PARALLELIZABLE PARAFAC DECOMPOSITION OF 3-WAY TENSORS

Viet-Dung Nguyen[1] *Karim Abed-Meraim*[1] *Nguyen Linh-Trung*[2]

[1] PRISME Lab., University of Orléans, France, [2] Faculty of Elec. and Telecom., VNU Univ. of Eng. & Tech., Vietnam

ABSTRACT

This paper introduces a new PARAFAC algorithm for a class of third-order tensors. Particularly, the proposed algorithm is based on subspace estimation and solving a non-symmetrical joint diagonalization problem. To deal with large scale problem, a procedure for overcoming scale and permutation ambiguities is proposed in a parallel computing scheme leading to a significant cost reduction of our method. Performance comparison with some state-of-the-art algorithms produces promising results.

Index Terms— Tensor decomposition, PARAFAC, subspace estimation, non-symmetrical joint diagonalization, parallel and distributed computation.

1. INTRODUCTION

Tensor decomposition is a useful tool for a wide range of applications such as chemometrics [1], signal processing [2], computer vision [3], and so on. Although there exist many decomposition techniques, the Tucker and PARAFAC decomposition are widely used, and can be considered as generalizations of the singular value decomposition (SVD) for multi-way array (higher than two-ways). While PARAFAC was found useful for finding latent variables because of uniqueness property, Tucker decomposition generally is used for compression. Here, we focus on PARAFAC decomposition of a three-way array (third-order tensor). In literature, many algorithms are developed for fitting the PARAFAC model [4],[5]. One of the 'workhorse' algorithms for long time is the alternating least square (ALS). The drawback of ALS is that it may require many iterations before convergence and is not guaranteed to the global optimum [5]. Few works giving approximate 'closed-form' [1] solutions were proposed such as [7] and [6]. These algorithms transform PARAFAC model for tensor into joint diagonalization (JD) of several sets of matrices. Although having promising results, they are either not robust with noise (as shown later) or/and expensive in term of computation complexity (required to compute HOSVD (Higher-Order-Singular Value Decomposition) [6] or a bi-linear mapping [7] before solving JD).

In this paper, we propose a new closed-form algorithm to solve the PARAFAC problem. The key steps of our algorithm is based on solving a non-symmetrical joint diagonalization (NJD) problem. These steps comparing to [7],[6] are simpler and more robust to noise. Moreover, the performance of our algorithm is near identical with ALS's which is close to CRLB as show in [8]. We also develop a procedure which allows integrate our algorithm in parallel computing scheme, thereby can be suitable for large-scale problem.

Notations: From now on, we follow the notation in [5]. Calligraphic letters are used for tensors ($\mathcal{A}, \mathcal{B}, \ldots$). Matrices, vectors, and scalars are denoted by boldface capital, boldface lowercase, and lowercase respectively; for example $\mathbf{A}$, $\mathbf{a}$, and a. Element (i,j,k) of a tensor $\mathcal{A} \in \mathbb{R}^{I\times J\times K}$ is symbolized as a_{ijk}, element (i,j) of a matrix $\mathbf{A} \in \mathbb{R}^{I\times J}$ as a_{ij}, and i-th entry of a vector $\mathbf{a} \in \mathbb{R}^{I}$ as a_i. Moreover, $\mathbf{A} \otimes \mathbf{B}$ defines Kronecker product of $\mathbf{A}$ and $\mathbf{B}$; $\mathbf{A} \odot \mathbf{B}$ defines Khatri-Rao (column-wise Kronecker) product of $\mathbf{A}$ and $\mathbf{B}$; $\mathbf{a} \circ \mathbf{b}$ stands for outer product of $\mathbf{a}$ and $\mathbf{b}$.

2. PARAFAC DECOMPOSITION

2.1. PARAFAC model

PARAFAC decomposition can be formulated in tensor form as follows

$$\mathcal{X} = \sum_{r=1}^{R} \mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{c}_r \tag{1}$$

