RANK-ONE TENSOR INJECTION : A NOVEL METHOD FOR CANONICAL POLYADIC TENSOR DECOMPOSITION

Anh-Huy Phan[1], *Petr Tichavský*[2] *and Andrzej Cichocki*[1]

[1]Brain Science Institute, RIKEN, Wakoshi, Japan
[2]Institute of Information Theory and Automation,
P.O.Box 18, 182 08 Prague 8, Czech Republic

ABSTRACT

Canonical polyadic decomposition of tensor is to approximate or express the tensor by sum of rank-1 tensors. When all or almost all components of factor matrices of the tensor are highly collinear, the decomposition becomes difficult. Algorithms, e.g., the alternating algorithms, require plenty of iterations, and may get stuck in false local minima. This paper proposes a novel method for such decompositions. The method injects one or a few rank-1 tensors into the data tensor in order to control the decompositions of the rank-expanded data, while still preserving the estimation accuracy of the original tensor. To achieve this, we develop a method to automatically generate the injected tensor which satisfies a specific estimation accuracy such that this tensor should not dominate rank-1 tensors of the data tensor, but is still able to be retrieved with a sufficient accuracy. Simulations on tensors with highly collinear factor matrices will illustrate efficiency of the proposed injecting method.

Index Terms— CANDECOMP/PARAFAC, tensor decomposition, tensor injection, tensor deflation, Cramér–Rao–induced bound

1. INTRODUCTION

CANDECOMP/PARAFAC (CP) approximates multiway data by sum of rank-1 tensors, and has found many applications, such as in chemometrics, telecommunication, data mining, neuroscience, separated representations and blind source separation [1–3]. There are plenty algorithms developed for CPD since this decomposition was first rediscovered independently by Harshman [4] and Carroll and Chang [5]. For example, the alternating algorithms update individual factor matrices sequentially, or all-at-once optimisation algorithms update all parameters simultaneously [6–8]. Despite of that, the alternating least squares (ALS) algorithm [4, 5, 9] is still the most widely used algorithm, because its update scheme is simple to implement and easy to extend to decompositions with additional constrains, e.g., non negativity [10], orthogonality [11, 12], sparsity and smoothness [1], or decomposition of incomplete data [15]. However, as other algorithms for CPD, the ALS algorithm often fails for data with factors of different magnitudes [6], or collinear loading components such as bottlenecks and swamps [13, 14].

In order to improve performance and speed-up convergence of ALS, numerous variants of this algorithm have been proposed. For example, the ALS algorithm incorporates the line search extrapolation methods [4, 15] in which step size is linearly decreased from a suitably chosen scalar constant. The algorithm works more efficiently with the exact or enhanced line search [16, 17] which finds the optimal step as root of a polynomial of degree $(2N - 1)$ for order-N tensors, yielding the smallest approximation error. Unfortunately, this line search is expensive for high order tensors and mostly suited for order-3 tensors. Alternatively, one can switch between the simple ALS update and the second-order line search [18], or between the ALS update and the rotation method [19].

ALS and its variants are useful for some difficult tensor decompositions. However, the main problem for ALS still remains. More specifically, so far, there is no method to control performance of the algorithms. The estimation process may stop before a desired result is attained, or it is still kept running even when good estimates to the solution have been achieved.

In this paper, we propose a novel method to control the tensor decomposition. The basic principle is based on the tensor deflation [20], which estimates a rank-1 tensor from a rank-R tensor without estimating all the other components. In Fig. 1, we illustrate the rank-1 tensor deflation from a tensor $\mathcal{Y}$ of rank-$(R + 1)$ which is on the left side. The second block on the right side is of rank of R. Two important properties of the rank-1 tensor deflation are exploited in this paper

- Estimation accuracy of the rank-1 tensor on the right side in Fig. 1 is comparable to that of this tensor in the rank-$(R + 1)$ tensor decomposition on the left side.
- Estimation accuracies of components in the rank-R decomposition of the second tensor on the right side are comparable to those of the components in the rank-$(R + 1)$ tensor decomposition on the left side.

