A FAST ALGORITHM FOR JOINT EIGENVALUE DECOMPOSITION OF REAL MATRICES

Rémi André(1,2), Tual Trainin(1,2), Xavier Luciani(1,2), 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.

ABSTRACT
We introduce an original algorithm to perform the joint eigenvalue decomposition of a set of real matrices. The proposed algorithm is iterative but does not resort to any sweeping procedure such as classical Jacobi approaches. Instead we use a first order approximation of the inverse of the matrix of eigenvectors and at each iteration the whole matrix of eigenvectors is updated. This algorithm is called Joint eigenvalue Decomposition using Taylor Expansion and has been designed in order to decrease the overall numerical complexity of the procedure (which is a trade off between the number of iterations and the cost of each iteration) while keeping the same level of performances. Numerical comparisons with reference algorithms show that this goal is achieved.

Index Terms—Joint eigenvalue decomposition, joint diagonalization, canonical polyadic decomposition, ICA.

1. INTRODUCTION
In this paper, we consider the Joint Eigenvalue Decomposition (JEVD), also called joint diagonalization by similarity. The principle of JEVD is to diagonalize a set of \( K \) non-defective matrices, as the following way:
\[
\forall k = 1, ..., K \quad M^{(k)} = AD^{(k)}A^{-1},
\]
where the matrix of eigenvectors \( A \) and the \( K \) diagonal matrices \( D^{(k)} \) are unknown. We only consider here the real case for simplicity. Note that the matrix of eigenvectors \( A \) is estimated up to a permutation and scaling indeterminacy. This limitation is inherent to the JEVD problem. The uniqueness condition of the JEVD problem is given in [1]. Let the \( N \times K \) matrix
\[
\Omega = \begin{pmatrix}
D^{(1)}_{11} & \cdots & D^{(K)}_{11} \\
\vdots & \ddots & \vdots \\
D^{(1)}_{N} & \cdots & D^{(K)}_{N}
\end{pmatrix}
\]
be the matrix whose columns are the diagonals of each matrix \( D^{(k)} \). JEVD is unique if and only if the rows of \( \Omega \) are all distinct. The JEVD is useful in several contexts: direction of arrival estimation [2], joint angle-delay estimation [3], multi-dimensional harmonic retrieval [4], Canonical Polyadic Decomposition (CPD) of tensors [1], [5], [6], and Independent Component Analysis (ICA) [7], [8], [9], [10], [11]. Note that this problem is different than classical Joint Diagonalization by Congruence (JDC), for which the right matrix \( A^{-1} \) is replaced by \( A^T \) [12], [13], [14]. Of course, if \( A \) is orthogonal JEVD and JDC problems are equivalent. JEVD algorithms usually resort of a Jacobi-like procedure using different matrix decompositions such as QR, LU and polar factorizations. Most algorithms are based on the polar decomposition: "sh-rt" algorithm [15] allows to diagonalize the \( K \) matrices correctly but it is not as good regarding the estimation of the matrix of eigenvectors, JUST algorithm [16] can sometimes improves these results. Before going further with this short JEVD historic, we would like to insist here that an important application of JEVD is the CPD of multiway arrays (also called PARAFAC). Indeed the CPD can be rewritten as a JEVD problem and this approach has given birth to very fast CPD algorithms such as CFS [5] and DIAG [6]. Thereby, in order to accentuate this strong point, it was important to develop fast JEVD algorithms. In this context, JDTM algorithm [6], still based on polar decomposition, provides satisfying results in most situations and requires only few iterations to converge. However it keeps a high computational cost per iteration. Conversely, JET-O algorithm [17], [18] based on the LU decomposition has low computational cost per iteration but it can require many more iterations to converge. Therefore, we propose here a new approach able to decrease the overall numerical complexity of the process by keeping low both the cost per iteration and the average number of iterations to reach the convergence and the same performance level that JET-O and JDTM. This algorithm is called Joint eigenvalues Decomposition based on a Taylor Expansion (JDTE) and has been inspired by the algorithm H-NOODLES for JDC [19].

2. NOTATIONS
In this paper scalars are denoted by a lower case \( a \), vectors by a lower case boldface \( \mathbf{a} \) and matrices by an upper case boldface \( \mathbf{A} \). The \( i-th \) element of the vector \( \mathbf{a} \) is denoted by \( a_i \) and the \( (i, j) -th \) element of a matrix \( \mathbf{A} \) is denoted by \( A_{ij} \). The identity matrix is denoted by \( \mathbf{I} \). The operator \( \text{Diag}(\cdot) \) represents the diagonal matrix built from the diagonal of the
matrix argument, the operator $\text{ZDiag}\{\cdot\}$ sets to zero the diagonal of the matrix argument and $\|\cdot\|$ is Frobenius norm of the argument matrix.

