Sensitivity in Tensor Decomposition

Petr Tichavský, Anh-Huy Phan, and Andrzej Cichocki

Abstract—**Canonical polyadic (CP) tensor decomposition is an important task in many applications. Many times, the true tensor rank is not known, or noise is present, and in such situations, different existing CP decomposition algorithms provide very different results. In this letter, we introduce a notion of *sensitivity* of CP decomposition and suggest to use it as a side criterion (besides the fitting error) to evaluate different CP decomposition results. Next, we propose a novel variant of a Krylov-Levenberg-Marquardt CP decomposition algorithm which may serve for CP decomposition with a constraint on the sensitivity. In simulations, we decompose order-4 tensors that come from convolutional neural networks. We show that it is useful to combine the CP decomposition algorithms with an error-preserving correction.**

I. Introduction

TENSOR decompositions and especially the canonical polyadic tensor representations are increasingly popular since 70 s and the number of papers related to this problem grows constantly. Also, the number of their applications is increasing. For example, recently tensor decompositions were used to speed-up learning and inference of deep neural networks, namely convolutional neural networks (CNN's) [1]–[8].

The workhorse algorithm, the Alternating Least Squares (ALS) often exhibits a slow convergence [9], [10]. Significantly faster convergence is obtained by second-order methods like Gauss-Newton or damped Gauss-Newton method (Levenberg-Marquardt, LM) [16], [17]. These methods need computation of Hessian for the problem. Although inversion of the Hessian can be evaluated relatively easily thanks to its special structure, these methods still require to store the Hessian in computer memory. This issue is alleviated in conjugate gradient algorithms, and in nonlinear optimization algorithm, e.g., in Tensorlab [11]–[13].

Although the existing algorithms are quite fast and seem powerful, there is one aspect that is mostly overlooked in the existing literature. Usually, it is assumed that the tensor decomposition is unique, and all CP decomposition algorithms should converge to the same solution. In practice, however, the picture is often much more complex. Usually, given data contain noise of some kind, and numerically one might obtain very different results by doing CP decomposition. Moreover, the true rank is often unknown, so that we should rather talk about a small rank approximation of the data rather than CP decomposition. Some practitioners use various kind of regularizations to avoid unstable decompositions, but the problem is that there is no unique methodology allowing to compare the performance of different penalty terms.

In this letter, we propose a novel criterion to measure the sensitivity of approximate CP decompositions with respect to small deviations in the estimated factor matrices. Especially in the application in CNN's, the sensitivity of the decomposition is important. The main idea here is to replace one convolution layer with many coefficients by several simpler layers corresponding to individual factor matrices of the tensor. The tensor build of the factor matrices should not be overly sensitive to small changes in the factor matrices.

It appears in our simulations that the sensitivity of CP decomposition is related to the Error Preserving Correction (EPC) [26]. EPC seeks, given approximate CP decomposition, another approximation with the same rank and the same fitting error (measured as a Frobenius norm of the error), having a lower sum of squared Frobenius norm of the rank-one components. Although the criterion of EPC is different, minimizing it usually reduces the sensitivity as well. We show this later in the letter.

Second, we propose a novel variant of a recent CP tensor decomposition algorithm called KLM (Krylov-Levenberg-Marquardt), which minimizes the fitting error under the condition of the limited sensitivity of the decomposition. The key idea behind the KLM algorithm is a low-rank approximation of the Hessian, which is done through the Krylov subspace [14], [15], [22], using the fact, that products of the Hessian $\mathbf{H}$ with an arbitrary vector $\mathbf{x}$ of the appropriate dimension can be computed very efficiently thanks to a special structure of the Hessian, without actually forming the Hessian [12]. The number of operations per iteration is dominated by computing the error gradient (matricized tensor times KhatriRao product, MTTKRP), i.e. it is basically the same as in the ALS method [10]. The rank of the approximation of the Hessian is a design variable of the technique. A higher rank results in increasing complexity of each iteration, but it decreases the number of iterations needed to achieve the convergence. Some tuning here might be necessary to achieve the optimum performance, but usually even ad hoc choice of the rank gives a method outperforming other methods. An extension of the KLM algorithm for the case of weighted CP decomposition has been submitted as [19].

