Adaptive PARAFAC Decomposition for Third-Order Tensor Completion

Truong Minh-Chinh[1], Viet-Dung Nguyen[2], Nguyen Linh-Trung[1], Karim Abed-Meraim[2]
[1] University of Engineering and Technology, Vietnam National University Hanoi, Vietnam
[2] PRISME Laboratory, University of Orléans, France
tmchinh@hueuni.edu.vn,viet-dung.nguyen@univ-orleans.fr, linhtrung@vnu.edu.vn, karim.abed-meraim@univ-orleans.fr

***Abstract*—This paper proposed a tensor completion algorithm by tracking the Parallel Factor (PARAFAC) decomposition of incomplete third-order tensors with one dimension growing with time. The proposed algorithm first tracks a low-dimensional subspace, then updates the loading matrices of the PARAFAC decomposition. Simulation results showed that the algorithm is reliable and fast, in comparison to the state-of-the-art PARAFAC Weighted OPTimization algorithm.**

I. Introduction

Parallel Factor (PARAFAC) decomposition is a popular tool to analyze and process data represented by a higher-order tensor structure. Drawbacks of the state-of-the art PARAFAC algorithms are high complexity and batch mode operation and, thus, they may not be suitable for applications with streaming data or real time processing constraint. To overcome the complexity drawback when dealing with higher-order tensors of streaming data, Nion *et al.* proposed an *adaptive* algorithm in [1] and applied it in audio processing [2]. Recently, Nguyen *et al.* [3] have developed a faster algorithm for adaptive PARAFAC decomposition, with linear complexity.

Moreover, when the tensor data are also *incomplete*, i.e. the data are only acquired/observed partially, and under the assumption that the vectorized form of each slice of the tensor lives in a low-dimensional subspace, Mardani *et al.* [4] proposed an efficient algorithm for tensor completion that has two stages: (i) track the low-dimensional subspace using the exponentially weighted least-squares criterion regularized by the nuclear norm, and (ii) estimate the loading matrices in PARAFAC decomposition of low-rank incomplete tensor and hence complete the tensor.

Inspired by the algorithm of Mardani *et al.*, we also develop a two-stage algorithm to perform completion of incomplete third-order tensors in this paper. In the first stage, we use the Parallel Estimation and Tracking by REcursive Least Squares (PETRELS) algorithm proposed by Y. Chi *et al.* [5] to track the low-dimensional subspace. Unlike Mardani's algorithm, the cost function in our algorithm is solved by using the second-order stochastic gradient descent method. This algorithm is fast and has advantages in large-scale data [6]. For the second stage, we apply the algorithm proposed by Nion *et al.* [1].

This paper is organized as follows. Section II describe the proposed adaptive PARAFAC model for incomplete streaming data. Section III describes the adaptive PARAFAC decomposition algorithm and the PETRELS algorithm for partial observations which later faciliate the development of the proposed algorithm in Section IV-A. Section IV-B provides the simulation results of proposed algorithm in comparison with the CANDECOMP/PARAFAC Weighted OPTimization algorihtm (CP-WOPT) proposed by Acar *et al.* in [7], which is a batch PARAFAC algorithm.

Notations in use: Bold uppercase (*i.e.*, $\mathbf{X}$), bold lowercase (*i.e.*, $\mathbf{x}$) and bold calligraphic letters (*i.e.*, $\boldsymbol{\mathcal{X}}$) will denote matrices, column vectors, and tensor, respectively. Operators $(\cdot)^T$, $(\cdot)^\dagger$, $\odot$, $*$, $\circ$ denote matrix transposition, matrix pseudo-inverse, Khatri-Rao product, point-wise vector multiplication and outer vector product, respectively.

II. Adaptive PARAFAC Model for Incomplete Streaming Data

A third-order tensor, $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I\times J\times K}$ is called rank-one tensor if it can be written as the outer-product of three vectors as follows:

$$\boldsymbol{\mathcal{X}} = \mathbf{a}\circ\mathbf{b}\circ\mathbf{c}. \tag{1}$$

It means that all elements of $\boldsymbol{\mathcal{X}}$ are defined by $x_{ijk} = a_i b_j c_k$, for all values of the indices.

