FAST JACOBI ALGORITHM FOR NON-ORTHOGONAL JOINT DIAGONALIZATION OF NON-SYMMETRIC THIRD-ORDER TENSORS

Victor MAURANDI (1,2) and Eric MOREAU (1,2)

(1) Aix Marseille Université, CNRS, ENSAM, LSIS, UMR 7296, 13397 Marseille, France.
(2) Université de Toulon, CNRS, LSIS, UMR 7296, 83957 La Garde, France.
Email: {maurandi, moreau}@univ-tln.fr

ABSTRACT
We consider the problem of non-orthogonal joint diagonalization of a set of non-symmetric real-valued third-order tensors. This appears in many signal processing problems and it is instrumental in source separation. We propose a new Jacobi-like algorithm based on an LU decomposition of the so-called diagonalizing matrices. The parameters estimation is done entirely analytically following a strategy based on a classical inverse criterion and a fully decoupled estimation. One important point is that the diagonalization is directly done on the set of third-order tensors and not on their unfolded version. Computer simulations illustrate the overall good performances of the proposed algorithm.

Index Terms— Blind Source Separation, Independent Component Analysis, Joint Diagonalization, Third-Order Tensors.

1. INTRODUCTION
Joint diagonalization of sets of matrices or tensors is an important issue in source separation and independent component analysis (ICA) [1] [2]. It takes its origin in the two important papers [3] and [4] where the orthogonal joint diagonalization of matrices and the orthogonal diagonalization of a fourth-order tensor are introduced respectively in link with ICA. A generalization to statistics of any order greater than two is done in [5] and the specific case of the orthogonal joint diagonalization of third-order tensors is considered in [6].

Nowadays, attention has focused on the non-orthogonal joint diagonalization of matrices, see e.g. [9]–[14]. Moreover these methods often allow a potential computational parallelism. Concerning tensors, Canonical Polyadic Decomposition (CPD) [15]–[17] is a well-known multi-way decomposition that can be interpreted as the diagonalization of a single tensor. The non-orthogonal joint diagonalization of tensors is an open problem as such. To our best knowledge, even if a link exists with CPD, it seems that no works has directly been devoted to.

In this paper, we propose a Jacobi-like algorithm for the non-orthogonal joint diagonalization of non-symmetric real-valued third-order tensors. To our best knowledge, this is one of the first algorithm for non-unitary simultaneous tensor diagonalization. It is based on an LU parameterization of the so-called diagonalizing matrices. Contrary to the diagonalization of a third-order tensor which is handled in the literature as a problem of joint diagonalization of matrices unfolding the target tensor, in this paper we directly work on the tensors in order to joint diagonalize them, thus avoiding potential pre-processing step. We present two approaches of the algorithm based on a fully decoupled parameters estimation that allows an entirely analytical resolution for each approach. Numerical simulations illustrate the algorithm performances for both approaches and provide a comparison with the CPD algorithm based on the Alternate Least Squares (ALS) developed in the Matlab toolbox [18].

2. PROBLEM FORMULATION
We consider a set of third-order cubic \( N \times N \times N \), \( N \in \mathbb{N}^{*} \setminus \{1\} \), real-valued tensors \( T(k) = (T_{i_1,i_2,i_3}(k)) \), \( k = 1, \ldots, K \), \( K \in \mathbb{N}^{*} \setminus \{1\} \) and \((i_1,i_2,i_3) \in \{1,\ldots,N\}^3\). The tensors \( T(k) \) are assumed to ideally follow the following component-wise decomposition:

\[
T_{i_1,i_2,i_3}(k) = \sum_{j_1,j_2,j_3=1}^{N} D_{j_1,j_2,j_3}(k) A_{1,i_1,j_1} A_{2,i_2,j_2} A_{3,i_3,j_3} \tag{1}
\]

where \( A_x = (A_{x,i}) \) for \( x \in \{1,2,3\} \) are three \( N \times N \) invertible matrices, called factor matrices, and \( D(k) = (D_{j_1,j_2,j_3}(k)) \) are \( K \) diagonal tensors of dimension \( N \times N \times
there is a link between the decomposition in (1) and the CPD

\[ J = \sum_{k=1}^{K} \| Z \text{diag} \{ R(k) \} \|^2 \]