Following these properties, we can add one or a few rank-

This work was supported by the Czech Science Foundation through Project No. 14-13713S.

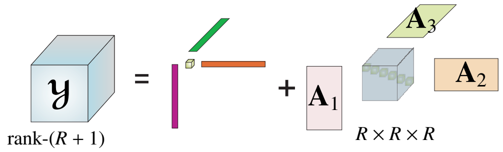

Fig. 1. Proceeding from left to right, the figure illustrates the rank-1 tensor deflation [20] which extracts a rank-1 tensor from the tensor $\mathcal{Y}$. However, proceeding from right to left, the rank-1 tensor injection method adds a rank-1 tensor into the data to create a new tensor which is applicable to the tensor deflation.

1 tensors into the data tensor to create a new tensor which is applicable to the deflation of the added rank-1 tensors. Instead of decomposition of the original tensor, we decompose the newly created tensor with rank of $(R + 1)$. For the rank-1 tensor injection method, the added rank-1 tensor is known and acts as a reference tensor. Therefore, we can control and stop the estimation process when estimated components of the reference tensor attain their Cramér–Rao error bounds.

The idea seems relatively simple. However, there are some technical issues which need to be resolved

- *How to generate a suitable injected or reference tensor?*
- *How to compute its error bound?*

Controlling the estimation accuracy of the added tensor is the main concern in our proposed method. We implement an automatic method which can control the estimation accuracy of the reference tensor, based on our recent developed Cramér–Rao–induced bound for CP decomposition [21] and the Cramér–Rao–induced bound for the tensor deflation [22].

The paper is organized as follows. The CANDECOMP/-PARAFAC tensor decomposition and the principle of the proposed method are introduced in Section 2. The main technique to control the accuracy of the added (reference) rank-1 tensor is presented in sub-section 2.1, whereas implementation of the proposed method is described in sub-section 2.2. Simulations are shown in Section 3. Section 4 concludes the paper.

2. INJECTION METHOD FOR CPD

Throughout the paper, we shall denote tensors by bold calligraphic letters, e.g., $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, matrices by bold capital letters, e.g., $\mathbf{A} = [\boldsymbol{a}_1, \boldsymbol{a}_2, \ldots, \boldsymbol{a}_R] \in \mathbb{R}^{I \times R}$, and vectors by bold italic letters, e.g., $\boldsymbol{a}_j$. The Kronecker, Khatri-Rao, Hadarmard (element-wise) and outer products are denoted by $\otimes$, $\odot$, $\circledast$, $\circ$ respectively.

The canonical polyadic decomposition or CANDECOMP/-PARAFAC approximates an order-N data tensor $\mathcal{Y}$ of size $I_1 \times I_2 \times \cdots \times I_N$ by a rank-R tensor $\widehat{\mathcal{Y}}$

$$\mathcal{Y} = \widehat{\mathcal{Y}} + \mathcal{E}, \tag{1}$$

where $\mathcal{E}$ represents the error tensor, which is often assumed to be Gaussian i.i.d.. The rank-R tensor $\widehat{\mathcal{Y}}$ is expressed as

$$\begin{aligned} \widehat{\mathcal{Y}} &= \sum_{r=1}^{R} \lambda_r \, \boldsymbol{a}_r^{(1)} \circ \boldsymbol{a}_r^{(2)} \circ \cdots \circ \boldsymbol{a}_r^{(N)} \\ &= [\![\boldsymbol{\lambda}; \mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \ldots, \mathbf{A}^{(N)}]\!]. \end{aligned}$$

where $\boldsymbol{\lambda} = [\lambda_1, \ldots, \lambda_R]$.

When almost all loading components are nearly collinear, as most of algorithms for CPD, the ALS algorithm does not work efficiently. This can be seen from its update rule

$$\mathbf{A}^{(n)} \leftarrow \mathbf{Y}_{(n)} \left(\bigodot_{k \neq n} \mathbf{A}^{(k)} \right) \left(\boldsymbol{\Gamma}^{(n)} \right)^{\dagger}, \quad (n = 1, 2, \ldots, N), \tag{2}$$

which needs to invert ill-conditioned matrices $\boldsymbol{\Gamma}^{(n)} = \mathbf{C}^{(1)} \circledast \cdots \circledast \mathbf{C}^{(n-1)} \circledast \mathbf{C}^{(n+1)} \circledast \cdots \circledast \mathbf{C}^{(N)}$, $\mathbf{C}^{(n)} = \mathbf{A}^{(n)T} \mathbf{A}^{(n)}$, where "†" denotes the pseudo-inverse, $\mathbf{Y}_{(n)}$ represent mode-n matricization of the tensor $\mathcal{Y}$.