The PARAFAC decomposition of tensor $\boldsymbol{\mathcal{X}}$ is a decomposition of $\boldsymbol{\mathcal{X}}$ as a sum of a minimal number R of rank-one tensors as

$$\boldsymbol{\mathcal{X}} = \sum_{r=1}^{R} \mathbf{a}_r\circ\mathbf{b}_r\circ\mathbf{c}_r, \tag{2}$$

where R is called the *rank* of $\boldsymbol{\mathcal{X}}$, and matrices $\mathbf{A} = [\mathbf{a}_1,\ldots,\mathbf{a}_R] \in \mathbb{R}^{I\times R}$, $\mathbf{B} = [\mathbf{b}_1,\ldots,\mathbf{b}_R] \in \mathbb{R}^{J\times R}$, $\mathbf{C} = [\mathbf{c}_1,\ldots,\mathbf{c}_R] \in \mathbb{R}^{K\times R}$ are called the loading matrices.

A tensor can be written in matrix form [2], such as $\mathbf{X}^{(1)}$ of size $IK\times J$, whose elements are defined by $\mathbf{X}^{(1)}_{(i-1)K+k,j} = x_{ijk}$. Then, the PARAFAC decomposition in (2) can be expressed in matrix form as

$$\mathbf{X}^{(1)} = (\mathbf{A}\odot\mathbf{C})\,\mathbf{B}^T. \tag{3}$$

In this paper, we consider third-order tensors that have $1 < R < \min(I,J,K)$ so that their PARAFAC decomposition is essentially unique (up to scales and permutation), almost surely [1].

We now build an adaptive PARAFAC model for incomplete streaming data as follows. Consider a third-order tensor $\boldsymbol{\mathcal{X}}(t) \in \mathbb{R}^{I\times J(t)\times K}$, where I and K are constants and $J(t)$ increases with time. At time t, a new slice with partial

978-1-5090-1801-7/16/$31.00 ©2016 IEEE

observation is added to the tensor (see Figure 1 for more details). In the vectorized representation, the new slice is represented as a vector which is seen as a new column of $\mathbf{X}^{(1)}(t)$ in Equation (3). The tensor $\boldsymbol{\mathcal{X}}(t)$ at time t is obtained from the tensor $\boldsymbol{\mathcal{X}}(t-1)$ at time $t-1$ by adding a new slice along dimension t. In other words, we have $J(t) = J(t-1)+1$ and, using the unfolding representation of the tensor, $\mathbf{X}^{(1)}(t)$ is given by

$$\mathbf{X}^{(1)}(t) = \begin{bmatrix} \mathbf{X}^{(1)}(t-1) & \mathbf{x}(t) \end{bmatrix}, \tag{4}$$

where $\mathbf{x}(t)$ is the vectorized representation of the new slice. Next, we will combine the model of adaptive PARAFAC decomposition proposed by [1] and the model of incomplete data in [5] to form our model of adaptive PARAFAC decomposition of incomplete streaming data.

Following the model in (3), we achieve an adaptive PARAFAC decomposition with

$$\mathbf{X}^{(1)}(t-1) = [\mathbf{A}(t-1) \odot \mathbf{C}(t-1)]\, \mathbf{B}^T(t-1) \tag{5a}$$

$$\mathbf{X}^{(1)}(t) = [\mathbf{A}(t) \odot \mathbf{C}(t)]\, \mathbf{B}^T(t). \tag{5b}$$

From the above, the vectorized representation of the new slice $\mathbf{x}(t)$ is given by [1]

$$\mathbf{x}(t) = [\mathbf{A}(t) \odot \mathbf{C}(t)]\, \mathbf{b}^T(t), \tag{6}$$

where $\mathbf{b}^T(t)$ is the t-th column of $\mathbf{B}^T(t)$.

