Comparative performance analysis of nonorthogonal joint diagonalization algorithms

Mesloub Ammar, Karim Abed-Meraim, Adel Belouchrani

► To cite this version:

Mesloub Ammar, Karim Abed-Meraim, Adel Belouchrani. Comparative performance analysis of nonorthogonal joint diagonalization algorithms. WOSSPA, May 2013, Zeralda, Algeria. hal-01002140

HAL Id: hal-01002140

<https://hal.science/hal-01002140>

Submitted on 13 Jan 2015

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.

COMPARATIVE PERFORMANCE ANALYSIS OF NON ORTHOGONAL JOINT DIAGONALIZATION ALGORITHMS

Mesloub Ammar*

Karim Abed-Meraim**

Adel Belouchrani***

*Ecole Militaire Polytechnique, BP 17 Bordj El Bahri, Algiers, Algeria

**Polytech' Orléans, Univ. Orléans 12 rue de Blois BP 6744, 45067 Orléans, France

***Ecole Nationale Polytechnique, 10 Avenue Hassen Badi, 16200 Algiers, Algeria
mesloub.a@gmail.com, karim.abed-meraim@univ-orleans.fr, adel.belouchrani@enp.edu.dz

ABSTRACT

Recently, many non orthogonal joint diagonalization (NOJD) algorithms have been developed and applied in several applications including blind source separation (BSS) problems. The aim of this paper is to provide an overview of major complex NOJD (CNOJD) algorithm and to study and compare their performance in adverse scenarios. This performance analysis reveals many interesting features that help the non expert user to select the CNOJD method depending on the application conditions.

1. INTRODUCTION

The problem of joint diagonalization (JD) can be encountered in many applications and mainly in BSS related problems [1] [2]. JD algorithms can be classified in orthogonal JD (OJD) methods which proceed in two steps: whitening and orthogonal diagonalization steps, and non orthogonal JD methods which avoid the whitening step and help improve the separation performance when applied in BSS [3]. In this paper, we provide a comparative performance analysis of major iterative CNOJD algorithms, namely: ACDC (Alternating Columns and Diagonal Centring) developed by Yeredor in 2002 [4], FAJD (Fast Approximative Joint diagonalization Algorithm) developed by Li and Zhang in 2007 [5], UWEDGE (UnWeighted Exhaustive Diagonalization using Gauss itErations) developed by Tichavsky and Yeredor in 2008 [6], JUST (Joint Unitary Shear Transformation) developed by Iferroudjene, Belouchrani and Abed-Meraim in 2009 [7], CVFFdiag (Complex Valued Fast Frobenius diagonalization) developed by Xu, Feng and Zheng in 2011 [8], LUCJD (LU decomposition for Complex Joint Diagonalization) developed by Wang, Gong and Lin in 2012 [9] and CJDi (Complex Joint Diagonalization) developed by Mesloub, Abed-Meraim and Belouchrani in 2012 [10].

Note that we restrict our comparative study only to iterative methods and we exclude the non iterative methods [11] [12] which are of higher complexity order especially for large dimensional systems. The aim of this paper is to provide a brief overview and compare¹ the previous algorithms in various scenarios including difficult cases (i) when the target matrices (to be jointly diagonalized) have a modulus of uniqueness (MOU defined in equation (7)) close to one [14], (ii) when the target (resp. mixing) matrices are ill conditioned, (iii) when the matrices' dimension is large and (iv) when the matrices are corrupted by additive noise.

¹Note that a comparative study has already been proposed in [13]. However, ours considers new algorithms as well as criteria of comparison not considered in [13].

2. PROBLEM FORMULATION

The problem of non orthogonal joint diagonalization can be formulated as follows: consider a set of K complex square matrices, $\mathbf{M}_k \in \mathbb{C}^{N \times N}$ $k = 1, \dots, K$ sharing the decomposition given in (1).

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

where $\mathbf{A}$ is a non-defective complex square matrix and $\mathbf{D}_k$ are diagonal complex matrices. Matrix $\mathbf{A}$ is known in BSS as a mixing matrix. $\mathbf{A}^H$ denotes the transpose conjugate of $\mathbf{A}$.