In practice, these tensors are typically obtained by some sample statistics through a finite number of data. The goal is, from only data tensors \( T(k), k = 1, \ldots, K \), to identify the inverse matrices, denoted by \( B_x \), of matrices \( A_x \), for all \( x \in \{1, 2, 3\} \). \( B_x \) are the so-called diagonalizing matrices. In order to do this, we introduce the transformed tensors defined as

\[ R(k) = T(k) \times_1 B_1 \times_2 B_2 \times_3 B_3. \]

If each \( B_x \) is equal to the inverse of the respective \( A_x \) (up to scaling factors and permutations) then it is easily seen that \( R(k) \) are diagonal tensors. Thus the purpose is to find the matrices \( B_x \) such that \( R(k) \) all are diagonal tensors.

In practice, this joint diagonalization of third-order tensors is done only in an approximate way because the set \( T(k) \) is estimated from statistics or other analysis operators. Classically, the approximation is measured using a criterion. In this paper, we propose to focus on a so-called inverse criterion defined as

\[ J(B_x) = \sum_{k=1}^{K} \| Z \text{diag} \{ R(k) \} \|^2 \]

where \( \| \cdot \| \) is the Frobenius norm and \( Z \text{diag} \{ \cdot \} \) is the zero diagonal tensor (with all zeros on the diagonal) built from the tensor argument. Once the criterion is fixed, the problem becomes an optimization one. Here one has to search for matrices \( B_x \) minimizing \( J \).

**Remark:** Because \( D(k) \) are diagonal tensors for all \( k \), there is a link between the decomposition in (1) and the CPD of a fourth order tensors. Indeed, we have

\[ T_{i_1i_2i_3i_4}(k) = \sum_{j=1}^{N} D_{i_1j}(k) A_{1i_1j} A_{2i_2j} A_{3i_3j} A_{4i_4j} \]

that is readily written as

\[ T'_{i_1i_2i_3i_4}(k) = \sum_{j=1}^{N} A_{1i_1j} A_{2i_2j} A_{3i_3j} A_{4i_4j} \]

where \( A_{1i_1j} = D_{i_1j}(k) \) and \( T'_{i_1i_2i_3i_4}(k) = T_{i_1i_2i_3i_4}(k) \) for all index values. This last writing is directly the CPD of the fourth-order tensor \( T' = (T'_{i_1i_2i_3i_4}) \).

3. PROPOSED ALGORITHM

3.1. Parameterization

An obvious solution minimizing the criterion \( J \) in (4) is \( B_x = 0 \) for all \( x \in \{1, 2, 3\} \) and where 0 is the zero matrix. Of course this solution is a degenerate one. Moreover we assumed that all diagonalizing matrices are invertible. That is why, it is important to consider a specific constraint on each matrix \( B_x \) allowing to find invertible solutions. For this, we constrain each matrix \( B_x \) to have a unit determinant. A way to do that consists of using an LU parameterization of each matrix \( B_x \). It is well-known that all square matrices can be decomposed as \( \text{DPLU} \), where \( D \) is a diagonal matrix, \( \text{P} \) is a permutation matrix and \( L \) and \( U \) are, respectively lower and upper triangular matrices with diagonal components all equal to 1. Thus, let us notice that \( \text{det}(L) = 1 \) and \( \text{det}(U) = 1 \) in such a way that \( LU \) has also a unit determinant. Since we are looking for matrices \( B_x \) up to scaling factors and permutations then \( D \) and \( P \) are useless. Then it remains to estimate the diagonalizing matrices all as \( B_x = L_x U_x \) for all \( x \in \{1, 2, 3\} \). Following [9] and [14], we here propose to estimate matrices \( L_x \) and \( U_x \) in an alternate way, i.e. by deriving \( U_x \) while keeping fixed \( L_x \) and vice versa.

