR.

At the end of each sweep the euclidean norm of the off-diagonal components of $\mathbf{M}^{(k,M)}$ is computed and compared to the value computed at the previous sweep. Algorithms are stopped when the relative deviation between two successive values is smaller than 10^{-7} . N_s is then the number of sweeps needed to reach the convergence. We define r_A as the relative squared error between the true eigenvector matrix and its estimate after having removed the scaling and permutation indeterminacies. Therefore, algorithm results are judged according to three criteria: r_D , N_s and r_A . Note that the three algorithms have comparable numerical complexities, thereby N_s is a pertinent criterion to judge the convergence speed.

Each situation is repeated 100 times with a new draw of the \mathbf{A} , $\mathbf{D}^{(k)}$ and $\mathbf{E}^{(k)}$ matrices at each time. We present here the median values of r_D and r_A along with the mean values of N_s obtained from each algorithm.

4.1 Influence of the SNR

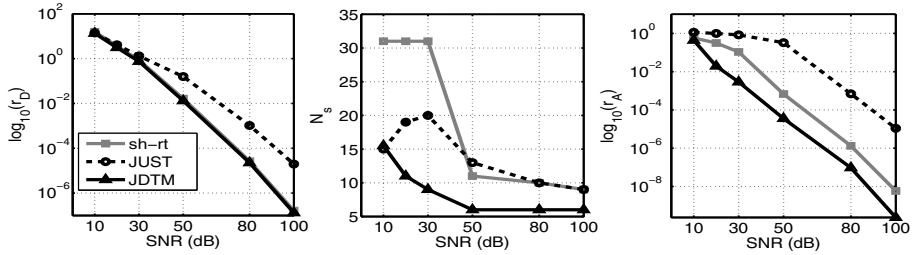
This first scenario varies the SNR from 10 dB to 100 dB whereas K and N are fixed to 50 and 10, respectively. Results are reported in figure 1(a). JD TM and sh-rt provide very close results in terms of diagonalization achievement whatever the SNR value, while the JUST algorithm is not as good in case of high SNR. At the exception of the 10 dB case (for which no algorithm converges) the JD TM algorithm requires at most half less sweeps than the other algorithms to reach the convergence. Furthermore, JD TM consistently provides a better estimation of the eigenvector matrix. Notably, at 20 and 30 dB one can note that at least 50% of the eigenvector matrices are well estimated by the JD TM algorithm, whereas this is not the case with the other algorithms.

4.2 Influence of the Matrix Size

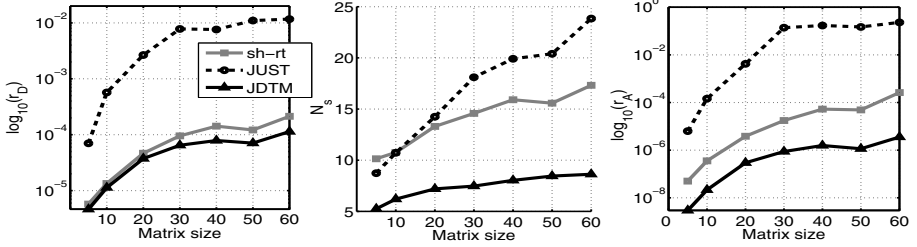
Now we vary the matrix dimension N from 5 to 60 whereas the size of the matrix set and the SNR are fixed to 50 and 80 dB respectively. Results are reported in figure 1(b). JD TM and sh-rt outclass the JUST algorithm. Both provides very satisfying results, even in the case of large matrices. Diagonalization achievement of both algorithms are quite similar but the distance between the two curves increases with the matrix size in favour of the JD TM algorithm. In addition, the latter needs between 5 to 8 sweeps to converge against 10 to 17 for sh-rt and this gap also increases with the matrix size. In the same way, the comparison of the r_A median values highlights the efficiency of the JD TM algorithm which clearly improves sh-rt results, whatever the considered matrix size.

4.3 Influence of the Size of the Matrix Set to Be Diagonalized

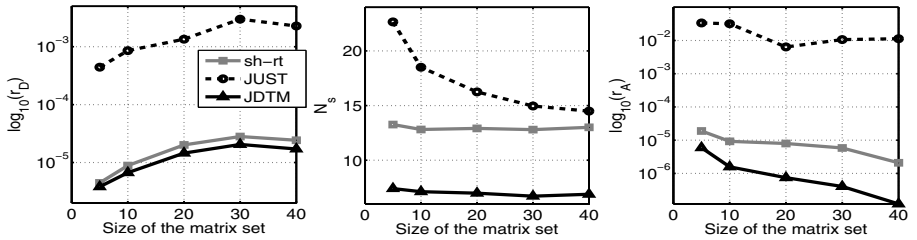
Finally we vary the number of matrices from 5 to 40 whereas the matrix size and the SNR are fixed to 20 and 80 dB, respectively. Results are reported in figure 1(c). These are similar to those observed previously. The JD TM algorithm