These vectors can be combined into the so-called loading matrices $\mathbf{A} = [\mathbf{a}_1 \ldots \mathbf{a}_R] \in \mathbb{R}^{I\times R}$, $\mathbf{B} = [\mathbf{b}_1 \ldots \mathbf{b}_R] \in \mathbb{R}^{J\times R}$, and $\mathbf{C} = [\mathbf{c}_1 \ldots \mathbf{c}_R] \in \mathbb{R}^{K\times R}$. Tensor can also be represented in slab form [9] (or frontal slices [5])

$$\mathbf{X}_k = \mathbf{A}\mathbf{D}_k\mathbf{B}^T, \qquad k = 1, \ldots, K \tag{2}$$

where $\mathbf{D}_k = diag(\mathbf{C}_{k:})$ (i.e. stacks k-th row of matrix $\mathbf{C}$ in a diagonal square matrix). The corresponding matrix form of (2) is given by

$$\mathbf{X} = (\mathbf{C} \odot \mathbf{A})\mathbf{B}^T \tag{3}$$

where $\mathbf{X} = [\mathbf{X}_1^T, \ldots, \mathbf{X}_K^T]^T$. Similar equations can be written for I- and J- modes.

PARAFAC decomposition is generically unique if the following condition is satisfied [10] [7]

$$2R(R-1) \le I(I-1)J(J-1), \qquad R \le K \tag{4}$$

[1] As commented in [6], the 'closed-form' in the literature is conflicting. Here, we consider JD and NJD to be closed-form.

In practice, (1) is only an approximate tensor, i.e.

$$\mathcal{X} = \sum_{r=1}^{R} \mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{c}_r + \mathcal{N} \tag{5}$$

where $\mathcal{N}$ is a noise tensor. Our objective here is, for a given data tensor $\mathcal{X}$, to achieve an approximate R-rank PARAFAC with reduced computation.

3. PARAFAC-NJD DECOMPOSITION

Let consider the matrix representation of a tensor $\mathcal{X}$ (i.e.(3))

$$\mathbf{X} = (\mathbf{C} \odot \mathbf{A})\mathbf{B}^T + \mathbf{N} \tag{6}$$

Our algorithm can be divided into three steps:
Step 1: Estimate the subspace which spans column space of matrix $\mathbf{C} \odot \mathbf{A}$ (i.e. $(\mathbf{C} \odot \mathbf{A})\mathbf{Q}$ whereas $\mathbf{Q}$ is an unknown (non-singular) matrix
Step 2: Estimate the matrices $\mathbf{A}$ and $\mathbf{C}$,and $\mathbf{Q}$ simultaneously using non-symmetrical joint diagonalization
Step 3: Estimate the loading matrix $\mathbf{B}$
Now we explain these steps in detail. To estimate the subspace of the matrix $\mathbf{A} \odot \mathbf{B}$, we use the approach based on computing the covariance matrix of $\mathbf{X}$

$$\mathbf{R_x} = \frac{1}{J}(\mathbf{X}\mathbf{X}^T) = (\mathbf{C} \odot \mathbf{A})\mathbf{R_b}(\mathbf{C} \odot \mathbf{A})^T + \mathbf{R_n} \tag{7}$$

where $\mathbf{R_n}$ is assumed either negligible or asymptotically (i.e. for $J \gg 1$) proportional to identity matrix. Then the principal subspace is extracted by partial eigenvalue decomposition (i.e. by extracting $\mathbf{U}$, the matrix of the R principal eigenvectors of $\mathbf{R_x}$). It is straightforward to demonstrate:

$$\mathbf{U} = (\mathbf{C} \odot \mathbf{A})\mathbf{Q} \tag{8}$$

where $\mathbf{Q}$ is a non-singular $R \times R$ unknown matrix. The subspace estimation here can be thought as compression or dimension reduction step. In the second step, we propose to use a non-symmetrical joint diagonalization to recover the loading matrices $\mathbf{A}$ and $\mathbf{C}$, and estimate the matrix $\mathbf{Q}$ simultaneously. Observe that (8) can be rewritten as

$$\mathbf{M}_k = \mathbf{A}\mathbf{D}_k\mathbf{Q}, \qquad k = 1, \ldots, K \tag{9}$$

where $\mathbf{D}_k = diag(\mathbf{C}_{k:})$. Thus, $\mathbf{A}$, $\{\mathbf{D}_k\}$, and $\mathbf{Q}$ can be found by minimizing the following function

$$f(\mathbf{A}, \mathbf{Q}, \{\mathbf{D}_k\}) = \sum_{k=1}^{K} \parallel \mathbf{M}_k - \mathbf{A}\mathbf{D}_k\mathbf{Q} \parallel^2 \tag{10}$$