Manuscript received July 3, 2019; revised September 18, 2019; accepted September 18, 2019. Date of publication September 23, 2019; date of current version October 7, 2019. This work was supported by the Ministry of Education and Science of the Russian Federation under Grant 14.756.31.0001. The work of P. Tichavský was supported by the Czech Science Foundation through Project 17-00902S. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Joao Paulo Paulo Papa. *(Corresponding author: Petr Tichavský.)*

P. Tichavský is with the Czech Academy of Sciences, Institute of Information Theory and Automation, 18208 Prague, Czech Republic (e-mail: tichavsk@utia.cas.cz).

A.-H. Phan and A. Cichocki are with the Skolkovo Institute of Science and Technology, 143026 Moscow, Russia (e-mail: a.phan@skolkovo.ru; A.Cichocki@skolkovo.ru).

Digital Object Identifier 10.1109/LSP.2019.2943060

1070-9908 © 2019 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

The rest of the letter is organized as follows. Section II introduces the notion of the sensitivity of CP decomposition. Section III presents the basics of the Levenberg-Marquardt method and its constrained version. The KLM method is explained in Section IV. Section V contains numerical examples, where the performance of the proposed techniques is compared with state-of-the-art techniques in Tensorlab. Section VI concludes the letter.

II. Sensitivity of CP Tensor Decomposition

For simplicity, we explain the idea on the case of the order-3 tensor. Assume that a tensor $\mathcal{T}$ is approximated as

$$\mathcal{T} \approx \mathcal{T}_A = [[\mathbf{A}, \mathbf{B}, \mathbf{C}]], \tag{1}$$

where $\mathbf{A}, \mathbf{B}$, and $\mathbf{C}$ are factor matrices of the approximate CP decomposition. The notation $[[\ldots]]$ means that

$$\mathcal{T}_A = \sum_{r=1}^{R} \mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{c}_r \tag{2}$$

where $\circ$ denotes the outer (tensor) product, R is the rank of the approximation, and $\{\mathbf{a}_r\}$, $\{\mathbf{b}_r\}$, and $\{\mathbf{c}_r\}$ are columns of the factor matrices $\mathbf{A}, \mathbf{B}$, and $\mathbf{C}$, respectively.

Most CP decomposition methods minimize the cost function

$$\varphi(\mathcal{T}, \mathbf{A}, \mathbf{B}, \mathbf{C}) = \|\mathcal{T} - [[\mathbf{A}, \mathbf{B}, \mathbf{C}]]\|_F^2. \tag{3}$$

where $\|\cdot\|$ is the Frobenius norm.

The fitting error should never be the only performance criterion. In [26], we proposed to monitor the sum of squared Frobenius norm of the individual rank-one components,

$$\begin{aligned} SFN(\mathbf{A}, \mathbf{B}, \mathbf{C}) &= \sum_{r=1}^{R} \|[[\mathbf{a}_r, \mathbf{b}_r, \mathbf{c}_r]]\|_F^2 \\ &= \sum_{r=1}^{R} \|\mathbf{a}_r\|^2 \|\mathbf{b}_r\|^2 \|\mathbf{c}_r\|^2. \end{aligned}$$

The idea was to avoid situations in which the estimated rank-one components have large Frobenius norms and cancel each other. We have designed an algorithm which seeks, for a given decomposition, another decomposition, which has the same fitting error but has reduced value of $SFN(\mathbf{A}, \mathbf{B}, \mathbf{C})$. It is called Error Preserving Correction (EPC). The technique has proved to improve the convergence of CP decomposition algorithms.

In [18], it was proposed to take the sum of Frobenius norms of the factor matrices as a constraint,

$$\begin{aligned} SSN(\mathbf{A}, \mathbf{B}, \mathbf{C}) &= \|\mathbf{A}\|_F^2 + \|\mathbf{B}\|_F^2 + \|\mathbf{C}\|_F^2 \\ &= \sum_{r=1}^{R} \|\mathbf{a}_r\|^2 + \|\mathbf{b}_r\|^2 + \|\mathbf{c}_r\|^2. \end{aligned}$$

The constraint was $SSN(\mathbf{A}, \mathbf{B}, \mathbf{C}) = c$, where c was a suitable constant. The technique helped to decompose some matrix multiplication tensors.