In practice, matrices $\mathbf{M}_k$ are given by some sample averaged statistics that are corrupted by estimation errors due to noise and finite sample size effects. Thus, they are approximate jointly diagonalizable matrices. These matrices can be rewritten as:

$$\mathbf{M}_k = \mathbf{A} \mathbf{D}_k \mathbf{A}^H + \mathbf{\Xi}_k, k = 1, \dots, K \quad (2)$$

where $\mathbf{\Xi}_k$ are perturbation (noise) matrices.

Given the set of complex matrices $\mathbf{M}_k$, the problem of JD is how to estimate the mixing matrix $\mathbf{A}$ and the diagonal matrices $\mathbf{D}_k$ in order to get the decomposition given in equation (1).

In BSS context, the JD problem consists of finding the demixing matrix $\mathbf{V}$ such that matrices $\mathbf{V} \mathbf{M}_k \mathbf{V}^H$ are diagonal or approximately diagonal if we consider the approximative joint diagonalization given in equation (2). Matrix $\mathbf{V}$ is called diagonalizing or demixing matrix.

3. JOINT DIAGONALIZATION CRITERIA

In this section, we present different criteria considered for joint diagonalization problem. In [4], the algorithm is based on minimizing the following criterion :

$$C_{DLS}(\mathbf{A}, \mathbf{D}_1, \mathbf{D}_2, \dots, \mathbf{D}_K) = \sum_{k=1}^K w_k \left\| \mathbf{M}_k - \mathbf{A} \mathbf{D}_k \mathbf{A}^H \right\|_F^2 \quad (3)$$

where $\|\cdot\|_F$ refers to the Frobenius norm, w_k are some positive weights and $(\mathbf{A}, \mathbf{D}_1, \mathbf{D}_2, \dots, \mathbf{D}_K)$ are the searched mixing matrix and diagonal matrices, respectively. This criterion is called in [11], the direct least-squares (DLS) criterion.

Another considered criterion is the indirect least squares criterion expressed as [11]:

$$C_{ILS}(\mathbf{V}) = \sum_{k=1}^K \left\| \mathbf{V} \mathbf{M}_k \mathbf{V}^H - \text{diag}(\mathbf{V} \mathbf{M}_k \mathbf{V}^H) \right\|_F^2 \quad (4)$$

This criterion is widely used in numerous algorithms [2] [10] [6] [14]. The problem with this criterion is that it can admit undesired solutions e.g. trivial solution $\mathbf{V} = \mathbf{0}$ and degenerate solutions where $\det(\mathbf{V}) = 0$. Consequently, the algorithms based on the minimization of (4) introduce a constraint to avoid these undesired solutions. In [2], the estimated mixing and diagonalizing matrices are unitary matrices so that undesired solutions are avoided. In [10], the diagonalizing matrix is estimated as a product of Givens and Shear rotations where undesired solutions are excluded. In [9], the diagonalizing matrix $\mathbf{V}$ is estimated in the form of LU (or LQ) factorization where $\mathbf{L}$ and $\mathbf{U}$ are lower and upper triangular matrices with ones at the diagonals and $\mathbf{Q}$ is a unitary matrix. These two factorizations LU and LQ impose a unit valued determinant for the diagonalizing matrix.

Another way to avoid undesired solutions is to modify the previous criterion, as in [5] by including the additional term $-\beta \text{Log}(|\det(\mathbf{V})|)$ so that the JD criterion becomes:

$$C'_{ILS}(\mathbf{V}) = \sum_{k=1}^K \left\| \mathbf{V} \mathbf{M}_k \mathbf{V}^H - \text{diag}(\mathbf{V} \mathbf{M}_k \mathbf{V}^H) \right\|_F^2 - \beta \text{Log}(|\det(\mathbf{V})|) \quad (5)$$

Another criterion has been introduced in [6] taking into account the two matrices $(\mathbf{A}, \mathbf{V})$ according to:

$$C_{LS}(\mathbf{V}, \mathbf{A}) = \sum_{k=1}^K \left\| \mathbf{V} \mathbf{M}_k \mathbf{V}^H - \mathbf{A} \text{diag}(\mathbf{V} \mathbf{M}_k \mathbf{V}^H) \mathbf{A}^H \right\|_F^2 \quad (6)$$

Note that, this criterion fuses the direct and indirect forms by relaxing the dependency between the mixing and the diagonalizing matrices. It is known as the least squares criterion. In equation (6), matrix $\mathbf{A}$ refers to the residual mixing matrix (not the original one).