There are many existing algorithms which tackle this problem, for example [11], [12], [13] or ALS [14]. At the final step, we can estimate $\mathbf{B}$ from the estimated subspace $\mathbf{U}$ and $\mathbf{Q}$ in the first and second steps respectively

$$\mathbf{B} = (\mathbf{U}^T\mathbf{X})^T\mathbf{Q}^T \tag{11}$$

Here, we exploit the fact that $\mathbf{U}$ is orthogonal. Thus, we do not need to calculate $\mathbf{C} \odot \mathbf{A}$ after the second step. This algorithm is referred to as PARAFAC-NJD (NJD stands for Non-symmetrical Joint Diagonalization).
Remarks:

- First, while our approach solves a non-symmetrical joint diagonalization directly, the method in [7] needs to build a bi-linear mapping which has high computational complexity in the case of the captured data with large dimension before solving a symmetrical joint diagonalization problem. More importantly, in a noisy environment, our algorithm is more robust to noise when comparing with [7] as indicated in the simulation section.

- Second, by exploiting structure of the estimated subspaces, our method is easy to incorporate into parallel and dynamic schemes as presented in next section.

- Third, the solution of (9) is unique only up to scaling and permutation indeterminacy. More specifically, any set of matrices of the form $\mathbf{A}\mathbf{P}\mathbf{\Lambda}$, $\{\mathbf{\Lambda}^{-1}\mathbf{P}^T\mathbf{D}_k\mathbf{P}\tilde{\mathbf{\Lambda}}^{-1}\}$, and $\tilde{\mathbf{\Lambda}}\mathbf{P}^T\mathbf{Q}$ is also a solution where $\mathbf{P}$ is a permutation matrix, and $\mathbf{\Lambda}$ and $\tilde{\mathbf{\Lambda}}$ are diagonal matrices.

3.1. Parallel PARAFAC-NJD Decomposition

In this section, we develop a parallel version for the proposed algorithm. In [15], GMNS method is proposed to estimate principal and minor subspaces in a parallel scheme. It allows to reduce overall numerical cost by a factor of L^2 (L is number of available DSPs) while still preserving the estimation accuracy. This algorithm can be applied to estimate the principal subspace $\mathbf{U}$. Moreover, we further reduce the complexity of solving NJD problem in our algorithm by proposing a new procedure which has the spirit close to the GMNS algorithm.
Our new procedure has two key steps besides applying PARAFAC-NJD procedure. The first one is to divide a big tensor into sub-tensors. We consider non-overlapping and overlapping cases. The second one is to solve permutation problem between the estimated loading matrices from each sub-tensor.

3.1.1. Parallel PARAFAC-NJD Decomposition with non-overlapping tensor partitioning

Assume that, we want to divide a big tensor into L non-overlapping sub-tensors along K-mode such that $d = K/L$ (for simplicity, d is assumed to be integer-valued). Thus, l-th subtensor can be represented in matrix and slab form respectively as follows,

$$\mathbf{X}^l = (\mathbf{C}_{d(l-1)+1:dl,:} \odot \mathbf{A})\mathbf{B}^T \tag{12}$$

and,

$$\mathbf{X}_k^l = \mathbf{A}\mathbf{D}_k^l\mathbf{B}^T, \qquad k = 1, \ldots, d \tag{13}$$

Following this partitioning procedure, the subtensors still keep the same structure like the original one but having a smaller dimension (i.e $I \times J \times d$ for each).
After partitioning tensor, we apply the PARAFAC-NJD algorithm for each sub-tensor. Specifically, the estimated subspace $\mathbf{U}^l$ of tensor $\mathcal{X}^l$ yields

$$\mathbf{U}^l = (\mathbf{C}^l \odot \mathbf{A}^l)\mathbf{Q}^l. \tag{14}$$

where $\mathbf{C}^l \in \mathbb{R}^{d\times R}$, $\mathbf{A}^l \in \mathbb{R}^{I\times R}$, and $\mathbf{Q}^l \in \mathbb{R}^{R\times R}$. Then we find $\mathbf{C}^l$, $\mathbf{A}^l$ and $\mathbf{Q}^l$ by minimizing the function

$$f(\mathbf{A}^l, \mathbf{Q}^l, \{\mathbf{D}_k^l\}) = \sum_{k=1}^{d} \parallel \mathbf{M}_k^l - \mathbf{A}^l\mathbf{D}_k^l\mathbf{Q}^l \parallel^2 \tag{15}$$