3. A FAST METHOD FOR JEVD BASED ON TAYLOR EXPANSION

Here, we consider that all matrices are of size $N \times N$. To solve the problem we have to find an estimated matrix $\hat{A}$ such that, for all matrices $M^{(k)}$, $A^{-1} \hat{M}^{(k)} \hat{A}$ are as diagonal as possible. The matrix $\hat{A}$ is estimated iteratively by successive updates. At each iteration $i$ a matrix $B_i$ is computed in order to decrease a diagonalization criterion and the matrix set is updated as

$$\forall i = 1, \ldots, S, \forall k = 1, \ldots, K \quad T_i^{(k)} = B_i^{-1} T_i^{(k)} B_i, \quad (3)$$

with $T_i^{(k)} = M_i^{(k)}$, and $S$ the number of iterations to reach the convergence. Thereby, if $S$ iterations allow to achieve the JEVD, $\hat{A}$ will be equal to $\prod_{i=1}^{S} B_i$ and all the $T_i^{(k)}$ matrices are diagonal.

To process, a classical approach is to reduce at each iteration the following criterion based on a quadratic measure of diagonality:

$$C(B_i) = \sum_{k=1}^{K} \|\text{ZDiag}\{B_i^{-1} T_i^{(k)} B_i\}\|^2. \quad (4)$$

As said before, at each iteration we have to determine the $N^2$ elements of the matrix $B_i$. Here, we simply decompose $B_i$ as follow:

$$B_i = (I + Z_i), \quad (5)$$

where $Z_i$ is equal to $\text{ZDiag}\{B_i\}$. Hence, at each iteration the updated data matrices become:

$$\forall k = 1, \ldots, K \quad T_i^{(k)} = (I + Z_i)^{-1} T_i^{(k)} (I + Z_i), \quad (6)$$

so that the criterion (4) depends only on $Z_i$ and can be written as:

$$C(B_i) = \tilde{C}(Z_i) = \sum_{k=1}^{K} \|\text{ZDiag}\{(I + Z_i)^{-1} T_i^{(k)} (I + Z_i)\}\|^2. \quad (7)$$

In the following, to simplify the notations, we drop the index $i$.

To solve this problem we aim to minimize the previous criterion, more precisely an approximation of this criterion. Indeed, it is compulsory to approximate our criterion if we want to work on an analytic form of $Z$. For this purpose, we assume that we are close to the solution and so that $\|Z\| \ll 1$ then we compute its first order Taylor expansion which yields:

$$(I + Z)^{-1} T^{(k)} (I + Z) \approx (I - Z) T^{(k)} (I + Z) \approx T^{(k)} - Z T^{(k)} + T^{(k)} Z - Z T^{(k)} Z \approx T^{(k)} - Z T^{(k)} + T^{(k)} Z. \quad (8)$$

Moreover each matrix $T^{(k)}$ can be decomposed as

$$T^{(k)} = \Lambda^{(k)} + O^{(k)}, \quad (9)$$

where $\Lambda^{(k)} = \text{Diag}\{T^{(k)}\}$ and $O^{(k)} = \text{ZDiag}\{T^{(k)}\}$. Once again, considering that we are close to the solution, all the $T^{(k)}$ matrices are almost diagonal and $\|O^{(k)}\| \ll 1$. This second approximation gives:

$$T^{(k)} - Z T^{(k)} + T^{(k)} Z \approx \Lambda^{(k)} + O^{(k)} - Z \Lambda^{(k)} + \Lambda^{(k)} Z \quad (10)$$

and thus, we finally resort to the following approximated JEVD criterion:

$$C_a(Z) = \sum_{k=1}^{K} \|\text{ZDiag}\{O^{(k)} - Z \Lambda^{(k)} + \Lambda^{(k)} Z\}\|^2 \approx \tilde{C}(Z). \quad (11)$$

Now, developing (11) yields:

$$C_a(Z) = \sum_{k=1}^{K} \sum_{m,n=1}^{N} (O^{(k)}_{mn} + Z_{mn} \Lambda^{(k)}_{mm} - Z_{mn} \Lambda^{(k)}_{mm})^2 \approx \sum_{m,n=1}^{N} f(Z_{mn}), \quad (12)$$

where

$$f(Z_{mn}) = \sum_{k=1}^{K} (O^{(k)}_{mn} + \Lambda^{(k)}_{mm} - \Lambda^{(k)}_{mn} Z_{mn})^2. \quad (13)$$