3.2. Jacobi procedure

The Jacobi procedure consists of decomposing each \( U_x \) and \( L_x \) matrices by a product of \( \frac{N(N-1)}{2} \) elementary matrices as

\[ U_x = \prod_{i=1}^{N-1} \prod_{j=i+1}^{N} U_x^{ij} \quad \text{and} \quad L_x = \prod_{i=1}^{N-1} \prod_{j=i+1}^{N} L_x^{ij}. \]

The elementary matrices \( U_x^{ij} \) and \( L_x^{ij} \) each correspond to an \( N \times N \) identity matrix but by replacing the \((i, j)\), \( i < j \) component by \( u_x^{ij} \) for \( U_x^{ij} \) and by replacing the \((j, i)\), \( i < j \) component by \( l_x^{ij} \) for \( L_x^{ij} \).

We can notice that each elementary matrices \( U_x^{ij} \) (respectively \( L_x^{ij} \)) depends only on one parameter \( u_x^{ij} \) (respectively \( l_x^{ij} \)). Finally, for all \( x \in \{1, 2, 3\} \), \( B_x \) is assumed to follow the decomposition

\[ B_x = \prod_{i=1}^{N-1} \prod_{j=i+1}^{N} U_x^{ij} \times \prod_{i=1}^{N-1} \prod_{j=i+1}^{N} L_x^{ij}. \]

3.3. Proposed algorithm

One of our main goal is to find a direct analytical minimizing solution for each parameter. For that, we consider the estimation of \( u_x^{ij} \) and we directly use the criterion \( J \) in (4) on \( U_x^{ij} \).

This reads

\[ J(U_x^{ij}) = \sum_{k=1}^{K} \| Z \text{diag} \{ R(k) \} \|^2 \]

where

\[ R(k) = R(k) \times_1 U_1^{ij} \times_2 U_2^{ij} \times_3 U_3^{ij}. \]
on the slice \( \{(p, i, q) \mid (p, q) \in \{1, \ldots, N\}^2\} \) and \( U_q^j \) on \( \{(p, q, i) \mid (p, q) \in \{1, \ldots, N\}^2\} \). Thus, in (10), the components of \( R(k) \) located at the intersection of at least two of these slices are transformed by at least two matrices \( U_q^j \). Thus, at these locations, the components of \( R^{ij}(k) \) are non-linear equations in at least two \( u_{ij}^j \). If we directly minimize (9) for estimating all \( U_q^j \) matrices, as in [9] for the matrix case, these non-linear terms do not allow an analytical solution.

Nevertheless, we can overcome this issue by estimating each \( U_q^j \) in a decoupled way, i.e. estimating \( U_q^j \) for fixed \( U_q^{i'} \) and \( U_q^{i'} \), then updating \( R(k) \) and \( B \), and then doing the same for the two other matrices recursively. This proposed algorithm is called T-ALUJA for Tensorial Alternate LU Jacobi Algorithm.

4. PARAMETERS DERIVATION

For a given \((i, j)\) with \(i < j\), we now derive the parameters minimizing \( \mathcal{J} \) in (9). We only consider \( U_q^j \) since for \( L_q^j \), all follows the same lines.

In this section, we propose two different approaches. The first one is a classical one by using directly the criterion (9) for the parameters estimation. It just consists of taking into account all the off-diagonal components impacted by the transformation (10). The second approach is similar but adapted to the case where we are close to a diagonalizing solution. In the following, we only develop the derivations for \( U_q^j \). For the other \( U_q^2 \) and \( U_q^3 \), all may be simply deduced by indexes permutations.