Other criteria exist in the literature but are not considered here since they are applied only for positive definite or real valued matrices (for more details see [15] and [16]) or for matrices with special structures, e.g [17].

4. REVIEW OF MAJOR NOJD ALGORITHMS

We present here the principle of each of the major NOJD algorithms we considered in our comparative study.

ACDC [4] This algorithm proceeds by alternating two phases, the AC (Alternating Columns) phase and DC (Diagonal Centers) phase. The solution of this algorithm is obtained by minimizing the direct least-squares criterion C_{DLS} given in (3). At AC phase and for each iteration, only one column in the mixing matrix is estimated by minimizing the considered criterion while the other parameters are kept fixed. In the DC phase, the diagonal matrices are estimated while the mixing matrix is kept fixed. Note that, the DC phase is followed by several AC phases in order to guarantee the algorithm's convergence.

FAJD [5] This algorithm deals with the modified indirect least squares criterion given in (5). At each iteration, the algorithm computes one column of the demixing matrix while the others are kept fixed. This step is repeated until reaching the convergence state. Note that there is no update step for target matrices and the value assigned to β in [5] is one.

UWEDGE [6] This algorithm considers the criterion given in (6) and estimates in alternative way the mixing and demixing matrices. At first, the demixing matrix $\mathbf{V}$ is initialized as $\mathbf{M}_1^{-\frac{1}{2}}$. This value is introduced in the considered criterion to find the mixing matrix $\mathbf{A}$. The minimization w.r.t. $\mathbf{A}$ is done using numerical Gauss iterations. Once an estimate $\hat{\mathbf{A}}$ of the mixing matrix is obtained, the demixing matrix is updated as $\mathbf{V}^{(i)} = \hat{\mathbf{A}}^{-1} \mathbf{V}^{(i-1)}$ (i is the iteration index). The previous steps are repeated until convergence.

JUST [7] This algorithm is developed for target matrices sharing the algebraic joint diagonalization structure $\mathbf{M}'_k = \mathbf{A} \mathbf{D}'_k \mathbf{A}^{-1}$. Hence, the target matrices sharing the decomposition given in (1) are first transformed to another set of new target matrices sharing the algebraic joint diagonalization structure by inverting the first target matrix and left multiplying it by the rest of the target matrices. Once the new set of target matrices is obtained, JUST algorithm estimates the demixing matrix by successive Shear and Givens rotations minimizing C_{ILS} criterion³.

CV FFdiag [8] This algorithm is developed for real NOJD in [18] and generalized to the complex NOJD in [19] [8]. It uses C_{ILS} criterion and estimates the demixing matrix $\mathbf{V}$ in an iterative scheme in the form $\mathbf{V}^{(n)} = (\mathbf{I} + \mathbf{W}^{(n)}) \mathbf{V}^{(n-1)}$ where $\mathbf{W}^{(n)}$ is a matrix having null diagonal elements. The latter is estimated in each iteration by optimizing the first order Taylor expansion of C_{ILS} .

LUCJD [9] This algorithm considers C_{ILS} criterion as the previous one. It decomposes the mixing matrix in its LU form where $\mathbf{L}$ and $\mathbf{U}$ are lower and upper triangular matrices with diagonal entries equal to one. This algorithm is developed in [15] for real NOJD and generalized to complex case in [9]. Matrices $\mathbf{L}$ and $\mathbf{U}$ are optimized in alternating way by minimizing C_{ILS} . Note that the entries of $\mathbf{L}$ and $\mathbf{U}$ are updated one by one (keeping the other entries fixed).

CJDi [10] This algorithm can be considered as a generalization of JDi given in [20] for real NOJD problem. It considers a simplified version of C_{ILS} criterion (see [20] for more details) and estimates the demixing matrix by using successive Givens and Shear rotations.

5. PERFORMANCE COMPARISON STUDY

The aim of this section is to compare the considered NOJD methods in different scenarii. More precisely, we have chosen to evaluate and compare the algorithms' sensitiveness to different factors that may affect the JD quality. Next subsection describes the different criteria of comparison used in our study.

5.1. Criteria of comparison

In 'good JD conditions' all previous algorithms perform well with slight differences in terms of convergence rate or JD quality. However, in adverse JD conditions, many of these algorithms lose their

²This initial value of $\mathbf{V}$ is known as the whitening matrix in BSS context and assumes that $\mathbf{M}_1$ is positive definite. Otherwise, other initializations can be considered.