(a) according to the SNR



(b) according to the matrix size



(c) according to the size of the number of matrices to be diagonalized

Fig. 1. Evolution of the three comparison criteria: r_D (median values), N_s (mean values) and r_A (median values)

consistently surpasses the other algorithms. Notably, it only requires around 7 sweeps to converge against 13 for sh-rt, whatever the size of the matrix set. In addition, regarding the eigenvector matrix estimation, the gap observed between the two curves in favour of the JD TM algorithm gradually increases with the size of the matrix set.

5 Conclusion

In spite of its simplicity we observed that the proposed algorithm invariably outperforms the reference algorithms in all the considered situations and according to both classical criteria, which are the diagonalization achievement and the number of sweeps. In addition it has been shown that the JD TM algorithm

also provides a better estimation of the eigenvector matrix. From the presented results, it notably appears that the JD_{TM} algorithm is a fast algorithm with a good estimation precision of the eigenvector matrix, particularly in the most difficult cases. Finally this study also highlights the sensitivity of Jacobi-like algorithms to the choice of the optimization criterion.

References

1. van der Veen, A.J., Ober, P.B., Deprettere, E.F.: Azimuth and elevation computation in high resolution DOA estimation. *IEEE Trans. Signal Proc.* 40, 1828–1832 (1992)
2. Lemma, A.N., van der Veen, A.J., Deprettere, E.F.: Analysis of joint angle-frequency estimation using ESPRIT. *IEEE Trans. Signal Proc.* 51, 1264–1283 (2003)
3. Haardt, M., Nossek, J.A.: Simultaneous Schur decomposition of several nonsymmetric matrices to achieve automatic pairing in multidimensional harmonic retrieval problems. *IEEE Trans. Signal Proc.* 46, 161–169 (1998)
4. Albera, L., Ferréol, A., Chevalier, P., Comon, P.: ICAR, a tool for blind source separation using fourth order statistics only. *IEEE Trans. Signal Proc.* 53(10-1), 3633–3643 (2005)
5. Roemer, F., Haardt, M.: A closed-form solution for multilinear PARAFAC decompositions. In: *IEEE SAM 2008*, pp. 487–491 (2008)
6. Bunse-Gerstner, A., Byers, R., Mehrmann, V.: Numerical Methods for Simultaneous Diagonalization. *SIAM J. Matrix Anal. Applicat.* 14 (4), 927–949
7. Yeredor, A.: Non-Orthogonal Joint Diagonalization in the Least-Squares Sense with Application in Blind Source Separation. *IEEE Trans. Signal Proc.* 50(7), 1545–1553 (2002)
8. Karfoul, A., Albera, L., Birot, G.: Blind underdetermined mixture identification by joint canonical decomposition of HO cumulants. *IEEE Trans. Signal Proc.* 58(2), 638–649 (2010)
9. Strobach, P.: Bi-iteration multiple invariance subspace tracking and adaptive ESPRIT. *IEEE Trans. Signal Proc.* 48, 442–456 (2000)
10. Fu, T., Gao, X.: Simultaneous Diagonalization with Similarity Transformation for Non-defective Matrices. In: *IEEE ICASSP 2006*, pp. 1137–1140 (2006)
11. Goldstine, H.H., Horwitz, L.P.: A procedure for the diagonalization of normal matrices. *J. ACM* 6(2), 176–195 (1959)
12. Eberlein, P.J.: A Jacobi-like method for the automatic computation of eigenvalues and eigenvectors of an arbitrary matrix. *Journal of the Society for Industrial and Applied Mathematics* 10(1), 74–88 (1962)
13. Ruhe, A.: On the quadratic convergence of a generalization of the Jacobi method to arbitrary matrices. *BIT Numerical Mathematics* 8, 210–231 (1968)
14. Souloumiac, A.: Nonorthogonal joint Diagonalization by Combining Givens and Hyperbolic Rotations. *IEEE Trans. Signal Proc.* 57(6), 2222–2231 (2009)
15. Ifferroudjene, R., Abed Meraim, K., Belouchrani, A.: A New Jacobi-like Method for Joint Diagonalization of Arbitrary non-defective Matrices. *Applied Mathematics and Computation* 211, 363–373 (2009)
16. Cardoso, J.-F., Souloumiac, A.: Jacobi Angles for Simultaneous Diagonalization. *SIAM Journal on Matrix Analysis and Applications* 17(1), 161–164 (1996)