In this section, instead of decomposition of $\mathcal{Y}$, we decompose another tensor which is generated from $\mathcal{Y}$ by simply adding a rank-1 tensor $\mathcal{X} = \alpha \, \boldsymbol{u}^{(1)} \circ \boldsymbol{u}^{(2)} \circ \cdots \circ \boldsymbol{u}^{(N)}$ into it, where $\alpha > 0$ and $\boldsymbol{u}^{(n)T} \boldsymbol{u}^{(n)} = 1$ for $n = 1, \ldots, N$

$$\begin{aligned} \widetilde{\mathcal{Y}} &= \mathcal{Y} + \alpha \, \boldsymbol{u}^{(1)} \circ \boldsymbol{u}^{(2)} \circ \cdots \circ \boldsymbol{u}^{(N)} \\ &= [\![[\boldsymbol{\lambda}, \alpha]; [\mathbf{A}^{(1)}, \boldsymbol{u}^{(1)}], [\mathbf{A}^{(2)}, \boldsymbol{u}^{(2)}], \ldots, [\mathbf{A}^{(N)}, \boldsymbol{u}^{(N)}]]\!] + \mathcal{E} \, . \end{aligned}$$

The injection method expands the factor matrices $\mathbf{A}^{(n)}$ by the components $\boldsymbol{u}^{(n)}$. With an appropriate choice of the tensor $\mathcal{X}$, the new tensor $\widetilde{\mathcal{Y}}$ will be of rank-$(R+1)$, whereas the injected tensor $\mathcal{X}$ can be deflated from $\widetilde{\mathcal{Y}}$ with a sufficient accuracy. In general, this can be achieved when the components $\boldsymbol{u}^{(n)}$ are linearly independent of $\mathbf{A}^{(n)}$, while its Cramér–Rao error bound should be small and attainable.

When the reference tensor $\mathcal{X}$ is able to be deflated from $\widetilde{\mathcal{Y}}$, we perform a rank-$(R + 1)$ decomposition of $\widetilde{\mathcal{Y}}$ to find the factors $\mathbf{A}^{(n)}$, while inspecting the estimation accuracy of the reference components $\boldsymbol{u}^{(n)}$. The algorithm stops only when estimation of $\boldsymbol{u}^{(n)}$ achieves the error bound. Next section will detail how to generate a suitable reference tensor.

2.1. Controlling the estimation accuracy of the rank-1 tensor $\mathcal{X}$

In practice, we can generate the components $\boldsymbol{u}^{(n)}$ randomly from a normal distribution with zero mean and unit variance, and normalize them to have unit length. However, the most important requirement is that the injected (added) tensor $\mathcal{X}$ should not make the decomposition harder. The components $\boldsymbol{u}^{(n)}$ of $\mathcal{X}$ need to be rather easily estimated, while the injected tensor $\mathcal{X}$ should not dominate the rank-1 tensor components of the tensor $\mathcal{Y}$. That is, its magnitude α should not be extensively higher than $\boldsymbol{\lambda}$ of the decomposition of $\mathcal{Y}$, because loading components are unit length vectors, otherwise, the algorithms tend to estimate only the added components $\boldsymbol{u}^{(n)}$, and underestimate the loading components of the original data $\mathbf{A}^{(n)}$. On the other hand, α should not be relatively small, otherwise, we cannot retrieve the components $\boldsymbol{u}^{(n)}$ accurately. For decomposition of tensors whose factors are of different magnitudes, we refer to the papers [6, 7]. Therefore, the most challenging problem in this method is to control the magnitudes of the injected (added) rank-1 tensors.