³The considered C_{ILS} in JUST can be expressed as in (4) by replacing $\mathbf{V}^H$ by $\mathbf{V}^{-1}$.

effectiveness or otherwise diverge. In this study, adverse conditions are defined according to the following criteria:

- *Modulus of uniqueness (MOU)*: defined in [14] [20] as the maximum correlation factor of vectors $\mathbf{d}_i = [D_1(i, i), \dots, D_K(i, i)]^T$, i.e.

$$\text{MOU} = \max_{i,j} \left(\frac{|\mathbf{d}_i^H \mathbf{d}_j|}{\|\mathbf{d}_i\| \|\mathbf{d}_j\|} \right) \quad (7)$$

It is shown in [20] that the JD quality decreases when the MOU get close to 1.

- *Mixing matrix condition number*: The JD quality depends on the good or ill conditioning of mixing matrix $\mathbf{A}$ (denoted $\text{cond}(\mathbf{A})$). The comparative study reveals the algorithm's robustness w.r.t. $\text{cond}(\mathbf{A})$.
- *Diagonal matrices condition number*: In BSS context the dynamic range of source signals affect the conditioning of diagonal matrices $\mathbf{D}_k$ and hence their separation quality. By comparing the algorithm's sensitiveness w.r.t. this criterion, we reveal their potential performance if used for separating sources with high dynamic ranges.
- *Matrices dimensions*: The JD problem is more difficult for large dimensional matrices ($N \gg 1$) and hence we compare the algorithms performance for $N = 5$ (small dimension) and $N = 50$ (large dimension).
- *Noise effect*: In practice, matrices $\mathbf{M}_k$ correspond to some sample averaged statistics and hence are affected by finite sample size and noise effects. In that case, the exact JD (EJD) becomes approximate JD (AJD) and the algorithms performance is lower bounded by the noise level as shown by our comparative study.

5.2. Simulation experiment set up

This subsection is about the simulation experiments set up. Firstly, we have chosen the classical performance index (PI) to measure the JD quality. It can be expressed as:

$$PI(\mathbf{G}) = \frac{1}{2N(N-1)} \sum_{n=1}^N \left(\sum_{m=1}^N \frac{|G(n,m)|^2}{\max_k |G(n,k)|^2} - 1 \right) + \frac{1}{2N(N-1)} \sum_{n=1}^N \left(\sum_{m=1}^N \frac{|G(m,n)|^2}{\max_k |G(k,n)|^2} - 1 \right) \quad (8)$$

where $G(n, m)$ is the $(n, m)^{th}$ entry of global matrix $\mathbf{G} = \hat{\mathbf{V}}\mathbf{A}$. The closer the PI is to zero, the better is the JD quality. This criterion is the same for all algorithms and allows us to compare them properly.

Secondly, we have organized our simulations scenarios as follows: simulations are divided in two experiment sets. The first one is dedicated to assess the algorithm's performance for exact joint diagonalization (EJD) case whilst the second experiment investigates the noise effect by considering AJD as mentioned in equation (2). Also, for each experiment, we have considered two cases namely the small dimensional case ($N = 5$) and the large dimensional case ($N = 50$). Finally, for each of these cases, we considered four simulation experiments: (i) a reference simulation of relatively good conditions where $\text{MOU} < 0.8$, $\text{Cond}(\mathbf{A}) < 50$ and $\text{Cond}(\mathbf{D}_k) < 100$; (ii) a simulation experiment with $\text{MOU} > 0.999$; (iii) a simulation experiment with $\text{Cond}(\mathbf{A}) > 100$ and (iv) a simulation experiment with $\text{Cond}(\mathbf{D}_k) > 10^4$.

Fig. 1. EJD results for 5×5 dimensional matrices.

Fig. 2. EJD results for 50×50 dimensional matrices.

Simulation parameters are summarized in the sequel. Mixing matrix entries are generated as independent and normally distributed variables of zero mean. Similarly, the diagonal entries of $\mathbf{D}_k$ are independent and normally distributed variables centred at $(1 + j)$. Target matrices are computed as in (1) for EJD case. The number of matrices is set to $K = 5$. For AJD case, these matrices are corrupted by additive noise as given in (2). The perturbation level can be expressed as:

$$\text{PL(dB)} = 10 \log_{10} \left(\frac{\|\mathbf{A}\mathbf{D}_k\mathbf{A}^H\|_F^2}{\|\Xi_k\|_F^2} \right) \quad (9)$$

In order to simplify the perturbation level manipulation, we have written matrix perturbation Ξ_k as follow:

$$\Xi_k = \beta_k \mathbf{N}_k \quad (10)$$

where $\mathbf{N}_k$ is a random perturbation matrix (regenerated at each Monte Carlo run) and β_k is a positive number allowing us to tune the perturbation level.