In this letter, we propose something similar, but numerically slightly different. Assume that the factor matrices are perturbed by random matrices $d\mathbf{A}$, $d\mathbf{B}$, $d\mathbf{C}$, having random i.i.d. elements with Gaussian distribution of zero mean and variance σ^2. Then, the sensitivity of the decomposition can be measured by the expectation of the normalized squared Frobenius norm of the difference

$$s(\mathbf{A}, \mathbf{B}, \mathbf{C}) = \mathrm{E}\|\mathcal{T}_A - [[\mathbf{A} + d\mathbf{A}, \mathbf{B} + d\mathbf{B}, \mathbf{C} + d\mathbf{C}]]\|_F^2/\sigma^2.$$

where $\mathrm{E}\{.\}$ is the expectation operator. In other words, we study sensitivity of the tensor $\mathcal{T}_A = [[\mathbf{A}, \mathbf{B}, \mathbf{C}]]$ with respect to perturbations in individual factor matrices.

After a straightforward computation we get

$$\begin{aligned} s(\mathbf{A}, \mathbf{B}, \mathbf{C}) &= \mathrm{tr}((\mathbf{A}^T\mathbf{A}) * (\mathbf{B}^T\mathbf{B}) + (\mathbf{B}^T\mathbf{B}) * (\mathbf{C}^T\mathbf{C}) \\ &\quad + (\mathbf{A}^T\mathbf{A}) * (\mathbf{C}^T\mathbf{C})) \\ &= \sum_{r=1}^{R} \|\mathbf{a}_r\|^2(\|\mathbf{b}_r\|^2 + \|\mathbf{c}_r\|^2) + \|\mathbf{b}_r\|^2\|\mathbf{c}_r\|^2 \end{aligned}$$

where "tr" is the matrix trace (sum of diagonal elements), and "*" is the elementwise (Hadamard) product.

Note that in order to achieve the minimum sensitivity, the power of each rank-1 component should be uniformly distributed in all modes, i.e. we should have

$$\|\mathbf{a}_r\|^2 = \|\mathbf{b}_r\|^2 = \|\mathbf{c}_r\|^2 \tag{4}$$

for all $r = 1, \ldots, R$. This condition can be achieved by appropriate scaling of the factors.

For decomposition of order-4 tensors, $\mathcal{T}_A = [[\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}]]$, the sensitivity is defined analogously. The resultant expression is

$$\begin{aligned} s(\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}) = \sum_{r=1}^{R} &\|\mathbf{a}_r\|^2\|\mathbf{b}_r\|^2(\|\mathbf{c}_r\|^2 + \|\mathbf{d}_r\|^2) \\ &+ \|\mathbf{c}_r\|^2\|\mathbf{d}_r\|^2(\|\mathbf{a}_r\|^2 + \|\mathbf{b}_r\|^2) \end{aligned} \tag{5}$$

where $\{\mathbf{d}_r\}$ are columns of the factor matrix $\mathbf{D}$.

III. Levenberg–Marquardt Technique and Constrained LM

Let the estimated parameters-factor matrices $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$ - be arranged in one long vector $\boldsymbol{\theta}$,

$$\boldsymbol{\theta} = [\mathrm{vec}(\mathbf{A})^T, \mathrm{vec}(\mathbf{B})^T, \mathrm{vec}(\mathbf{C})^T]^T. \tag{6}$$

Levenberg-Marquardt method consists in iterations

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - (\mathbf{H} + \mu\mathbf{I})^{-1}\mathbf{g} \tag{7}$$

where

$$\mathbf{H} = \mathbf{J}^T\mathbf{J}, \qquad \mathbf{J} = \frac{\partial \mathrm{vec}(\mathcal{T}_A(\boldsymbol{\theta}))}{\partial \boldsymbol{\theta}}, \quad \mathbf{g} = \mathbf{J}^T \mathrm{vec}(\mathcal{T} - \mathcal{T}_A(\boldsymbol{\theta})) \tag{8}$$

and μ is a damping parameter, which is sequentially updated according to a rule described in [24]. Closed-form expressions for the approximate Hessian $\mathbf{H}$ and gradient $\mathbf{g}$ will be discussed later, for further details see [16].

In some difficult scenarios, it might be suitable to optimize parameter μ (perform a rough line search) in each iteration step of the main loop. Then, each iteration step would be more complex, but the number of iterations needed to achieve convergence might be reduced. The algorithm is summarized in Table 1.