A naive method is that one can generate injected rank-1 tensors $\alpha\, \boldsymbol{u}^{(1)} \circ \boldsymbol{u}^{(2)} \circ \cdots \circ \boldsymbol{u}^{(N)}$ with different α. Based on the estimated results $\hat{\lambda}_r$ of λ_r, $\hat{\alpha}$ of α, together with estimation accuracy of $\boldsymbol{u}^{(n)}$, a further decomposition with lower or higher α can be taken. This accuracy controlling method is expensive, and may take plenty of trial-and-error before achieving a desired outcome. Indeed we do not recommend this method, but instead, we propose a more convenient method which can automatically control the accuracy of the injected tensor.

First of all, we will assess the estimation accuracy of injected components through the squared angular error [21], defined as

$$SAE(\boldsymbol{a}, \hat{\boldsymbol{a}}) = \arccos^2\left(\frac{\boldsymbol{a}^T \hat{\boldsymbol{a}}}{|\boldsymbol{a}||\hat{\boldsymbol{a}}|}\right) . \tag{3}$$

Cramér–Rao–induced bound (CRIB) on this error measure between the true and estimated vectors for CPD is derived in [21], whereas the CRIB for the tensor deflation is derived in [22]. The two error bounds are comparable, when a rank-1 tensor is applicable to the tensor deflation. Hence, they will be used to verify whether the tensor deflation conditions have been met. If these two bounds are significantly different, the rank-1 tensor $\mathcal{X}$ is not a good reference tensor, and should be replaced. Instead of comparing the two CRIBs, we compare the SAEs of components obtained by the rank-$(R+1)$ CPD and the rank-1 tensor deflation.

According to Theorem 3 in [21], the CRIB between the true and estimated vectors for CPD can be computed as

Theorem 3. [21]:

$$\text{CRIB}(\boldsymbol{u}^{(n)}) = \frac{\sigma^2}{\alpha^2}\left\{(I_n - 1) g_{n,1} - \text{tr}\left[\mathbf{B}_n\left((\boldsymbol{g}_n \boldsymbol{g}_n^T) \otimes \mathbf{T}_n\right)\right]\right\} \tag{4}$$

where $\boldsymbol{g}_n$, $\mathbf{B}_n$ and $\mathbf{T}_n$ are vectors and matrices which are computed from the expanded matrices $[\mathbf{A}^{(n)}, \boldsymbol{u}^{(n)}]$, $g_{n,1}$ is the first entry of $\boldsymbol{g}_n$, and σ^2 is the variance of the Gaussian noise. For detail definitions of the $\boldsymbol{g}_n$, $\mathbf{B}_n$ and $\mathbf{T}_n$, we refer to the paper [21].

Since the factor matrices $\mathbf{A}^{(n)}$ for $n = 1, \ldots, N$ are unknown, the CRIB on $\boldsymbol{u}^{(n)}$ cannot be computed explicitly. However, a beautiful property is that $\text{CRIB}_\alpha(\boldsymbol{u}^{(n)})$ is inversely proportional to α^2 (Theorem 5 [21]). More specifically, from (4) we have

Lemma 1. $\alpha^2\, CRIB_\alpha(\boldsymbol{u}^{(n)}) = CRIB_{\alpha=1}(\boldsymbol{u}^{(n)})$.

This relation also holds for the SAE of $\boldsymbol{u}^{(n)}$ when varying the weight α. Following this relation, we can adjust the weight α to an arbitrary accuracy degree, after only one decomposition. More specifically, we first generate an injected rank-1 tensor $\mathcal{X}$ with a large magnitude α. This setting will make the algorithm easy to estimate $\boldsymbol{u}^{(n)}$, and the SAEs of $\boldsymbol{u}^{(n)}$ attain their CRIBs. We decompose the new tensor $\widetilde{\mathcal{Y}}$, assess the $\text{SAE}(\boldsymbol{u}^{(n)}, \hat{\boldsymbol{u}}^{(n)})$ between $\boldsymbol{u}^{(n)}$ and their estimates $\hat{\boldsymbol{u}}^{(n)}$. The interest magnitude $\alpha_{interest}$ for some SAE of interest is then simply computed as

$$\alpha_{interest} = \alpha \sqrt{\frac{\text{SAE}(\boldsymbol{u}^{(n)}, \hat{\boldsymbol{u}}^{(n)})}{\text{SAE}_{interest}}} . \tag{5}$$