4.1. Classical approach

Let us derive the optimal solution for \( U_q^j \) setting \( U_q^2 \) and \( U_q^3 \) to the identity. Hence (10) becomes

\[
R_q^{ij}(k) = R(k) \times U_q^j. \tag{11}
\]

Let the sets \( N = \{1, \ldots, N\} \) and \( P = N \setminus \{i\} \), the components of \( R_q^{ij}(k) \) can be read

\[
R_{ij,mn}^q(k) = R_{mn}(k) + R_{jmn}(k)u_{ij}^j
\]

\[
R_{i,jmn}^q(k) = R_{pmn}(k)
\]

with \((m, n) \in N^2 \) and \( p \in P \).

Let us define the sets \( Q = \{(i, b, c) \mid (b, c) \in N^2 \setminus \{i, i, i\}\} \) and \( S = \{(a, b, c) \mid a \in P, (b, c) \in N^2 \setminus \{(p, p, p) \mid p \in P\}\} \), we can also rewrite \( \mathcal{J} \) in (9) as

\[
\mathcal{J}(u_{ij}^j) = \sum_{k=1}^{K} \{ \sum_{(i,m,n) \in Q} R_{ij,mn}^2(k) + \sum_{(a,b,c) \in S} R_{ij,abc}^2(k) \}. \tag{13}
\]

Now using (12) in (13) leads to

\[
\mathcal{J}(u_{ij}^j) = \sum_{k=1}^{K} \{ \sum_{(i,m,n) \in Q} (R_{ij,mn}(k) + R_{jmn}(k)u_{ij}^j)^2 + \sum_{(a,b,c) \in S} R_{ij,abc}^2(k) \}. \tag{14}
\]

We now have to solve \( \partial \mathcal{J}/\partial u_{ij}^j = 0 \). After straightforward derivations, we obtain the following analytical optimal solution

\[
u_{ij}^j = -\frac{\sum_{k=1}^{K} \sum_{(i,m,n) \in Q} R_{ij,mn}(k)R_{jmn}(k)}{\sum_{k=1}^{K} \sum_{(i,m,n) \in Q} R_{ij,mn}^2(k)}. \tag{15}\]

4.2. Adapted approach

A close look at the above derivations shows that some transformed tensor components are more weighted than the others. Suppose that we are close to a diagonalizing solution in the sense that for all \(k \in \{1, \ldots, K\}\), all \(R(k)\) off-diagonal components have a very small magnitude in comparison to the \(R(k)\) diagonal component with the smallest one, i.e. for all \((a, b, c) \in \{N^2 \setminus \{(p, p, p) \mid p \in N\}\}, |R_abc(k)| \ll 1\). It is then rather easy to show that \(|u_{ij}^j| \ll 1\) for all \(x \in \{1, 2, 3\}\).

Thus in (12), in the first equation, for \((m, n) \in \{N^2 \setminus \{(j, j)\}\}, |R_{jmn}(k)u_{ij}^j| \ll |R_{mn}(k)|\), so this equation can be approximated by

\[
R_{i,mn}^j(k) \approx R_{mn}(k). \tag{16}\]

Hence, in \( R_{i,jmn}^j(k) \) after the transformation (11), it approximately remains only one non-constant off-diagonal component that is

\[
R_{i,ijj}^j(k) = R_{ijj}(k) + R_{jjj}(k)u_{ij}^j. \tag{17}\]

Finally, using (16) and (17) in (13), after straightforward derivation, (15) can be approximated by

\[
u_{ij}^j \approx -\frac{\sum_{k=1}^{K} R_{ijj}(k)R_{jjj}(k)}{\sum_{k=1}^{K} R_{jjj}^2(k)} \tag{18}\]

Comparing with the classical approach, the complexity of each parameter estimation is reduced. Moreover the estimations of \(u_{ij}^1\), \(u_{ij}^2\) and \(u_{ij}^3\) are now independent. Indeed the \(R(k)\) update using this approximate solution have no influence on the estimation of \(u_{ij}^2\) and \(u_{ij}^3\). Hence the estimation of these three parameters can be done all together in a same loop. This is a clear supplementary advantage.
5. COMPUTER SIMULATIONS

We now illustrate the performances of the proposed T-ALUJA algorithm by considering the two versions. The one using the classical approach is simply denoted by T-ALUJA and the one using the adapted approach is denoted by T-ALUJA-A. We also compare them with CPD-ALS algorithm given in [18].