Similar to (11), the loading matrix $\mathbf{B}^l$ can be calculated as

$$\mathbf{B}^l = ((\mathbf{U}^l)^T\mathbf{X}^l)^T(\mathbf{Q}^l)^T \tag{16}$$

where $\mathbf{B}^l \in \mathbb{R}^{J\times R}$. Until now, we have a set of matrices $\{\mathbf{A}^l\}$, $\{\mathbf{B}^l\}$ and $\{\mathbf{C}^l\}$. Note that, $\{\mathbf{A}^l\}$ and $\{\mathbf{B}^l\}$ are estimates of $\mathbf{A}$ and $\mathbf{B}$ correspondingly up to scale and permutation. On the other hand, each $\mathbf{C}^l$ presents a sub-matrix of $\mathbf{C}$ (see (25) and (26)).
We now solve scaling and permutation problem. For loading matrix $\mathbf{A}$, we consider loading matrices of two successive subtensors, $\mathbf{A}^l = \mathbf{A}\mathbf{P}^l\mathbf{\Lambda}^l$ and $\mathbf{A}^{l+1} = \mathbf{A}\mathbf{P}^{l+1}\mathbf{\Lambda}^{l+1}$. Then, for each column of matrix $\mathbf{A}^l$, we search

$$i = \underset{j}{\operatorname{argmax}} \frac{|(\mathbf{A}_{:,k}^l)^T(\mathbf{A}_{:,j}^{l+1})|}{\|\mathbf{A}_{:,k}^l\|\|\mathbf{A}_{:,j}^{l+1}\|} \qquad k = 1, \ldots, R \tag{17}$$

$$\mathbf{P}^{l+1,l}(i,k) = 1 \tag{18}$$

$$\mathbf{\Lambda}^{l+1,l}(i,i) = \frac{(\mathbf{A}_{:,i}^{l+1})^T(\mathbf{A}_{:,k}^l)}{\|\mathbf{A}_{:,i}^{l+1}\|} \tag{19}$$

where initialization of $\mathbf{P} = \mathbf{0}$ which is an $R \times R$ zero matrix and initialization of $\mathbf{\Lambda} = \mathbf{I}$ which is an identity matrix. Thus, loading matrix $\mathbf{A}$ can be found by (in informal way)

$$\mathbf{A}^{l+1} = \mathbf{A}^{l+1}(\mathbf{\Lambda}^{l+1,l})(\mathbf{P}^{l+1,l}) \tag{20}$$

For loading matrix $\mathbf{B}$, we note that

$$\mathbf{B}^l = ((\mathbf{U}^l)^T\mathbf{X}^l)^T(\mathbf{Q}^l)^T \tag{21}$$

$$= \mathbf{B}\mathbf{P}^l(\tilde{\mathbf{\Lambda}}^l)^T \tag{22}$$

We thus can apply the same procedure in previous description to $\mathbf{B}^l$ and $\mathbf{B}^{l+1}$ to find out $\tilde{\mathbf{\Lambda}}^{l+1,l}$. Consequently,

$$\mathbf{B}^{l+1} = \mathbf{B}^{l+1}(\tilde{\mathbf{\Lambda}}^{l+1,l})(\mathbf{P}^{l+1,l}) \tag{23}$$

Finally, from results of $\mathbf{A}$ and $\mathbf{B}$, each row of loading matrix $\mathbf{C}^{l+1}$ is given by

$$\mathbf{D}_k^{l+1} = (\mathbf{\Lambda}^{l+1,l})^{-1}(\mathbf{P}^{l+1,l})^T\mathbf{D}_k^{l+1}(\mathbf{P}^{l+1,l})(\tilde{\mathbf{\Lambda}}^{l+1,l})^{-1} \tag{24}$$

We refer to this procedure as PARAFAC-NJD-NO.