As above, SAE of $\boldsymbol{u}^{(n)}$ should be of a reasonable level. When the CRIB in (dB), i.e., $-10 \log_{10} \text{CRIB}(\boldsymbol{u}^{(n)})$ is lower than 20 dB, implying that the standard angular deviation of the factor exceeds 5.73 degrees, the components $\boldsymbol{u}^{(n)}$ is hard to be estimated, and the SAE often does not attain the CRIB [7, 21]. When the CRIB is of 25 dB, the estimation of $\boldsymbol{u}^{(n)}$ becomes easier, and its SAE often reaches the bound. In practice, we set the SAE of interest between 25 to 30 dB in the logarithmic scale, i.e., the angular error is about 2 or 3 degrees.

2.2. Implementation

The proposed injecting method comprises 5 steps and is summarised in Algorithm 1. In Stage 1, for decomposition of a rank-R tensor $\mathcal{Y}$, we first generate a random rank-1 tensor $\mathcal{X} = \boldsymbol{u}^{(1)} \circ \boldsymbol{u}^{(2)} \circ \cdots \circ \boldsymbol{u}^{(N)}$ with unit length components, $\boldsymbol{u}^{(n)T} \boldsymbol{u}^{(n)} = 1$ for all n.

In Stage 2, we perform a decomposition of rank-$(R+1)$ for the tensor $\mathcal{Y} + \alpha\, \mathcal{X}$. Here we can simply set $\alpha = \|\mathcal{Y}\|_F / \sqrt{R}$ or $\alpha = 1$ after normalizing the tensor $\mathcal{Y}$ to have unit Frobenius norm, i.e., $\|\mathcal{Y}\|_F = 1$. Note that the first round tensor decomposition is relatively fast, since the weight α is set to a high value, the CPD algorithm mostly estimates components $\boldsymbol{u}^{(n)}$ of the tensor $\mathcal{X}$. Moreover, the reference components are used to initialise the decomposition. The first decomposition can even proceed faster, when the alternating subspace update algorithm for tensor deflation is applied to extract the rank-1 reference tensor $\mathcal{X}$ [20]. Computational cost of this algorithm is only $O(R^3)$, cheaper than those of the ALS algorithm, i.e., $O(R^4)$, and any other algorithms for CPD.

Conditions for tensor deflation can be verified in this stage. Namely, we compare the SAEs of $\boldsymbol{u}^{(n)}$ with its estimated by CPD and the tensor deflation. In practice, the conditions often fulfil, and we can perform either the tensor deflation or the rank-$(R+1)$ CPD with high α.

In Stage 3, the weight α of the reference tensor is then adjusted as in (5) in which $-10 \log_{10}(\text{SAE}_{interest}) = 30$ dB, and mean SAEs of components are used in placed of SAE for each component $\boldsymbol{u}^{(n)}$.

In Stage 4, a further rank-$(R+1)$ CP decomposition of the new rank-expanded tensor is then executed, and the reference tensor $\mathcal{X}$ is used to control the estimation process. The algorithm will stop when SAEs of the components of $\mathcal{X}$ achieve the predefined accuracy SAE in Stage 3.

Finally, from output of the last decomposition $\mathbf{B}^{(n)}$, we select the components which are highly correlated with the reference $\boldsymbol{u}^{(n)}$. The rest R components of $\mathbf{B}^{(n)}$ are estimates of the factor matrices $\mathbf{A}^{(n)}$.