A. Constrained Optimization

In this subsection, a simple modification of the LM method to a constrained optimization is proposed. Assume that we wish to optimize the cost function (3) under the constraint $c(\boldsymbol{\theta}) = c_0$. We mainly consider the constraint function $c(\boldsymbol{\theta}) = s(\mathbf{A}, \mathbf{B}, \mathbf{C})$. The idea is to minimize the second-order approximation of the cost function at the point of the current approximation $\boldsymbol{\theta}_0$,

$$\varphi(\boldsymbol{\theta}) \approx \varphi(\boldsymbol{\theta}_0) + \mathbf{g}^T(\boldsymbol{\theta} - \boldsymbol{\theta}_0) + \frac{1}{2}(\boldsymbol{\theta} - \boldsymbol{\theta}_0)^T\mathbf{H}(\boldsymbol{\theta} - \boldsymbol{\theta}_0). \quad (9)$$

Instead of the true constraint function, we shall minimize the approximation in (9) in the tangent plane to the corresponding variety,

$$(\boldsymbol{\theta} - \boldsymbol{\theta}_0)^T\mathbf{u}_0 = 0 \qquad \text{where} \qquad \mathbf{u}_0 = \left.\frac{\partial c(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}\right|_{\boldsymbol{\theta}=\boldsymbol{\theta}_0}. \quad (10)$$

The augmented criterion to be minimized is

$$\mathbf{g}^T(\boldsymbol{\theta} - \boldsymbol{\theta}_0) + \frac{1}{2}(\boldsymbol{\theta} - \boldsymbol{\theta}_0)^T\mathbf{H}(\boldsymbol{\theta} - \boldsymbol{\theta}_0) - \lambda(\boldsymbol{\theta} - \boldsymbol{\theta}_0)^T\mathbf{u}_0$$

where λ is a Lagrange multiplier. The optimum $\boldsymbol{\theta}$ is

$$\boldsymbol{\theta}_1' = \boldsymbol{\theta}_0 - \mathbf{H}^{-1}(\mathbf{g} - \lambda\mathbf{u}_0) \quad (11)$$

The λ for which the solution obeys the approximate constraint is $\lambda = \mathbf{u}_0^T\mathbf{H}^{-1}\mathbf{g}/(\mathbf{u}_0^T\mathbf{H}^{-1}\mathbf{u}_0)$. Thus, the minimizer in the tangent plane would be

$$\boldsymbol{\theta}_1' = \boldsymbol{\theta}_0 - \mathbf{H}^{-1}\mathbf{g} + \frac{\mathbf{u}_0^T\mathbf{H}^{-1}\mathbf{g}}{\mathbf{u}_0^T\mathbf{H}^{-1}\mathbf{u}_0}\mathbf{H}^{-1}\mathbf{u}_0. \quad (12)$$

Instead of using (12) directly, we propose replacing $\mathbf{H}^{-1}$ by $(\mathbf{H} + \mu\mathbf{I})^{-1}$ as in the LM method in order to avoid inverting $\mathbf{H}$, which is singular in our case. The next iteration of $\boldsymbol{\theta}$ would be obtained by an appropriate scale change, $\boldsymbol{\theta}_1 = \alpha\boldsymbol{\theta}_1'$, where $\alpha = (c_0/c(\boldsymbol{\theta}_1'))^{1/4}$, so that the condition $c(\boldsymbol{\theta}_1) = c_0$ is fulfilled.

The proposed method works well, but we do not claim its optimality. There are other nonlinear optimization methods [25]. Selection of the optimum method will be subject of a further research.

IV. Krylov Subspace Technique

In the Levenberg-Marquardt algorithm, it is needed to compute products of the type $(\mathbf{H} + \mu\mathbf{I})^{-1}\mathbf{g}$, where $\mathbf{H}$ is a symmetric positive semidefinite matrix and $\mathbf{g}$ is a vector called "error gradient". Assume that we can quickly compute the product $\mathbf{y} = \mathbf{H}\mathbf{x}$ for arbitrary vector $\mathbf{x}$ of appropriate dimension, without the need of having the matrix $\mathbf{H}$ available directly.