In the situation of partial observations, the new slice at time t of the incomplete data, $\tilde{\mathbf{x}}(t)$, can be modeled as [5]

$$\tilde{\mathbf{x}}(t) = \mathbf{p}(t) * \mathbf{x}(t), \tag{7}$$

where $\mathbf{p}(t)$ is an observation mask vector such that $p_i(t) = 1$ if the i-th entry of $\mathbf{x}(t)$ was observed and $p_i(t) = 0$ if it was not observed.

Given the tensor $\boldsymbol{\mathcal{X}}(t-1)$ at time $t-1$, the partial data $\tilde{\mathbf{x}}(t)$ and the corresponding mask $\mathbf{p}(t)$ at time t, our goal is to estimate $\mathbf{b}^T(t)$, then to update the loading matrices $\mathbf{A}(t)$ and $\mathbf{C}(t)$ and, thus, to recover the full tensor $\boldsymbol{\mathcal{X}}(t)$ at time t.

In the above model for adaptive PARAFAC decomposition of incomplete streaming data, *i.e.* the set of Equations (4), (5), (10) and (7), we use the following two assumptions:

A1: The loading matrices $\mathbf{A}(t)$ and $\mathbf{C}(t)$ change slowly between two successive observations, as in [1].

A2: The set of vectors $\{\mathbf{x}(\tau)\}_{\tau=1}^{t}$ live in a low-dimensional subspace whose rank is upper-bounded by R and changes slowly over time, as in [5].

To facilitate our proposed algorithm in Section IV, we will next briefly describe the adaptive PARAFAC decomposition algorithm in [1] and the PETRELS algorithm in [5] for partial observations.

III. Related Works

A. Adaptive PARAFAC decomposition

An adaptive PARAFAC decomposition algorithm is proposed by Nion *et al.* in [1] for third-order tensors that have one dimension varies with time, *i.e.* $\boldsymbol{\mathcal{X}}(t) \in \mathbb{R}^{I \times J(t) \times K}$, where I and K are constants. The goal of this adaptive PARAFAC algorithm is to estimate the time-varying loading matrices

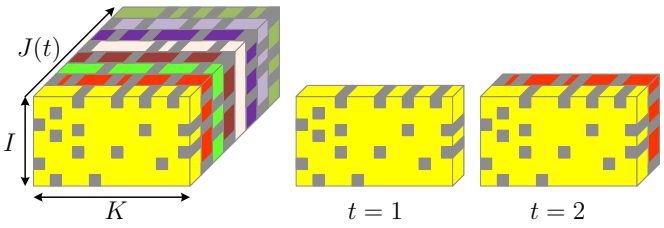

Fig. 1. Third-order tensor with size of $I \times J(t) \times K$. At time t, new slice adding with partial observation.

$\mathbf{A}(t)$, $\mathbf{B}(t)$ and $\mathbf{C}(t)$ at time t based on the past $t-1$ observations.

By setting $\mathbf{H}(t) = \mathbf{A}(t) \odot \mathbf{C}(t)$, the mode-1 unfolding matrix $\mathbf{X}^{(1)}(t)$ in (5b) is rewritten as

$$\mathbf{X}^{(1)}(t) = \mathbf{H}(t)\mathbf{B}^T(t). \tag{8}$$

Under Assumption A1 (in Section II), we can approximate $\mathbf{H}(t)$ by $\mathbf{H}(t-1)$, where $\mathbf{H}(t-1) = \mathbf{A}(t-1) \odot \mathbf{C}(t-1)$. Then, $\mathbf{B}^T(t)$ can be expressed as

$$\mathbf{B}^T(t) \approx \begin{bmatrix} \mathbf{B}^T(t-1) & \mathbf{b}^T(t) \end{bmatrix}. \tag{9}$$

Therefore, according to (10), we have

$$\mathbf{x}(t) = \mathbf{H}(t)\mathbf{b}^T(t). \tag{10}$$

It means that, given the new data slice $\mathbf{x}(t)$ at time t, we only need to estimate $\mathbf{b}^T(t)$, then update $\mathbf{B}^T(t)$, and estimate the other loading matrices $\mathbf{A}(t)$ and $\mathbf{C}(t)$.