3. SIMULATIONS

In this section, we will illustrate efficiency of the injecting method for CP decomposition of order-3 tensors of size $I \times I \times I$

```
Algorithm 1: Injecting Method for CPD
Input: Data tensor $\mathcal{Y}$: $(I_1 \times I_2 \times \cdots \times I_N)$, rank $R$
Output: $\lambda \in \mathbb{R}^N$, $N$ matrices $\mathbf{A}^{(n)} \in \mathbb{R}^{I_n \times R}$
begin
    % Stage 1: Generate a random rank-1 tensor-----
    $\mathcal{X} = \boldsymbol{u}^{(1)} \circ \boldsymbol{u}^{(2)} \circ \cdots \circ \boldsymbol{u}^{(N)}$ where $\boldsymbol{u}^{(n)T}\boldsymbol{u}^{(n)} = 1$
    % Stage 2: Decompose the rank-expanded tensor--
    $[\![\gamma; \mathbf{B}^{(1)}, \ldots, \mathbf{B}^{(N)}]\!] = \mathtt{CPD}(\mathcal{Y} + \mathcal{X}, R + 1)$    % Or
    $[\![\gamma; \boldsymbol{b}^{(1)}, \ldots, \boldsymbol{b}^{(N)}]\!] = \mathtt{Tensor\_Deflation}(\mathcal{Y} + \mathcal{X})$
    % Stage 3: Check the SAEs($\boldsymbol{u}^{(n)}, \hat{\boldsymbol{u}}^{(n)}$) and adjust $\alpha$
    $\alpha_{new} = \sqrt{\dfrac{\sum_n \mathrm{SAE}_n(\boldsymbol{u}^{(n)}, \hat{\boldsymbol{u}}^{(n)})}{N\,\mathrm{SAE}_{interest}}}$
    % Stage 4: Decompose the new tensor------------
    $[\![\gamma; \mathbf{B}^{(1)}, \ldots, \mathbf{B}^{(N)}]\!] = \mathtt{CPD}(\mathcal{Y} + \alpha_{new}\mathcal{X}, R + 1)$
    % Stage 5: Set output $\mathbf{A}^{(n)}$ as $R$ columns of $\mathbf{B}^{(n)}$ -
```

Table 1. Comparison of ALS and ALS with injection method in decomposition of order-3 tensors. CRIB and InjCRIB show CRIBs of components of the original tensors, and CRIBs of the components in the rank-expanded tensors, respectively.

	$R = 5$			$R = 10$		
	SAE (dB)	No. Iters	Error (dB)	SAE (dB)	No. Iters	Error (dB)
CRIB	35.46			35.93		
InjCRIB	35.42			35.92		
ALS	35.22	710	20.12	35.86	922	20.07
Inj-ALS	35.17	63	20.12	35.82	295	20.07
	$R = 15$			$R = 20$		
	SAE (dB)	No. Iters	Error (dB)	SAE (dB)	No. Iters	Error (dB)
CRIB	36.19			36.34		
InjCRIB	36.18			36.34		
ALS	36.18	1066	20.05	36.18	1144	20.04
Inj-ALS	36.17	501	20.05	36.10	639	20.04

and rank-R, where $I = 3R$ and $R = 5$, 10, 15 and 20. Components of the second and third factor matrices were highly collinear, with $\boldsymbol{a}_r^{(2)T}\boldsymbol{a}_s^{(2)} = \boldsymbol{a}_r^{(3)T}\boldsymbol{a}_s^{(3)} = 0.98$ for all $r \neq s$, whereas $\boldsymbol{a}_r^{(1)T}\boldsymbol{a}_s^{(1)} = 0.2$ and $\boldsymbol{a}_r^{(n)T}\boldsymbol{a}_r^{(n)} = 1$ for all n. The coefficients λ_r were set to 1. Gaussian noise was added into the tensor $\mathcal{Y}$ with Signal-Noise-Ratio SNR = 40 dB. The noise level was kept to relatively low in order to preserve the structure of loading components $\boldsymbol{u}^{(n)}$. The mean CRIBs on components when SNR = 40 dB were around 35 dB as shown in Table 1. For higher noise, e.g., SNR = 10 or even 20 dB, the CRIBs of the components were only 5 or 15 dB, and it was nearly impossible to retrieve $\mathbf{A}^{(n)}$ accurately.