The main idea is to work with a low-rank approximation of the Hessian $\mathbf{H}$ of size $N \times N$, which may look as

$$\mathbf{H} \approx \mathbf{U}\mathbf{W}\mathbf{U}^T \quad (13)$$

where $\mathbf{U}$ is a tall matrix of the size $N \times M$, $M \ll N$, and $\mathbf{W}$ would be a symmetric $M \times M$ matrix. Then, the integer M can be called rank of the approximation. Given (13) and applying the matrix inversion lemma, the expression $(\mathbf{H} + \mu\mathbf{I})^{-1}\mathbf{g}$ can be estimated as

$$(\mathbf{H} + \mu\mathbf{I})^{-1}\mathbf{g} \approx \frac{1}{\mu}\mathbf{g} - \frac{1}{\mu}\mathbf{U}(\mu\mathbf{W}^{-1} + \mathbf{U}^T\mathbf{U})^{-1}(\mathbf{U}^T\mathbf{g}). \quad (14)$$

This computation requires inversion of two matrices of the size $M \times M$, which is not demanding, if M is not too big. In total it requires $O(NM + M^3)$ operations.

The Krylov method [14], [15], being inspired by power method of estimating eigenvalues and corresponding eigenvectors, takes $\mathbf{U}$ as the orthogonal basis of the linear hull of

$$[\mathbf{g}, \mathbf{H}\mathbf{g}, \mathbf{H}^2\mathbf{g}, \ldots, \mathbf{H}^{M-1}\mathbf{g}].$$

In the Krylov basis, the first vector is taken arbitrarily or at random, the only requirement is that it should not be orthogonal to the leading eigenvectors. In our application, since we want to compute (14), the initial vector is taken as the error gradient $\mathbf{g}$. Let $\mathbf{U} = [\mathbf{u}_1, \ldots, \mathbf{u}_M]$. Computing the basis $\mathbf{U}$ can proceed through the Gram-Schmidt orthogonalization process

$$\mathbf{u}_1 = \frac{\mathbf{g}}{\|\mathbf{g}\|} \quad (15)$$

$$\mathbf{u}_{i+1}' = \mathbf{H}\mathbf{u}_i - \sum_{j=1}^{i}\mathbf{u}_j^T(\mathbf{H}\mathbf{u}_i)\mathbf{u}_j \quad (16)$$

$$\mathbf{u}_{i+1} = \frac{\mathbf{u}_{i+1}'}{\|\mathbf{u}_{i+1}'\|} \quad \text{for } i = 1, \ldots, M-1. \quad (17)$$

In this construction, $\mathbf{U}$ is orthogonal, hence $\mathbf{U}^T\mathbf{U}$ in (14) is the identity matrix. Next, the matrix $\mathbf{W}$ in (13) can be obtained by multiplying by $\mathbf{U}^T$ and $\mathbf{U}$ from the left and from the right, respectively,

$$\mathbf{W} = \mathbf{U}^T\mathbf{H}\mathbf{U}. \quad (18)$$

Note that $\mathbf{W}$ is computed as a side product of the Gram-Schmidt orthogonalization.

The product $\mathbf{y} = \mathbf{H}\mathbf{x}$ can be computed with $O(IR^2)$ operations, where $I = I_A + I_B + I_C$ and (I_A, I_B, I_C) is the size of the tensor [12], [19]. One iteration of KLM executes this computation M times. The Gram-Schmidt orthogonalization requires additional $O(IMR^2)$ operations. Computing the error gradient $\mathbf{g}$ requires $O(I_AI_BI_CR)$ operations. The total complexity per iteration of KLM is then $O(I_AI_BI_CR + IM^2R + M^3)$ operations.

V. Simulations

A. Example 1: An Order-4 Tensor

We decomposed the tensor of size $5 \times 5 \times 48 \times 256$ that is a part of the second convolutional layer of the celebrated neural network Alex-net. Performance of KLM is compared to the performance of the Nonlinear Least Squares (NLS) method of Tensorlab. We can see a big difference between these two algorithms: NLS is intended to minimize its cost function and it does it very well and quickly. However, the decompositions obtained by this method have a much higher sensitivity to possible