B. PETRELS for Partial Observations

The PETRELS algorithm is proposed by Chi *et al.* in [5] to estimate low-dimensional subspaces adaptively from partial observations. PETRELS works under the Assumption A2 (in Section II). At time τ, partial observation $\tilde{\mathbf{x}}_\tau$ can be expressed as

$$\tilde{\mathbf{x}}_\tau = \mathbf{p}(\tau) * \mathbf{x}(\tau) = \mathbf{P}(\tau)\mathbf{x}(\tau), \tag{11}$$

where $\mathbf{x}(\tau)$ is the data vector of length IK to be observed, $\mathbf{p}(\tau) = [p_1(\tau), p_2(\tau), \ldots, p_{IK}(\tau)]^T$ is the observation mask vector as defined in Section II, and $\mathbf{P}(\tau)$ is the masking matrix deduced from $\mathbf{p}(\tau)$ by $\mathbf{P}(\tau) = \operatorname{diag}\{\mathbf{p}(\tau)\}$.

Given the input sequence of incomplete observations via the set of pairs $\{(\tilde{\mathbf{x}}_\tau, \mathbf{p}_\tau)\}_{\tau=1}^{t}$, PETRELS give two outputs for time t: (i) the $IK \times R$ matrix $\mathbf{H}(t)$ which in turn gives the corresponding low-dimensional subspace as the span of the column vectors of $\mathbf{H}(t)$, and the coefficient vector $\mathbf{b}^T(t)$.

At time t, $\mathbf{H}(t)$ is obtained by solving

$$\mathbf{H}(t) = \underset{\mathbf{H} \in \mathbb{R}^{IK \times R}}{\arg\min} \sum_{\tau=1}^{t} \lambda^{t-\tau} f_\tau(\mathbf{H}), \tag{12}$$

where

$$f_\tau(\mathbf{H}) = \min_{\mathbf{b}^T} \left\| \mathbf{P}(\tau) \left(\mathbf{x}(\tau) - \mathbf{H}\mathbf{b}^T(\tau) \right) \right\|_2^2, \tag{13}$$

and λ, with $0 \ll \lambda < 1$, is a forgetting factor.

In (13), $\mathbf{b}^T(\tau)$ is given by

$$\hat{\mathbf{b}}^T(\tau) = \min_{\mathbf{b}^T \in \mathbb{R}^R} \left\| \mathbf{P}(\tau) \left(\tilde{\mathbf{x}}_\tau - \mathbf{H}(\tau-1)\mathbf{b}^T \right) \right\|_2^2 = \left[\mathbf{H}^T(\tau-1)\mathbf{P}(\tau)\mathbf{H}(\tau-1) \right]^\dagger \mathbf{H}^T(\tau-1)\tilde{\mathbf{x}}_\tau(\tau), \quad (14)$$

and $\mathbf{x}(\tau)$ is then estimated as

$$\mathbf{x}(\tau) = \mathbf{H}(\tau-1)\hat{\mathbf{b}}^T(\tau), \quad (15)$$

where $\mathbf{H}(\tau-1)$ has already been obtained at time $t-1$.

Then, $\mathbf{H}(t)$ in (12) is estimated row-wise, that is the m-th row of $\mathbf{H}(t)$ is obtained by solving

$$\mathbf{h}_m(t) = \arg\min_{\mathbf{h}_m} \sum_{i=1}^{t} \lambda^{t-i} p_m(i) \left(x_m(i) - \hat{\mathbf{b}}(i)\mathbf{h}_m \right)^2, \quad (16)$$

with $m = 1, 2, \ldots, IK$.