3.1.2. Parallel PARAFAC-NJD Decomposition with overlapping tensor partitioning

Similar to non-overlapping case, we want to divide a big tensor into L overlapping sub-tensors along K-mode. Let q be the number of overlapping slides. Hence, each overlapping tensor has $d + q$ slabs. The matrix and slab representation of this procedure for two successive subtensor $\mathcal{X}^l$ and $\mathcal{X}^{l+1}$ are given by

$$\mathbf{X}^l = (\mathbf{C}_{d(l-1)+1:dl+q,:} \odot \mathbf{A})\mathbf{B}^T \tag{25}$$

$$\mathbf{X}^{l+1} = (\mathbf{C}_{dl+1:d(l+1)+q,:} \odot \mathbf{A})\mathbf{B}^T \tag{26}$$

and

$$\mathbf{X}_k^l = \mathbf{A}\mathbf{D}_k^l\mathbf{B}^T, \qquad k = 1, \ldots, d+q \tag{27}$$

$$\mathbf{X}_k^{l+1} = \mathbf{A}\mathbf{D}_k^{l+1}\mathbf{B}^T \tag{28}$$

respectively, where, $\mathbf{D}_k^l = diag(\mathbf{C}_{k:}^l)$ and $\mathbf{D}_k^{l+1} = diag(\mathbf{C}_{k:}^{l+1})$ and $\mathbf{C}^l = \mathbf{C}_{d(l-1)+1:dl+q,:}$ and $\mathbf{C}^{l+1} = \mathbf{C}_{dl+1:d(l+1)+q,:}$.
Each subtensor now has size of $I \times J \times (d+q)$. We then apply the proposed algorithm for each sub-tensor to get a set of matrices $\{\mathbf{A}^l\} \in \mathbb{R}^{I\times R}$, $\{\mathbf{B}^l\} \in \mathbb{R}^{J\times R}$ and $\{\mathbf{C}^l\} \in \mathbb{R}^{(d+q)\times R}$. The scale and permutation problem can be solved by applying the same procedure in non-overlapping case for $\mathbf{A}$ and $\mathbf{B}$.
For the loading matrix $\mathbf{C}$, we consider two successive subtensors which yield $\mathbf{C}^l$ and $\mathbf{C}^{l+1}$

$$\mathbf{C}^l = \begin{bmatrix} \mathbf{C}^{l,nonoverlap} \\ \mathbf{C}^{l,overlap} \end{bmatrix} \mathbf{P}^l\mathbf{\Lambda}^l = \begin{bmatrix} \mathbf{C}_{d(l-1)+1:dl,:} \\ \mathbf{C}_{dl+1:dl+q,:} \end{bmatrix} \mathbf{P}^l\mathbf{\Lambda}^l \tag{29}$$

$$\mathbf{C}^{l+1} = \begin{bmatrix} \mathbf{C}^{l+1,overlap} \\ \mathbf{C}^{l+1,nonoverlap} \end{bmatrix} \mathbf{P}^{l+1}\mathbf{\Lambda}^{l+1} = \begin{bmatrix} \mathbf{C}_{dl+1:d(l+1)} \\ \mathbf{C}_{d(l+1)+1:d(l+1)+q} \end{bmatrix} \mathbf{P}^{l+1}\mathbf{\Lambda}^{l+1} \tag{30}$$

We then apply the described procedure for overlapping part of $\mathbf{C}^l$ and $\mathbf{C}^{l+1}$. Thus, loading matrix $\mathbf{C}$ is given by

$$\mathbf{C}^{l+1} = \mathbf{C}^{l+1}(\mathbf{\Lambda}^{l+1,l})(\mathbf{P}^{l+1,l}) \tag{31}$$

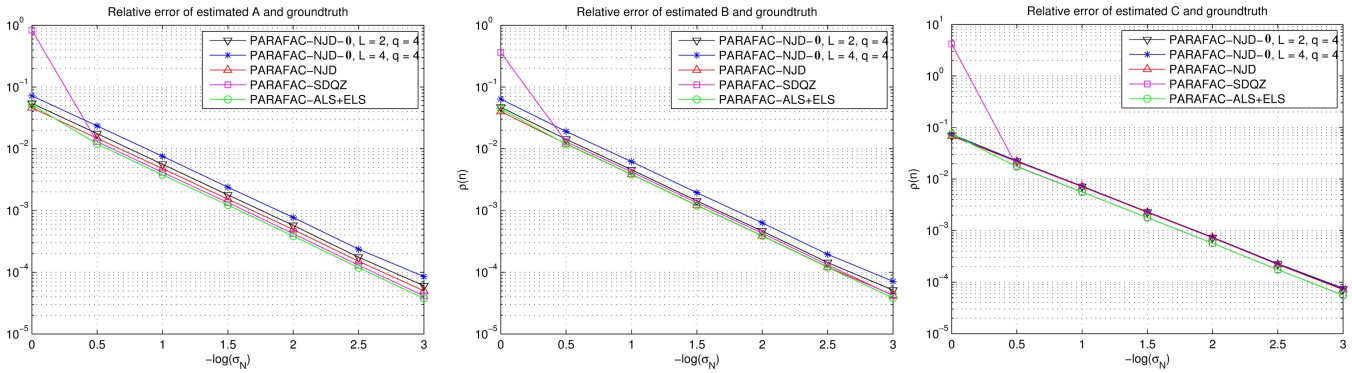
After matrices alignment, we can improve the estimates of $\mathbf{A}$ and $\mathbf{B}$ by averaging