TABLE I
OUTLINE OF THE KLM ALGORITHM WITH A LINE SEARCH

```
Input: tensor $\mathcal{Y}$, initial factor matrices $\mathbf{A}, \mathbf{B}, \mathbf{C}$ of $R$ columns, integer $M$,
range $\mathcal{R}$ of parameter $\mu$, say $\mathcal{R} = \text{logspace}(-2, 2, 20)$.
Repeat (until convergence)
    1) Compute the error gradient $\mathbf{g} = [\mathbf{g}_A; \mathbf{g}_B; \mathbf{g}_C]$.
    2) Compute basis $\mathbf{U}$ of the Krylov space of rank-$M$
       and corresponding matrix $\mathbf{W}$.
    3) For all $\mu \in \mathcal{R}$ compute
       $\mathbf{d}(\mu) = (\mathbf{H} + \mu\mathbf{I})^{-1}\mathbf{g}$ by (14)
       Define $\mathbf{d}_A, \mathbf{d}_B, \mathbf{d}_C$ by $\mathbf{d}(\mu) = [\text{vec}(\mathbf{d}_A); \text{vec}(\mathbf{d}_B); \text{vec}(\mathbf{d}_C)]$
       $\mathbf{A}'(\mu) = \mathbf{A} + \mathbf{d}_A$, $\mathbf{B}'(\mu) = \mathbf{B} + \mathbf{d}_B$, $\mathbf{C}'(\mu) = \mathbf{C} + \mathbf{d}_C$
       error$(\mu) = \|\mathcal{T} - [\mathbf{A}'(\mu), \mathbf{B}'(\mu), \mathbf{C}'(\mu)]\|^2$
    4) Select the $\mu$ with the minimum error$(\mu)$ and set the corresponding
       update of $\mathbf{A}'(\mu), \mathbf{B}'(\mu), \mathbf{C}'(\mu)$ as the next iteration of $\mathbf{A}, \mathbf{B}, \mathbf{C}$.
End
```

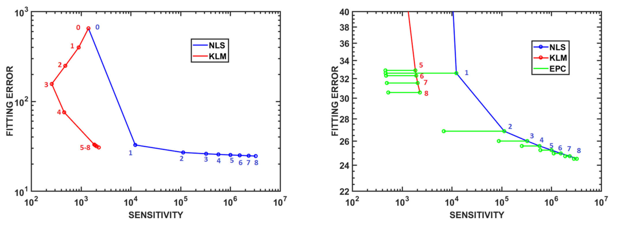

Fig. 1. Performance of Tensorlab and KLM method with $M = 50$ and rank $R = 200$ (left diagram) and performance after applying EPC (right diagram).

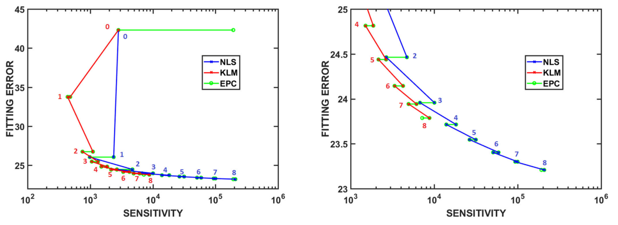

Fig. 2. Performance of Tensorlab and KLM method combined with EPC, fitting error versus sensitivity. Order-4 tensor, $R = 200$.

errors or perturbations in the factor matrices. We demonstrate this phenomenon in the plane of the sensitivity versus the fitting error. We start both algorithms from the same initial point obtained by random factor matrices followed with five iterations of NLS. Then, we run the algorithms, and measure fitting error and sensitivity after every $N(j)$ iterations, where $N(j)$ are logarithmically distributed between 10 and 1000, $j = 1, \ldots, 8$. The results are plotted in Fig. 1. We can see that NLS converges much faster than KLM, but NLS produces results with much greater sensitivity.

The sensitivity of the decompositions can be reduced by applying the error preserving correction (EPC) called Alternating Error Preserving Correction (AEPC) [26], in particular, 200 iterations of it. We started with the solutions provided in Fig. 1, and the results are shown in Fig. 2. We can see that the AEPC algorithm works well for not very low fitting errors, and reduces the sensitivity significantly. However, when the fitting error becomes close to its minimum, the sensitivity is large, and it can be reduced only a little.

