Robust Tensor Tracking with Missing Data and Outliers: Novel Adaptive CP Decomposition and Convergence Analysis

Le Trung Thanh, Karim Abed-Meraim, *Fellow, IEEE*, Nguyen Linh Trung, *Senior Member, IEEE*, and Adel Hafiane, *Member, IEEE*.

***Abstract*—Canonical Polyadic (CP) decomposition is a powerful multilinear algebra tool for analyzing multiway (a.k.a. tensor) data and has been used for various signal processing and machine learning applications. When the underlying tensor is derived from data streams, adaptive CP decomposition is required. In this paper, we propose a novel method called robust adaptive CP decomposition (RACP) for dealing with high-order incomplete streaming tensors that are corrupted by outliers. At each time instant, RACP first performs online outlier rejection to accurately detect and remove sparse outliers, and then performs tensor factor tracking to efficiently update the tensor basis. A unified convergence analysis of RACP is also established in that the sequence of generated solutions converges asymptotically to a stationary point of the objective function. Extensive experiments were conducted on both synthetic and real data to demonstrate the effectiveness of RACP in comparison with state-of-the-art adaptive CP algorithms.**

***Index Terms*—CANDECOMP/PARAFAC (CP) decomposition, adaptive algorithm, streaming tensor, missing data, outlier.**

I. INTRODUCTION

Nowadays, many modern datasets can be represented by multiway arrays which are referred to as *tensors*, e.g., a color video surveillance sequence can be represented as a 4^{th}-order tensor of dimensionality, width × height × channel × time. Accordingly, tensor decomposition, which factorizes a tensor into a sequence of basic components, has become a popular analysis tool for processing high-dimensional and multivariate data [1]–[4].

CANDECOMP/PARAFAC (CP) decomposition and Tucker decomposition are well-known and widely-used types of tensor decomposition [1], [5] . Under CP decomposition, a tensor can be expressed as a linear combination of rank-1 tensors, which are formed by an outer product of vectors. As a result, this decomposition offers several nice properties [1], [4]. Among them are the following: (i) CP only requires a linear space complexity w.r.t. the tensor order; (ii) hence, it can avoid the "curse of dimensionality", a phenomenon whereby the memory storage grows drastically when the dimension increases; (iii) under certain conditions, its expression is essentially unique up to a permutation and scale. The merits of CP decomposition have already been demonstrated in various applications, such as wireless communications [6]–[8], neuroscience [9]–[11], and remote sensing [12]–[14].

In recent years, the demand for adaptive (i.e., online) processing has been increasing due to the fact that many applications generate a huge number of data streams over time [15]–[17]. Such data streams are often with high *veracity* and high *velocity*. Veracity requires robust algorithms so as to deal with uncertain, noisy and imperfect data, while velocity requires online real-time processing [15]. These characteristics lead to several critical computational issues: (i) increase in size of the data streams over time, (ii) time-dependent and varying models, and (iii) uncertainty and incompleteness. A robust variant of tensor decomposition for tensors derived from such data streams, namely *robust tensor tracking* (RTT), has been emerging as a good approach. The main goal of this paper is to propose a scalable and effective method for RTT under the CP model.

A. Related Work

Many methods for CP decomposition have been proposed, and standard algorithms and their applications have been nicely surveyed in [1], [2], [5], [18]. However, most CP algorithms are either sensitive to data imperfection or designed only for batch processing. Online or adaptive algorithms are needed when tensors are derived from data streams. The very first adaptive CP algorithms were developed by Nion and Sidiroppoulos in [19] more than a decade ago. Specifically, the authors proposed to track the low-dimensional subspace of the underlying streaming tensor and then reconstruct the loading factors by exploiting its Khatri-Rao structure. Since then, many adaptive CP algorithms have been proposed for factorizing tensors derived from data streams, such as [20]–[23], to name a few. Vandecappelle *et al.* in [20] developed a nonlinear least-squares (NLS)-based adaptive CP algorithm for factorizing streaming tensors of order 3. In [21], Zhou *et al.* introduced the so-called OLCP algorithm which is capable of tracking higher-order streaming tensors. Smith *et al.* in [22] proposed an adaptive algorithm specifically for streaming sparse tensors, namely CP-stream. In [23], Rambhatla *et al.* introduced another adaptive CP algorithm called TensorNOODL using online dictionary learning. Nevertheless, none of them are designed for dealing with missing data and outliers.

Le Trung Thanh is with the University of Orléans, INSA CVL, PRISME, France, and AVITECH, University of Engineering and Technology, Vietnam National University, Hanoi, Vietnam. Email: trung-thanh.le@univ-orleans.fr.

Karim Abed-Meraim and Adel Hafiane are with the University of Orléans, INSA CVL, PRISME, France. Karim Abed-Meraim is also a member of IUF (Institut Universitaire de France). Email: karim.abed-meraim@univ-orleans.fr, adel.hafiane@insa-cvl.fr.