Setting the derivative of (16) to zero leads to

$$\mathbf{D}_m(t)\mathbf{h}_m(t) = \mathbf{s}_m(t), \quad (17)$$

where

$$\mathbf{D}_m(t) = \sum_{i=1}^{t} \lambda^{t-i} p_m(i) \hat{\mathbf{b}}^T(i)\hat{\mathbf{b}}(i)$$

$$\mathbf{s}_m(t) = \sum_{i=1}^{t} \lambda^{t-i} p_m(i) x_m(i) \hat{\mathbf{b}}^T(i).$$

Hence, $\mathbf{h}_m(t)$ is given by

$$\mathbf{h}_m(t) = \mathbf{D}_m^\dagger(t)\mathbf{s}_m(t), \quad (18)$$

and can be computed adaptively as

$$\mathbf{h}_m(\tau) = \mathbf{h}_m(\tau-1) + p_m(\tau)[x_m(\tau) - \hat{\mathbf{b}}(\tau)\mathbf{h}_m(\tau-1)]\lambda^{-1}\mathbf{v}_m(\tau)\beta_m^{-1}(\tau), \quad (19)$$

where

$$\beta_m(\tau) = 1 + \lambda^{-1}\hat{\mathbf{b}}(\tau)\mathbf{D}_m^\dagger(\tau-1)\hat{\mathbf{b}}^T(\tau),$$
$$\mathbf{v}_m(\tau) = \lambda^{-1}\mathbf{D}_m^\dagger(\tau-1)\hat{\mathbf{b}}^T(\tau),$$
$$\mathbf{D}_m^\dagger(\tau) = \lambda^{-1}\mathbf{D}_m^\dagger(\tau-1) - p_m(\tau)\beta_m^{-1}(\tau)\mathbf{v}_m(\tau)\mathbf{v}_m^T(\tau).$$

IV. Proposed Adaptive PARAFAC for Tensor Completion

A. Proposed Algorithm

We proposed a new adaptive algorithm for third-order tensor completion via adaptive PARAFAC decomposition. Given the estimated loading matrices $\mathbf{A}(t-1)$, $\mathbf{B}(t-1)$ and $\mathbf{C}(t-1)$ at time $t-1$, the new incomplete data slice $\tilde{\mathbf{x}}(t)$ at time t and its corresponding observation mask matrix $\mathbf{P}(t)$, the forgetting factor λ, the proposed algorithm proceeds as follows:

- Estimate the low-dimensional subspace as column subspace of $\mathbf{H}(t)$ and
- Update the loading matrix $\mathbf{B}(t)$ and estimate the loading matrices $\mathbf{A}(t)$ and $\mathbf{C}(t)$ of $\boldsymbol{\mathcal{X}}(t)$.

In detail, the algorithm includes the following three steps:

Step 1 – Estimate $\hat{\mathbf{b}}^T(t)$ and $\mathbf{H}(t)$

To estimate $\hat{\mathbf{b}}^T(t)$ and $\mathbf{H}(t)$, we use PETRELS with the following input parameters: $\mathbf{H}(t-1) = \mathbf{A}(t-1) \odot \mathbf{C}(t-1)$, $\tilde{\mathbf{x}}(t)$, $\mathbf{P}(t)$ and λ.

Step 2 – Extract $\mathbf{A}(t)$ and $\mathbf{C}(t)$ from $\mathbf{H}(t)$

We extract $\mathbf{A}(t)$ and $\mathbf{C}(t)$ from $\mathbf{H}(t)$ using the bi-SVD method similar to the one in [1]:

$$\mathbf{a}_i(t) = \mathbf{H}_i^T(t)\mathbf{c}_i(t-1), \quad (20)$$

$$\mathbf{c}_i(t) = \frac{\mathbf{H}_i(t)\mathbf{a}_i(t)}{\| \mathbf{H}_i(t)\mathbf{a}_i(t) \|}, \quad (21)$$

with $i = 1, \ldots, R$.

Step 3 – Update $\mathbf{B}(t)$ from $\mathbf{B}(t-1)$

The loading matrix $\mathbf{B}(t)$ is updated from $\mathbf{B}(t-1)$ by adding $\hat{\mathbf{b}}^T(t)$ as t-th row of $\mathbf{B}(t-1)$ according to (9).