In the next experiment, we tried to improve the previous results by alternating between NLS/KLM and EPC. After executing $N(j)$ steps of algorithms, we apply 200 iterations of EPC

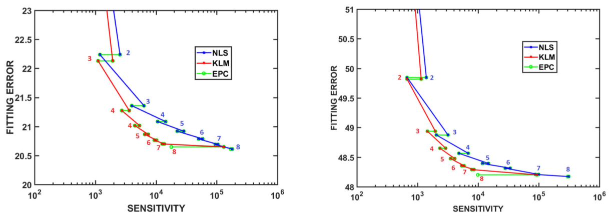

Fig. 3. Performance of NLS + EPC and KLM + EPC methods for order-3 tensor and rank $R = 200$ (left diagram) and $R = 100$ (right diagram).

(AEPC). In this way, a better trade-off between the fitting error and sensitivity is obtained. The lowest fitting error of the value 23.23 was obtained by NLS, it was for sensitivity 2.02×10^5. The best fitting error obtained by KLM/EPC was 23.59 with sensitivity 7.07×10^3, i.e., with sensitivity $30\times$ lower. The squared Frobenius norm of the original tensor was 196.25. It means that the relative error of NLS was 11.9% for NLS and 12.5% for KLM. Note that the alternating way allows to achieve lower fitting errors than the errors in the former experiment.

B. *Example 2: An Order-3 Tensor*

The order-4 tensor in the previous subsection has the meaning of 48×256 of patches (convolution masks) of the size 5×5. If we decompose the tensor as the order-4 tensor, all these patches would have rank one. Some designers of CNN's may think that the requirement that all patches should have rank one is redundant, and leads to a compromised accuracy of the decomposition. In other words, we can also say that a higher rank is needed to maintain the same accuracy.

Therefore we consider CP decomposition of the Alex-net tensor reshaped to the size $25 \times 48 \times 256$, and try to approximate it by rank 200 and 100. Again, we decompose the tensor by KLM + EPC and NLS + EPC, as in the previous example, and show the performance in the plane sensitivity versus the fitting error. Results are shown in Figure 4. We can see that KLM + EPC provides a decomposition with nearly the same fitting error but nearly ten times smaller sensitivity.

For rank $R = 100$, we were able to reduce the fitting error at sensitivity $s = 10^4$ to the value of 48.15.

VI. CONCLUSIONS

We presented a new concept of CP tensor decomposition, which takes the sensitivity of the decomposition into account. We show that novel algorithm that is KLM, provides estimates with the lower value of sensitivity than e.g. NLS algorithm of Tensorlab. In general, if the goal is to obtain low sensitivity and low fitting error simultaneously, it is advised to combine the CPD algorithms with the error preserving correction (EPC), e.g., the AEPC algorithm. Constrained KLM can still improve the results.

ACKNOWLEDGMENT

The authors would like to thank Dr. N. Vervliet for introducing us in Tensorlab at the Summer School on Tensor Signal Processing, Leuven, 2018, and into techniques which it involves.

REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. Conf. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.

[2] Q. Zhang, L. Tianruo, Y. Zhikui, C. Peng, and K. Li, "An improved deep computation model based on canonical polyadic decomposition," *IEEE Trans. Syst., Man, Cybernet., Syst.*, vol. 55, no. 4, pp. 1657–1666, Oct. 2018.

[3] V. Lebedev, Y. Ganin, M. Rakhuba, I. V. Oseledets, and V. S. Lempitsky, "Speeding-up convolutional neural networks using fine-tuned CP-decomposition," *Comput. Res. Repository*, 2014. [Online]. Available: https://arxiv.org/abs/1412.6553

[4] Y. Ma *et al.*, "A unified approximation framework for deep neural networks," Jul. 2018, *arXiv:1807.10119v2*.

[5] N. Cohen, O. Sharir, and A. Shashua, "On the expressive power of deep learning: A tensor analysis," in *Proc. 29th Annu. Conf. Learn. Theory*, 2016, pp. 698–728

[6] Y. Cheng, D. Wang, P. Zhou, and T. Zhang, "Model compression and acceleration for deep neural networks: The principles, progress, and challenges," *IEEE Signal Process. Mag.*, vol. 35, no. 1, pp. 126–136, Jan. 2018.

[7] M. Jaderberg, A. Vedaldi, and A. Zisserman, "Speeding up convolutional neural networks with low rank expansions," in *Proc. British Mach. Vision Conf.*, Sep. 2014, pp. 1–13.