Since for all $m, n, f(Z_{mn})$ is non-negative, minimizing $C_a(Z)$ is equivalent to minimize all the $f(Z_{mn})$ independently. In order to find the elements of the updating matrix $Z$, we look for the values of $Z_{mn}$ which annul the derivative:

$$\forall m, n \neq n, \quad \frac{\partial f(Z_{mn})}{\partial Z_{mn}} = 2 \sum_{k=1}^{K} (O^{(k)}_{mn} + \Lambda^{(k)}_{mm} - \Lambda^{(k)}_{mn}) (\Lambda^{(k)}_{mn} - \Lambda^{(k)}_{mm}). \quad (14)$$

Finally, we deduce the analytic expression of $Z_{mn}$:

$$\forall m, n \neq n, Z_{mn} = -\frac{\sum_{k=1}^{K} O^{(k)}_{mn} (\Lambda^{(k)}_{mm} - \Lambda^{(k)}_{mn})}{\sum_{k=1}^{K} (\Lambda^{(k)}_{mn} - \Lambda^{(k)}_{mm})^2}. \quad (15)$$
Algorithm 1: The JDTE algorithm

1. Define a stopping criterion crit and a maximal number of iterations $S_{\text{max}}$.
2. Initialize $A$.
3. $i = 1$
4. while crit is false and $i \leq S_{\text{max}}$ do
5.   for $m = 1$ to $N$ do
6.     for $n = 1$ to $N$ do
7.       if $m \neq n$ then
8.         Compute $Z_{mn}$ using (15);
9.       end if
10.   end for
11. end for
12. Compute $B = I + Z$;
13. Compute $B^{-1}$;
14. for $k = 1$ to $K$ do
15.   $T^{(k)} \leftarrow B^{-1}T^{(k)}B$;
16. end for
17. $\hat{A} \leftarrow \hat{A}B$;
18. $i = i + 1$;
19. update crit
20. end while

Algorithm 1 gives an algorithmic description of the method. Note that here, one iteration consists in only one update of matrices $\hat{A}$ and $T^{(k)}$. Several stopping criteria can be used, for our simulations, algorithm are stopped as soon as the following condition is fulfilled:

$$\frac{|C(B_i) - C(B_{i-1})|}{C(B_{i-1})} \leq \varepsilon.$$  \hspace{1cm} (16)

4. NUMERICAL COMPLEXITIES OF JEVDO ALGORITHMS

We define here the numerical complexity (denoted $\Gamma$) of an iterative algorithm as the minimal number of multiplications computed by this algorithm during each iteration. According to this definition the numerical complexity of the JDTE algorithm is given by:

$$\Gamma(\text{JDTE}) \simeq (N(N - 1)(2K + 1) + 2N^2 + 2KN^3), \hspace{1cm} (17)$$

this value has to be compared to the numerical complexities of reference JEVDO algorithms. Hence we have:

$$\Gamma(\text{JET-O}) \simeq KN(N - 1)(N + 7)/6 + N^2(N + 1)/2$$
$$+ (5K + 4KN + N/3)(N - 1)N/2, \hspace{1cm} (18)$$

$$\Gamma(\text{sh-rt}) \simeq (3K + 16KN + 14N)(N - 1)N/2, \hspace{1cm} (19)$$

$$\Gamma(\text{JDTM}) \simeq (6K + 16KN + 8N)(N - 1)N/2, \hspace{1cm} (20)$$

$$\Gamma(\text{JUST}) \simeq (32 - 6K + 22KN + 8N)(N - 1)N/2. \hspace{1cm} (21)$$

Costs per iteration are thus dominated by a term in $\alpha KN^3$, whatever the algorithm, and only the value of $\alpha$ depends on the considered algorithm. It then clearly appears that only JET-O can compete with JDTE in terms of cost per iteration. However the average number of required iterations to reach the stopping criterion varies from one algorithm to another. It’s why, we define the total numerical complexity as $\Gamma_{\text{tot}} = \sum_{k} \Gamma(k)$. Therefore total numerical complexities of the different algorithms have to be compared empirically by means of numerical simulations.

5. NUMERICAL SIMULATIONS

Matrix sets to be diagonalized are created as follow: for given values of $K$ and $N$ we randomly draw (according to a standard normal distribution) a set of $K$ diagonal matrices and a squared matrix $A$ of size $N$. Then we compute the $K$ matrices $M^{(k)}$ as:

$$\forall k = 1, \ldots, K \quad M^{(k)} = \frac{A}{\|AD^{(k)}A^{-1}\|} + \sigma \frac{E^{(k)}}{\|E^{(k)}\|}. \hspace{1cm} (22)$$

Where $E^{(k)}$ models a Gaussian random noise. Parameter $\sigma$ allows to regulate the noise power and obtain the desired Signal to Noise Ratio (SNR) value which is then defined as: $-20 \log_{10}(\sigma)$.