Finally, to complete the tensor from incomplete observations, we can recover $\mathbf{x}(t)$ using $\mathbf{x}(t) = [\mathbf{A}(t) \odot \mathbf{C}(t)]\,\mathbf{b}^T(t)$. We note, however, that if we try to recover $\mathbf{x}(t)$ at all time instants, it might not be necessary while increasing the computational complexity. Such a situation can be seen in Magnetic Resonance Imaging (MRI) wherein the radiologist may only need to observe the MRI images at some particular times. Therefore, only when needed then $\mathbf{x}(t)$ should be recovered.

B. Experimental Results

In this section, we implement the proposed algorithm and compare its performance with that of the CP-WOPT algorithm in [7]. CP-WOPT is done in Tensor Toolbox [8], in conjunction with Poplano Toolbox [9].

In the simulation, we use a time-varying PARAFAC model which is generated at each time as

$$\mathbf{A}(t) = (1-\varepsilon_A)\mathbf{A}(t-1) + \varepsilon_A \mathbf{N}_A(t), \quad (22)$$

$$\mathbf{C}(t) = (1-\varepsilon_C)\mathbf{C}(t-1) + \varepsilon_C \mathbf{N}_C(t), \quad (23)$$

where $\mathbf{A}(t)$, $\mathbf{N}_A(t)$, $\mathbf{C}(t)$, $\mathbf{N}_C(t)$ are random matrices whose entries follow the standard normal distribution $\mathcal{N}(0,1)$, constants ε_A and ε_C are used to control the variation of $\mathbf{A}(t)$ and $\mathbf{C}(t)$ between two successive observations, and $\mathbf{b}(t)$ is a random vector whose entries follow the $\mathcal{N}(0,1)$. To simulate the partial observation at each time t, we generate the observation mask vector $\mathbf{p}(t)$ at random (with $\rho\%$ of missing entries), and then create the input data $\tilde{\mathbf{x}}(t)$ by

$$\tilde{\mathbf{x}}(t) = \mathbf{p}(t) * ([\mathbf{A}(t) \odot \mathbf{C}(t)]\,\mathbf{b}^T(t)).$$

Other parameters of the proposed algorithm are listed in Table I.

TABLE I
PARTICULAR PARAMETERS SET IN OUR EXPERIMENT

I	K	T	R	ε_A	ε_C	λ	ρ
20	25	500	8	10^{-3}	10^{-3}	0.8	60 %

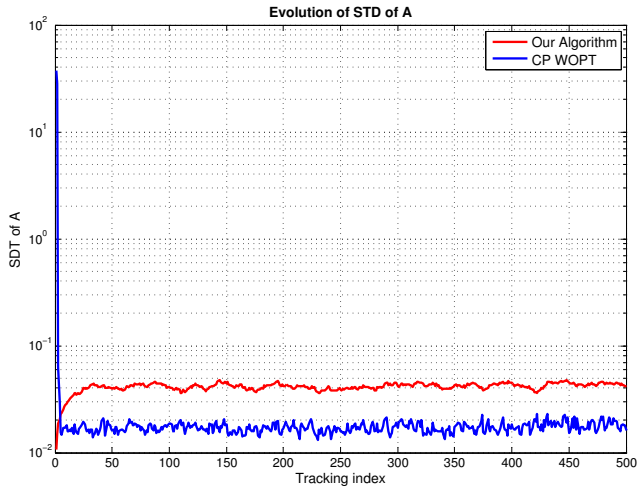

Fig. 2. STD of $\mathbf{A}$, with $R = 8$ and $\rho = 60\%$.