[8] K. Osawa, A. Sekiya, H. Naganuma, and R. Yokotal, "Accelerating matrix multiplication in deep learning by using low-rank approximation," in *Proc. Int. Conf. High Perform. Comput. Simul.*, 2017, pp. 186–192.

[9] P. Comon, X. Luciani, and A. L. F. De Almeida, "Tensor decompositions, alternating least squares and other tales," *J. Chemometrics, J. Chemometrics Soc.*, vol. 23, pp. 393–405, 2009.

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

[11] L. Sorber, M. Van Barel, and L. De Lathauwer, "Optimization-based algorithms for tensor decompositions: Canonical polyadic decomposition, decomposition in rank(Lr,Lr,1) terms, and a new generalization," *SIAM J. Optim.*, vol. 23, no. 2, pp. 695–720, 2013.

[12] N. Vervliet and L. De Lathauwer, "Numerical optimization based algorithms for data fusion," in *Data Fusion Methodology and Applications*, M. Cocchi, Ed. New York, NY, USA: Elsevier, 2019.

[13] N. Vervliet *et al.*, "Tensorlab 3.0," Mar. 2016. [Online]. Available: https://www.tensorlab.net

[14] A. N. Krylov, "On the numerical solution of equation by which are determined in technical problems the frequencies of small vibrations of material systems" (in Russian), Izvestija AN SSSR (News of Academy of Sciences of the USSR), Otdel. mat. i estest. nauk, VII, No. 4, pp. 491–539, 1931.

[15] J. Liesen and Z. Strakos, *Krylov Subspace Methods: Principles and Analysis*, Oxford, U.K.: Oxford Science, 2012.

[16] A.-H Phan, P. Tichavský, and A. Cichocki, "Low complexity damped gauss-newton algorithms for parallel factor analysis," *SIAM J. Matrix Anal. Appl.*, vol. 34, no. 1, pp. 126–147, Jan. 2013.

[17] P. Tichavský, A. H. Phan, and A. Cichocki, "A further improvement of a fast damped Gauss–Newton algorithm for CANDECOMP-PARAFAC tensor decomposition," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, Vancouver, Canada, May 2013, pp. 5964–5968.

[18] P. Tichavský, A-H. Phan, and A. Cichocki, "Numerical CP decomposition of some difficult tensors," *J. Comput. Appl. Math.*, vol. 317, pp. 362–370, Jun. 2017.

[19] P. Tichavský, A.-H. Phan, and A. Cichocki, "Krylov weighted canonical polyadic tensor decomposition," in *Proc. IEEE ICASSP*, submitted for publication.

[20] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction Method of Multipliers," *Found. Trends Mach. Learn.* vol. 3, no. 1, pp. 1–122, Jan. 2011. [Online]. Available: http://dx.doi.org/10.1561/2200000016

[21] D. M. Dunlavy, T. G. Kolda, and E. Acar, "Poblano v1.0: A Matlab toolbox for gradient-based optimization," , Sandia National Laboratories, Livermore, CA, USA, Tech. Rep. SAND2010-1422, Mar. 2010.

[22] O. Vinyals and D. Povey, "Krylov subspace descent for deep learning," in *Proc. 15th Int. Conf. Artif. Intell. Statist.*, La Palma, Canary Islands, pp. 1261–1268, 2012.

[23] A. P. Liavas, G. Kostoulas, G. Lourakis, K. Huang, and N. D. Sidiropoulos, "Nesterov-based alternating optimization for nonnegative tensor factorization: Algorithm and parallel implementations," *IEEE Trans. Signal Process.*, vol. 66, no. 4, pp. 944–953, Feb. 2018.

[24] H. B. Nielsen, "Damping parameter in Marquardt's method," Tech. Rep. IMM-REP-1999-05, Technical University of Denmark, Lyngby, Denmark, 1999.

[25] J. Nocedal and S. J. Wright, *Numerical Optimization*, Berlin, Germany: Springer 2006.

[26] A. H. Phan, P. Tichavský, and A. Cichocki, "Error preserving correction: A method for CP decomposition at a target error bound," *IEEE Trans. Signal Process.*, vol. 67, no. 5, pp. 1175–1190, Mar. 2019.