In order to validate the proposed approach, we have compared the performance of JDTE algorithm with those of JUST, JDTM and JET-O according two different scenarios. For each simulation i.e. each time we vary a simulation parameter, 100 Monte-Carlo runs are performed. For each MC run a new random matrix set is built and the three algorithms are applied to this new set. The same stopping criterion (16) is used for the four algorithms with $\varepsilon = 10^{-6}$. All the algorithms are initialized with the identity matrix. At the end of each run two indicators are computed for each algorithm along with the numerical complexities. Our first indicator quantifies the relative deviation between the eigenvalues estimated by a JEVDO algorithm and the actual eigenvalues of matrices $M^{(k)}$. In the same way, our second indicator quantifies the relative deviation between the estimated matrix of eigenvectors and the actual one. These indicators are denoted $r_D$ and $r_A$ respectively and are computed as follow, after removing scaling and permutation indeterminacy:

$$r_D = \frac{1}{K} \sum_{k=1}^{K} \frac{\|d^{(k)} - \hat{d}^{(k)}\|}{\|d^{(k)}\|}, \hspace{1cm} (23)$$

$$r_A = \frac{\|A - \hat{A}\|}{\|A\|}, \hspace{1cm} (24)$$

where $d^{(k)}$ and $\hat{d}^{(k)}$ are vectors containing actual and estimated eigenvalues of $M^{(k)}$ respectively. Finally we resort to three criteria in order to assess algorithm performances: the
median values of $r_D$ and $r_A$ and the average value of $\Gamma_{\text{tot}}$ over the 100 MC runs.

For our first scenario, we consider sets of 20 matrices of size 5 and we vary the SNR value from 10 to 100. Evolutions of the three comparison criteria as a function of the SNR are reported on figures 1(a), 1(b) and 1(c). It can be seen that the JDTE, JET-O and JDTM algorithms provide very similar performances in terms of estimation precision of the eigenvalues and of the matrix of eigenvectors while JUST is less efficient. As expected, the numerical complexity criterion allows to discriminate between these three algorithms. We observe that for the lowest SNR values JET-O is the less costly algorithm respectively followed by JDTE and JDTM. Note that if we consider that the median values of $r_A$ must be lower than $10^{-1}$ no algorithm gives satisfying results under 30 dB. Then, above 40 dB, JDTE becomes clearly the less costly solution and this tendency increases with the SNR value.

The second scenario is performed in the same way but this time, we vary the size of the matrix sets from 2 to 50 while the SNR and the matrix size are set to 50 dB respectively. Results are plotted on figures 2(a), 2(b) and 2(c). Once again the diagonalization criterion does not allow to decide between JDTE, JDTM and JET-O and although JET-O algorithm provides now the best estimation of the matrix of eigenvectors, performances remains very close. Conversely, looking at the $\Gamma_{\text{tot}}$ criterion, it appears clearly that if we are looking for rapidity JDTE is a recommended choice. Indeed it is consistently less costly than its competitors and it is noteworthy that the gap increases linearly with the size of the matrix set. This makes JDTE a suitable solution for the CP decomposition since in most CPD application the size of the matrix set can be very large. More precisely, the ratios $\Gamma_{\text{tot}}\{\text{JDTM}\}/\Gamma_{\text{tot}}\{\text{JDTE}\}$, $\Gamma_{\text{tot}}\{\text{JET-O}\}/\Gamma_{\text{tot}}\{\text{JDTE}\}$ and $\Gamma_{\text{tot}}\{\text{JUST}\}/\Gamma_{\text{tot}}\{\text{JDTE}\}$ remains stable around 1.7, 1.4 and 3.4 respectively. In conclusion, numerical simulations, we can show that the proposed method is the least costly while keeping the same performances as JDTM and JET-O.

6. CONCLUSION
We have introduced here a new algorithm to compute the JEVD of a set of real matrices. Contrary to classical ap-
proaches our algorithm does not resort to any sweeping procedure so that at each iteration all the element of the matrix of eigenvectors are directly computed. This approach is achieved by using a Taylor expansion of the diagonalization criterion and allows significantly to reduce the cost per iteration. Thus, we have shown that the numerical complexity of one JDTE iteration is much lower than the one of most other JEVD algorithms. Numerical simulations confirm that JDTE clearly appears as the quickest algorithm in many practical situations, notably when the signal to noise ratio is not too low. Moreover, it offers similar performances than reference JEVD algorithms in term of estimation precision of eigenvectors and eigenvalues. This makes JDTE a good candidate to perform the JEVD step included in recent fast CPD algorithms.

REFERENCES