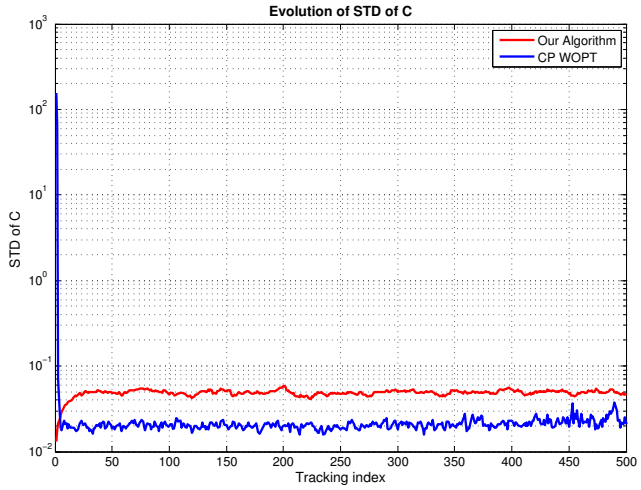

Fig. 3. STD of $\mathbf{C}$, with $R = 8$ and $\rho = 60\%$.

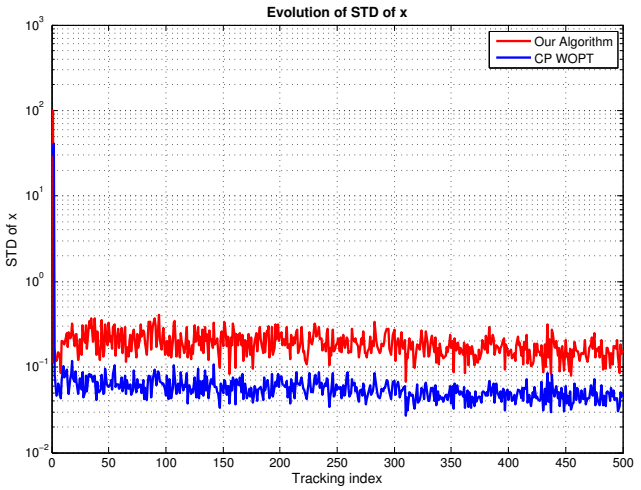

Fig. 4. STD of $\mathbf{x}$, with $R = 8$ and $\rho = 60\%$.

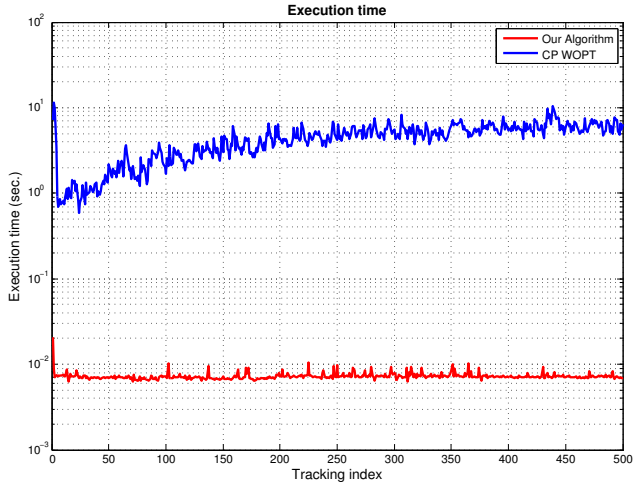

Fig. 5. Execution time of tracking, with $R = 8$ and $\rho = 60\%$.

While CP-WOPT is a batch algorithm, for a fair comparison with the proposed algorithm we use CP-WOPT in an adaptive way such that at time t the input of CP-WOPT is the output of CP-WOPT at time $(t-1)$.

The performance criteria for the estimation of $\mathbf{A}(t)$ and $\mathbf{C}(t)$, are measured by the standard deviation (STD) between the true loading matrices and their estimates $\mathbf{A}_{\text{es}}(t)$ and $\mathbf{C}_{\text{es}}(t)$ up to scale and permutation at each time, as defined as

$$\text{STD}_{\mathbf{A}}(t) = \| \mathbf{A}(t) - \mathbf{A}_{\text{es}}(t) \|_F, \tag{24}$$

$$\text{STD}_{\mathbf{C}}(t) = \| \mathbf{C}(t) - \mathbf{C}_{\text{es}}(t) \|_F . \tag{25}$$

The criterion for $\mathbf{x}(t)$ is defined as

$$\text{STD}_{\mathbf{x}}(t) = \| \mathbf{x}(t) - \mathbf{x}_{\text{es}}(t) \|_F, \tag{26}$$

where $\mathbf{x}_{\text{es}}(t)$ is the estimate of the vectorized representation of the slice $\mathbf{x}(t)$ at time t.