In order to evaluate the algorithm performances, we use the performance index proposed in [5] [19] [20]. It compares each matrix $S_x = B_x A_x = (S_{x,ij})$ to the product of a permutation matrix and a diagonal matrix. It is defined by

$$I(S_x) = \frac{1}{2N(N-1)} I'(S_x)$$

where

$$I'(S_x) = \sum_{i=1}^{N} \left( \sum_{j=1}^{N} \frac{|S_{x,ij}|^2}{S_{x,i}} - 1 \right) + \sum_{j=1}^{N} \left( \sum_{i=1}^{N} \frac{|S_{x,ij}|^2}{S_{x,j}} - 1 \right)$$

with $S_{x,i}^2 = \max_m |S_{x,im}|^2$ and $S_{x,j}^2 = \max_m |S_{x,mj}|^2$. This normalized index is zero if $S_x$ satisfies $B_x = DPA_x^{-1}$. For the CPD-ALS algorithm, we first identify the three estimated matrices $A_x$ corresponding to each searched matrices $A_x$ and we directly consider $S_x = A_x^{-1} A_x$. Finally we take the mean value of $I(S_x)$ for $x \in \{1, 2, 3\}$.

We consider 10 real-valued tensors of size $10 \times 10 \times 10$ defined as $T(k) + t N(k)$ where $N(k)$ are noise tensors and $t$ represents the noise level. All components of matrices $A_x$ and diagonal tensors $D(k)$ follow a uniform law on $[-1, 1]$ whereas the components of $N(k)$ follow a zero mean unit variance normal distribution. Each $B_x$ is initialized randomly by using a zero mean unit variance normal distribution.

The Fig. 2 displays the index value w.r.t. the sweeps of the two Jacobi-like algorithms and CPD-ALS for 10 independent draws for $t = 10^{-4}$. Once again, T-ALUJA-A exhibits a much better convergence speed than the two other algorithms. Moreover, T-ALUJA-A also reaches a better average level of performance after 500 sweeps with $-39.7 dB$ against only $-29.9 dB$ for CPD-ALS and $-20.8 dB$ for T-ALUJA. Notice that, once again, the proposed algorithms are more robust than CPD-ALS of which two draws diverge. However, for the converging draws, CPD-ALS seems to often achieve a better level of performance than T-ALUJA-A.

Finally, the Fig. 3 displays the mean level of performance after 500 sweeps for 500 independent draws w.r.t. the level of noise $t$. T-ALUJA-A always reaches a better mean level of performance than T-ALUJA. CPD-ALS exhibits a good behavior in a very noisy context ($t \geq 10^{-3}$). Nevertheless, CPD-ALS gets an almost constant mean level of performance for each value of $t$. This is certainly due to some diverging draws. Our T-ALUJA approach seems more robust.

6. CONCLUSION

We proposed one of the first Jacobi-like algorithm for the non-orthogonal joint diagonalization of non-symmetric real-
Fig. 3. Mean performance index obtained after 500 sweeps versus the level of noise (500 independent draws of sets of 10 tensors of size $10 \times 10 \times 10$ for each level of noise).

valued third-order tensors. It is based on an LU decomposition and a fully decoupled estimation of the diagonalizing matrices parameters. Two approaches of this algorithm have been tackled. Each one rely on the analytical optimizing solution using an inverse cost function. The first approach takes into account all terms involved by the criterion whereas the second one is based on a selection of involved terms by considering the proximity to a diagonalizing solution. The numerical simulation illustrates the good behavior of both algorithms. Nevertheless T-ALUJA-A provides a much better convergence speed and accuracy than T-ALUJA in the simulation framework. Moreover T-ALUJA-A compares favorably to a CPD-ALS algorithm in term of robustness and convergence speed.

REFERENCES