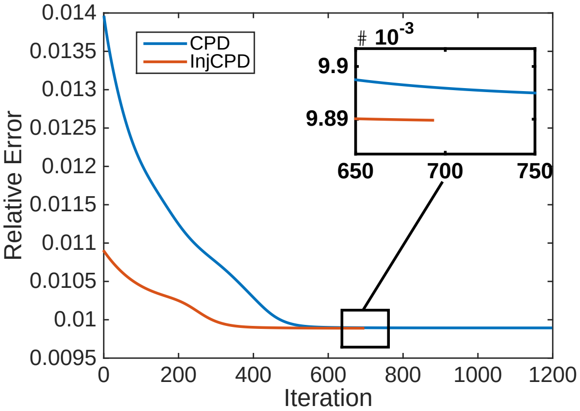

Fig. 2. Illustration of relative errors of the ALS in decomposition of a tensor of rank-20 (CPD) and its rank-1 expanding tensor (InjCPD).

We compare performances of the ALS algorithm for CPD and CPD with injection (Inj+ALS). Initial values were generated using the leading singular vectors, and passed through the fLM algorithm [7] in only 10 iterations. This is just to generate good initial points for both algorithms ALS and Inj+ALS. The algorithms stopped when the relative errors $\varepsilon = \dfrac{\|\mathcal{Y} - \widehat{\mathcal{Y}}\|_F}{\|\mathcal{Y}\|_F}$ were lower than 10^{-8}, or until the maximum number of iterations (2000) was achieved. In addition, for the Inj+ALS algorithm, the reference was tracked until its estimate achieved a predefined SAE of 30 dB.

There were 100 independent runs for each setting of rank R. The mean squared angular error over all components, the number of iterations and relative errors (in dB) are compared in Table 1. While the two algorithms achieved similar results confirmed by the MSAEs and the relative approximation errors, the ALS needs approximately 600 more iterations than ALS with injection. This is because the injection method controlled the estimation process. In Fig. 2, we compare relative errors ε of CPD and CPD with injecting method as function of iterations, when $R = 20$. The decomposition with injected tensor was stopped earlier when SAEs of the reference components were achieved.

4. CONCLUSIONS

We have presented an application of the tensor deflation to control the estimation process for CPD by adding rank-1 tensor into the analysed data. The proposed method is based on our newly developed Cramér–Rao–induced bounds for CPD and the tensor deflation, and can automatically adjust the reference tensor to achieve a predefined estimation accuracy. The rank-1 tensor injection procedure is useful to many other algorithms not only to ALS. The method mainly can control the estimation accuracy of the CP decomposition. However, since the factor matrices $\mathbf{A}^{(n)}$ are expanded by linearly independent vectors $\boldsymbol{u}^{(n)}$, the new factor matrices do not comprise all highly collinear vectors, and the new tensor $\widetilde{\mathcal{Y}}$ tends to be easier to decompose than $\mathcal{Y}$. The injection method is also useful to choose a good result among many possible outcomes by different initial values. The proposed method can be used to inspect rank of a tensor. These further applications will be presented in a forthcoming extension of this paper.

5. REFERENCES

[1] A. Cichocki, R. Zdunek, A.-H. Phan, and S. Amari, *Nonnegative Matrix and Tensor Factorizations: Applications to Exploratory Multi-way Data Analysis and Blind Source Separation*, Wiley, Chichester, 2009.

[2] P. Comon, "Tensors: A brief introduction," *IEEE Signal Processing Magazine*, vol. 31, no. 1, pp. 44–53, 2014.

[3] A. Cichocki, D. P. Mandic, A.-H. Phan, C. Caifa, G. Zhou, Q. Zhao, and L. De Lathauwer, "Tensor decompositions for signal processing applications. from two-way to multiway component analysis," *IEEE Signal Processing Magazine*, vol. 32, no. 2, pp. 145–163, 2015.

[4] R.A. Harshman, "Foundations of the PARAFAC procedure: Models and conditions for an explanatory multimodal factor analysis," *UCLA Working Papers in Phonetics*, vol. 16, pp. 1–84, 1970.

[5] J.D. Carroll and J.J. Chang, "Analysis of individual differences in multidimensional scaling via an *n*-way generalization of Eckart–Young decomposition," *Psychometrika*, vol. 35, no. 3, pp. 283–319, 1970.