The simulation results (i.e performance index) are averaged over 200 Monte Carlo runs for small dimensional matrices and 20 Monte Carlo runs for large dimensional matrices.

Note that, all observations made out of these experiments are summarized in Table 1. Below, we provide only few comments to better explain certain results given in the plots.

5.3. Exact joint diagonalization

The simulation results for the EJD case are shown in figure 1 (for small matrices dimension) and figure 2 (for large matrices dimension). The top left subplots represent the reference cases of 'good JD conditions' as explained earlier.

We observe that even in the 'reference' case, the ACDC is quite sensitive to the system parameter realizations. In many of the Monte Carlo runs, the algorithm converges to local minima which explains the high value of its averaged PI .

We can also notice that certain curves are 'not smooth' due to the fact that the global convergence of their corresponding algorithms is not guaranteed (i.e for some of Monte Carlo runs, they converge to local minima) which affects significantly the averaged PI value.

5.4. Approximate joint diagonalization

The simulation results for the AJD case are given in figure 3 (for small matrices dimension) and figure 4 (for large matrices dimension), these results are obtained by using a maximum number of 100 sweeps. Compared to the EJD case, we observe here a floor effect due to the additive noise. Also, we observed in our experiments that, when it converges, the ACDC achieves the best JD quality in noisy environment. Unfortunately, its averaged PI value is generally poor because of its convergence problem.

Table 1 summarizes, in the simplified way, all the observation made through our simulation experiments.

In Table 2, we compare the NOJD algorithms w.r.t. convergence rate and their computational costs (details are omitted here).

Fig. 3. AJD results for 5×5 dimensional matrices.

Fig. 4. AJD results for 50×50 dimensional matrices.

6. CONCLUSION AND FUTURE WORKS

An overview followed by a thorough comparative study of existing NOJD algorithms is proposed. The algorithms robustness and sensitiveness to adverse JD conditions has been studied based on simulation experiments for both EJD and AJD cases. This performance comparison study reveals many interesting features summarized in Tables 1 and 2.

Based on this study, one can conclude that the CJD_i algorithm has the best performance for a relatively moderate computational cost. On the other hand, ACDC algorithm is the most sensitive one, diverging in most considered realizations. However, when it converges, it allows to reach the best (lowest) performance index values in the AJD case. Future works include the extension of this comparative study by considering other methods (Qdiag [21], ALS [22], ...), other matrix parameters (particularly, larger matrices with $N \geq 100$), and applications to specific BSS problems.

7. REFERENCES

- [1] J. F. Cardoso, A. Souloumiac, "Jacobi Angles For Simultaneous Diagonalization," *SIAM Journal of Matrix Anal. and Appl.*, 1996.
- [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] J.-F. Cardoso, "On the performance of orthogonal source separation algorithms," *In Proc. EUSIPCO, Edinburgh*, pp. 776–779, Sep. 1994.
- [4] A. Yeredor, "Non-Orthogonal Joint Diagonalization in the Least-Squares Sense With Application in Blind Source Separation," *IEEE Trans. on Sig. Proc.*, VOL. 50, NO. 7, JULY 2002.
- [5] X.-L. Li and X. D. Zhang, "Nonorthogonal Joint Diagonalization Free of Degenerate Solutions," *IEEE Trans. on Signal Process.*, Vol. 55, Is. 5, pp. 1803–1814, May 2007.
- [6] P. Tichavsky and A. Yeredor, "Fast Approximate Joint Diagonalization Incorporating Weight Matrices," *IEEE Trans. on Signal Process.*, Vol. 57, Is. 3, pp. 878–891, 2009.

	MOU	Cond(A)	Cond(D _k)	N	Noise
ACDC	H	H	H	H	L
FAJD	M	L	L	L	M
UWEDGE	L	L	H	L	M
JUST	L	L	L	L	M
CVFFdiag	H	H	L	H	L
LUCJD	M	H	L	L	L
CJD _i	L	M	L	L	L

Table 1. Algorithms performance and sensitiveness to different factors (H, M and L mean High sensitiveness, Moderate sensitiveness and Low sensitiveness, respectively).