$$\hat{\mathbf{A}} = \frac{1}{L}\sum_{1}^{L} \mathbf{A}^l \tag{32}$$

and

$$\hat{\mathbf{B}} = \frac{1}{L}\sum_{1}^{L} \mathbf{B}^l \tag{33}$$

We refer to this procedure as PARAFAC-NJD-O.



(a) Loading matrix $\mathbf{A}$ (b) Loading matrix $\mathbf{B}$ (c) Loading matrix $\mathbf{C}$

Fig. 1: Performance comparison of the proposed algorithms with ALS+ELS and PARAFAC-SDQZ (loading matrices)

4. SIMULATIONS

To assess the performance of the proposed algorithms, we use the same set-up as in [7]. The entries of a tensor $\mathcal{X}$ whose rank is R are drawn from a zero-mean unit-variance Gaussian distribution. Besides, $\mathcal{X}$ can be decomposed as in (3). A Gaussian noise term $\mathcal{N}$ is added to this tensor

$$\tilde{\mathcal{X}} = \frac{\mathcal{X}}{\| \mathcal{X} \|} + \sigma_N \frac{\mathcal{N}}{\| \mathcal{N} \|} \tag{34}$$

where the value of σ_N controls the noise level. The accuracy of the estimated loading matrices ($\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$) and matrix representation of tensor $\mathcal{X}$ ($\mathbf{X}$) is the relative error defined as

$$\rho(N) = \frac{1}{N} \sum_{n=1}^{N} \frac{\| \mathbf{H}_n - \mathbf{H}_{ex} \|}{\| \mathbf{H}_{ex} \|} \tag{35}$$

where N is the number of Monte Carlo runs and $\mathbf{H}_i$ is the estimate of the exact matrix $\mathbf{H}_{ex}$. In all experiments, N is set to 100. We compare the performance of our algorithms with two state-of-the-art ones, ALS+ELS from N-way toolbox[2] and PARAFAC-SDQZ [7]. Parameters for experiments are shown in Table1. Rank R is chosen to be small comparing to dimensions of tensor. The results from Figure 1 and 2 show that performance of PARAFAC-NJD is comparable to ALS+ELS and PARAFAC-SDQZ but with reduced computational cost. Moreover, it is more robust than PARAFAC-SDQZ in the noisy environment (i.e. $-log(\sigma_N) < 0.5$). Figures 1 and 2 also illustrate the effectiveness of proposed procedures in parallel computing scheme (likewise, for dynamic tensor analysis with the same parameters). The PARAFAC-NJD-0 ($L = 2, q = 4$) has slight performance loss comparing to the three batch algorithms. When number of processors L increases, PARAFAC-NJD-0 ($L = 4, q = 4$) gain more in term of complexity and have small performance loss. Thus changing L allows to balance between the accuracy and the complexity.

[2]N-way toolbox version 3.30 (including enhanced line search procedure [14]) is downloaded from http://www.models.life.ku.dk/nwaytoolbox

I	J	K	R	L	q
20	20	40	8	2;4	4

Table 1: Particular parameters are set in our experiments.

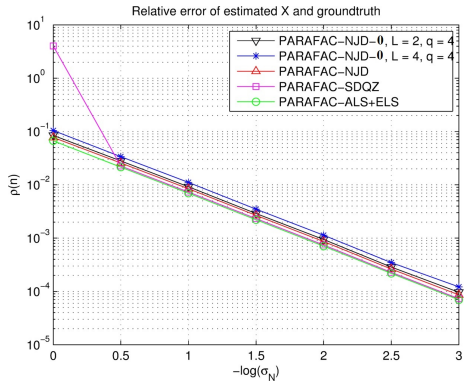


Fig. 2: Performance comparison of the proposed algorithms with ALS+ELS and PARAFAC-SDQZ (matrix representation of tensor $\mathcal{X}$).