The simulation results give $\text{STD}_{\mathbf{A}}(t)$ and $\text{STD}_{\mathbf{C}}(t)$ of CP-WOPT and the proposed algorithm as shown in Figures 2 and 3, $\text{STD}_{\mathbf{x}}(t)$ as shown in Figure 4, the execution time of tracking as shown in Figure 5.

It is obvious that while our algorithm is reliable, as the standard deviations of $\text{STD}_{\mathbf{A}}(t)$, $\text{STD}_{\mathbf{C}}(t)$ and $\text{STD}_{\mathbf{x}}(t)$ are around 10^{-1}, it has a better execution time than that of CP-WOPT.

V. Conclusions

We have proposed a new algorithm to track the PARAFAC decomposition of third-order tensors adaptively by first estimating the low-dimensional subspaces and then estimating the loading matrices in PARAFAC decomposition. The target subspace is ideally equivalent to the column space of $\mathbf{H}(t)$ which is the Khatri-rao product of $\mathbf{A}(t)$ and $\mathbf{C}(t)$. We know that the estimate of $\mathbf{H}(t)$ is not always in the Khatri-rao product form, and thus, for tensor completion, we can use $\mathbf{H}(t)$, instead of $(\mathbf{A} \odot \mathbf{C})$, as the input matrix for tracking to improve the performance. In some applications, one may only need to estimate the new slice of data but the PARAFAC decomposition, and thus we can use $\mathbf{H}(t)$ directly in the same way. On the other hand, we can exploit the Khatri-rao product form of $\mathbf{H}(t)$ to reduce the computational complexity of the proposed algortihm.

ACKNOWLEDGMENT

This research is funded by Vietnam National Foundation for Science and Technology Development (NAFOSTED) under grant number 102.02-2015.32.

REFERENCES

[1] D. Nion and N. D. Sidiropoulos, "Adaptive algorithms to track the PARAFAC decomposition of a third-order tensor," *IEEE Transactions on Signal Processing*, vol. 57, no. 6, pp. 2299–2310, 2009.

[2] D. Nion, K. N. Mokios, N. D. Sidiropoulos, and A. Potamianos, "Batch and adaptive PARAFAC-based blind separation of convolutive speech mixtures," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 6, pp. 1193–1207, 2010.

[3] V.-D. Nguyen, K. Abed-Meraim, and N. Linh-Trung, "Fast adaptive PARAFAC decomposition algorithm with linear complexity," in *41th IEEE Internaional Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. IEEE, 2016.

[4] M. Mardani, G. Mateos, and G. B. Giannakis, "Subspace learning and imputation for streaming big data matrices and tensors," *IEEE Transactions on Signal Processing*, vol. 63, no. 10, pp. 2663–2677, 2015.

[5] Y. Chi, Y. C. Eldar, and R. Calderbank, "Petrels: Parallel subspace estimation and tracking by recursive least squares from partial observations," *IEEE Transactions on Signal Processing*, vol. 61, no. 23, pp. 5947–5959, 2013.

[6] L. Bottou, "Large-scale machine learning with stochastic gradient descent," in *Proceedings of COMPSTAT'2010*. Springer, 2010, pp. 177–186.

[7] E. Acar, D. M. Dunlavy, T. G. Kolda, and M. Mørup, "Scalable tensor factorizations for incomplete data," *Chemometrics and Intelligent Laboratory Systems*, vol. 106, no. 1, pp. 41–56, 2011.

[8] B. Bader and T. Kolda, "MATLAB tensor toolbox version 2.4: http://csmr. ca. sandia. gov/˜ tgkolda," 2010.

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