	Convergence rate	Computational cost
ACDC	L	H
FAJD	M	M
UWEDGE	H	H
JUST	H	H
CVFFdiag	H	L
LUCJD	M	M
CJD _i	H	M

Table 2. Algorithms performance w.r.t. convergence rate and computational cost.

- [7] R. Iferroudjene, K. Abed-Meraim, A. Belouchrani, "A New Jacobi-Like Method For Joint Diagonalization Of Arbitrary Non-Defective Matrices," *Applied Mathematics and Computation*, ELSEVIER, 2009.
- [8] X.-Feng Xu, D.-Z. Feng and W. X. Zheng, "A Fast Algorithm for Nonunitary Joint Diagonalization and Its Application to Blind Source Separation," *IEEE Trans. on Sig. Pro.*, VOL. 59, NO. 7, JULY 2011.
- [9] K. Wang, X.F. Gong and Q. Lin, "Complex Non-Orthogonal Joint Diagonalization Based on LU and LQ Decomposition," *10th Int. Conf. Latent Variable analysis and signal separation*, pp. 50–57, March 12-15, 2012.
- [10] A. Mesloub, K. Abed-Meraim and A. Belouchrani, "Stable and efficient algorithms for complex non orthogonal joint diagonalization," *IEEE Trans. on Signal Process.*, submitted, 2012.
- [11] G. Chabriel and J. Barrere, "A Direct Algorithm for Nonorthogonal Approximate Joint Diagonalization," *IEEE Trans. on Signal Process.*, Vol. 60, No. 1, pp. 39–47, Jan. 2012.
- [12] A. Yeredor, "On Using Exact Joint Diagonalization for Noniterative Approximate Joint Diagonalization," *IEEE Signal Processing Letters*, Vol 12, NO. 9, pp. 645–648, Sep. 2005.
- [13] S. Degerine and E. Kane, "A Comparative Study of Approximate Joint Diagonalization Algorithms for Blind Source Separation in Presence of Additive Noise," *IEEE Trans. on Sig. Process.*, Vol 55, NO. 6, pp. 3022–3031, June 2007.
- [14] B. Afsari, "What can make joint diagonalization difficult?," *Int. Conf. Acoustics, Speech, Signal Processing (ICASSP)*, pp. 1377–1380, Apr. 2007.
- [15] B. Afsari, Charleston and J. Rosca, Eds. et al., "Simple LU and QR based non-orthogonal matrix joint diagonalization," *in Proc. 6th Int. Conf. Independent Compon. Analysis and Blind Source Separ.*, pp. 1148–1171, Berlin, Germany, 2006.
- [16] D.-T. Pham, "Joint Approximate Diagonalization Of Positive Definite Matrices," *SIAM J. on Matrix Anal. and Appl.*, 2001.
- [17] E. M. Fadaili, N. Moreau, and E. Moreau, "Non-orthogonal joint diagonalization/zero diagonalization for source separation based on time-frequency distributions," *IEEE Trans. Signal Process.*, vol. 55, no. 5, p. 16731687, May 2007.
- [18] A. Ziehe, P. Laskov, G. Nolte, K.-R. Mller, "Fast Algorithm For Joint Diagonalization With Non-Orthogonal Transformations And Its Application To Blind Source Separation," *Journal Machine Learning Research* 5, 2004.
- [19] K. Abed-Meraim, Y. Hua, and A. Belouchrani, "A general framework for blind source separation using second order

statistics,” *Eighth IEEE Digital Signal Processing Workshop*, Aug. 1998.

- [20] A. Souloumiac, “Nonorthogonal Joint Diagonalization by Combining Givens and Hyperbolic Rotations,” *IEEE Trans. on Sig. Proc.*, 2009.
- [21] R. Vollgraf and K. Obermayer, “Quadratic Optimization for Simultaneous Matrix Diagonalization,” *IEEE Trans. on Sig. Process.*, Vol. 54, NO 9, pp. 3270–3278, Sep. 2006.
- [22] T. Trainini, “Décomposition conjointe de matrices complexes application à la séparation de source,” *PhD thesis, University of Sud Toulon Var*, 2012.