5. CONCLUSION

We have described a new algorithm for tackling PARAFAC problem based on solving a non-symmetrical joint diagonalization problem. Its main advantages is its low complexity while preserving good estimation accuracy. Furthermore, a new procedure for solving scale and permutation ambiguities between the estimated loading matrices from different DSPs has been proposed to reduce complexity in a parallel computing scheme. In simulation, we have compared the proposed methods to ALS+ELS and PARAFAC-SDQZ and observed that the performance of our algorithm is similar to or even slightly better in specific contexts.

6. REFERENCES

[1] Rasmus Bro, "PARAFAC. tutorial and applications," *Chemometrics and intelligent laboratory systems*, vol. 38, no. 2, pp. 149–171, 1997.

[2] A Cichocki, C Mandic, AH Phan, C Caiafa, G Zhou, Q Zhao, L De Lathauwer, "Tensor decompositions for signal processing applications. from two-way to multiway component analysis," *IEEE Signal Processing Magazine*, p. accepted, 2014.

[3] M Alex O Vasilescu and Demetri Terzopoulos, "Multilinear analysis of image ensembles: Tensorfaces," in *Computer Vision, ECCV 2002*, pp. 447–460. Springer, 2002.

[4] Giorgio Tomasi and Rasmus Bro, "A comparison of algorithms for fitting the parafac model," *Computational Statistics & Data Analysis*, vol. 50, no. 7, pp. 1700–1734, 2006.

[5] Tamara G Kolda and Brett W Bader, "Tensor decompositions and applications," *SIAM review*, vol. 51, no. 3, pp. 455–500, 2009.

[6] Florian Roemer and Martin Haardt, "A closed-form solution for parallel factor (parafac) analysis," in *Acoustics, Speech and Signal Processing, 2008. ICASSP 2008. IEEE International Conference on*. IEEE, 2008, pp. 2365–2368.

[7] Lieven De Lathauwer, "A link between the canonical decomposition in multilinear algebra and simultaneous matrix diagonalization," *SIAM Journal on Matrix Analysis and Applications*, vol. 28, no. 3, pp. 642–666, 2006.

[8] Xiangqian Liu and Nikos D Sidiropoulos, "Cramer-Rao lower bounds for low-rank decomposition of multidimensional arrays," *Signal Processing, IEEE Transactions on*, vol. 49, no. 9, pp. 2074–2086, 2001.

[9] Nikos D Sidiropoulos, "Low-rank decomposition of multi-way arrays: A signal processing perspective," in *Sensor Array and Multichannel Signal Processing Workshop Proceedings, 2004*. IEEE, 2004, pp. 52–58.

[10] Joseph B Kruskal, "Three-way arrays: rank and uniqueness of trilinear decompositions, with application to arithmetic complexity and statistics," *Linear algebra and its applications*, vol. 18, no. 2, pp. 95–138, 1977.

[11] A-J Van Der Veen, "Joint diagonalization via subspace fitting techniques," in *Acoustics, Speech and Signal Processing, 2001. ICASSP 2001. IEEE International Conference on* vol. 5, pp. 2773–2776, 2001.

[12] Marco Congedo, Ronald Phlypo, and Dinh-Tuan Pham, "Approximate joint singular value decomposition of an asymmetric rectangular matrix set," *Signal Processing, IEEE Transactions on*, vol. 59, no. 1, pp. 415–424, 2011.

[13] K Abed-Meraim and Y Hua, "A least-squares approach to joint schur decomposition," in *Acoustics, Speech and Signal Processing, 1998. Proceedings of the 1998 IEEE International Conference on*. IEEE, 1998, vol. 4, pp. 2541–2544.

[14] Myriam Rajih, Pierre Comon, and Richard A Harshman, "Enhanced line search: A novel method to accelerate PARAFAC," *SIAM Journal on Matrix Analysis and Applications*, vol. 30, no. 3, pp. 1128–1147, 2008.

[15] Viet-Dung Nguyen, Karim Abed-Meraim, Nguyen Linh-Trung, and Rodolphe Weber, "Generalized MNS method for parallel minor and principal subspace analysis," in *Proceedings of European Signal Processing Conference (EUSIPCO)*. IEEE, 2014.