[6] P. Paatero, "A weighted non-negative least squares algorithm for three-way PARAFAC factor analysis," *Chemometrics Intelligent Laboratory Systems*, vol. 38, no. 2, pp. 223–242, 1997.

[7] A.-H. Phan, P. Tichavský, and A. Cichocki, "Low complexity damped Gauss-Newton algorithms for CANDECOMP/PARAFAC," *SIAM Journal on Matrix Analysis and Applications*, vol. 34, no. 1, pp. 126–147, 2013.

[8] L. Sorber, M. Van Barel, and L. De Lathauwer, "Structured data fusion," Tech. Rep., ESAT-SISTA, Internal Report 13-177, 2013.

[9] A.-H. Phan, P. Tichavský, and A. Cichocki, "Fast alternating LS algorithms for high order CANDECOMP/PARAFAC tensor factorizations," *IEEE Transactions on Signal Processing*, vol. 61, no. 19, pp. 4834–4846, 2013.

[10] A. Cichocki and A.-H. Phan, "Fast local algorithms for large scale nonnegative matrix and tensor factorizations," *IEICE Transactions*, vol. 92-A, no. 3, pp. 708–721, 2009.

[11] M. B. Dosse, J. M.F. Berge, and J. N. Tendeiro, "Some new results on orthogonally constrained Candecomp," *Journal of Classification*, vol. 28, pp. 144–155, 2011.

[12] M. Sørensen, L. Lathauwer, P. Comon, S. Icart, and L. Deneire, "Canonical polyadic decomposition with a columnwise orthonormal factor matrix," *SIAM Journal on Matrix Analysis and Applications*, vol. 33, no. 4, pp. 1190–1213, 2012.

[13] B. C. Mitchell and D. S. Burdick, "Slowly converging PARAFAC sequences: Swamps and two-factor degeneracies," *Journal of Chemometrics*, vol. 8, pp. 155–168, 1994.

[14] P. Comon, X. Luciani, and A. L. F. de Almeida, "Tensor decompositions, alternating least squares and other tales," *Journal of Chemometrics*, vol. 23, 2009.

[15] R. Bro, *Multi-way Analysis in the Food Industry - Models, Algorithms, and Applications*, Ph.D. thesis, University of Amsterdam, Holland, 1998.

[16] G. Tomasi, *Practical and Computational Aspects in Chemometric Data Analysis*, Ph.D. thesis, Frederiksberg, Denmark, 2006.

[17] M. Rajih, P. Comon, and R. A. Harshman, "Enhanced line search: A novel method to accelerate PARAFAC," *SIAM Journal of Matrix Analysis and Applications*, vol. 30, no. 3, pp. 1128–1147, 2008.

[18] Y. Chen, D. Han, and L. Qi, "New ALS methods with extrapolating search directions and optimal step size for complex-valued tensor decompositions," *IEEE Transactions on Signal Processing*, vol. 59, no. 12, pp. 5888–5898, 2011.

[19] P. Paatero, C. Navasca, and P. Hopke, "Fast rotationally enhanced alternating-least-squares," Workshop on Tensor Decompositions and Applications (TDA 2010), SIAM, 2010.

[20] A.-H. Phan, P. Tichavský, and A. Cichocki, "Tensor deflation for CANDECOMP/PARAFAC. Part 1: Alternating Subspace Update Algorithm," *IEEE Transaction on Signal Processing*, vol. 63, no. 12, pp. 5924–5938, 2015.

[21] P. Tichavský, A.-H. Phan, and Z. Koldovský, "Cramér-Rao-induced bounds for CANDECOMP/PARAFAC tensor decomposition," *IEEE Transactions on Signal Processing*, vol. 61, no. 8, pp. 1986–1997, 2013.

[22] A.-H. Phan, P. Tichavský, and A. Cichocki, "Tensor deflation for CANDECOMP/PARAFAC. Part 2: Initialization and Error analysis.," *IEEE Transaction on Signal Processing*, vol. 63, no. 12, pp. 5939–5950, 2015.