Nguyen Linh Trung (corresponding author) is with AVITECH, University of Engineering and Technology, Vietnam National University, Hanoi, Vietnam. E-mail: linhtrung@vnu.edu.vn.

This paper has supplementary downloadable material available at http://ieeexplore.ieee.org, provided by the authors. The material includes derivations of several results presented in the paper. This material is 330KB in size.

In the context of missing data, there exist some online CP algorithms that are able to deal with incomplete streaming tensors. Morteza *et al.* developed an online stochastic CP algorithm for RTT of third-order tensors, called TeCPSGD [24]. Thanks to the stochastic gradient descent method, the algorithm can efficiently update the loading factors. Kasai proposed an effective recursive estimator, OLSTEC, to learn low-rank components of the underlying data streams [25]. It yields a better estimation accuracy than TeCPSGD, but its complexity is much higher. Both TeCPSGD and OLSTEC are, however, not capable of tracking higher-order streaming tensors. To overcome this drawback, we recently proposed a new adaptive algorithm that is able to handle higher-order incomplete streaming tensors, called ACP [26], [27]. In spite of their computational merits, the above algorithms are sensitive to outliers.

To deal with outliers, Zhang *et al.* introduced an online Bayesian-based CP algorithm, namely BRST [28]. To capture sparse components or outliers affecting the tensor, BRST uses a Bayesian statistical model. However, BRST has high computational complexity and hence proves to be inefficient when dealing with fast-arriving and big data streams. Najafi *et al.* proposed another robust estimator for adaptive CP decomposition, called OR-MSTC [29]. Leveraging the alternating direction method of multipliers (ADMM), OR-MSTC is capable of handling gross corruptions in multi-aspect streaming tensor data. Lee *et al.* developed the so-called SOFIA method which is specifically designed for dealing with seasonal tensor streams with missing values and sparse corruptions [30]. SOFIA employs the Holt-Winters procedure, a well-known forecasting model for time series capable of dealing with trend and seasonality [31]. Convergence analysis of OR-MSTC and SOFIA is, however, not available.

Some other studies attempted to extend online robust PCA and subspace learning for high-order tensor data. Hu *et al.* proposed an incremental tensor subspace learning algorithm, called IRTSA, and applied it to robust visual tracking in video streams [32]. Li *et al.* presented a robust algorithm that can update the tensor dictionary and detect anomalies in an online manner, namely RTSL [33]. Sobral *et al.* introduced an online stochastic tensor algorithm for learning low-rank structure and sparse components in the tensor data [34]. Another incremental tensor decomposition was designed for video background and foreground separation in [35]. Li *et al.* developed an adaptive algorithm for robust low-rank tensor learning, called ORLTM [36]. Very recently, Dimitris *et al.* proposed the first robust online Tucker decomposition that can deal with streaming tensors in the presence of outliers [37]. However, none of the above algorithms are designed for handling missing data. The problem of robust tensor tracking for high-order incomplete streaming tensors remains largely unexplored.

B. Main Contributions

Since there exist several robustification methods for batch tensor decompositions with performance guarantees (see, for examples, [38]–[41]), we designed our algorithm in such a way that it casts such robustness guarantees on RTT. Our method involves the two well-known optimization frameworks: block-coordinate descent (BCD) [42] and majorization-minimization (MM) [43], [44]. To adapt to online learning, the iteration step of MM coincides with the arrival of a new tensor slice over time. Specifically, at each time instant, we decompose RACP into two stages: (i) online outlier rejection and (ii) tensor factor tracking. In the first stage, sparse outliers living in the underlying data streams are first detected by optimizing an ℓ_1-norm regularized loss function. Since the proposed loss function not only promotes sparsity but also remains convex, its convergence is guaranteed. Next, based on the past estimates, the second stage enables us to update the tensor basis by minimizing a majorizing surrogate of the main objective function. Accordingly, an efficient recursive estimator is developed to update the loading factors as well as to track their variation over time.

Our main contributions are summarized as follows. First, we propose a scalable and effective online CP algorithm with ability to (i) estimate low-rank components of streaming tensors derived from imperfect and noisy data streams due to missing observations and outlier corruptions, (ii) adapt the changes of the underlying data streams in dynamic and nonstationary environments, (iii) separate and reject sparse outliers in an online fashion with high accuracy, (iv) be capable of tracking tensor components derived from large data streams, and (v) easily incorporate prior information to deal with specific constraints on the tensor model, e.g., smoothness and nonnegativity.

Secondly, we show that RACP is a *provable* adaptive CP algorithm with a convergence guarantee. Under mild conditions, we prove that the sequence of solutions generated by RACP converges asymptotically to a stationary point of the empirical loss function. Moreover, the asymptotic variation of the solutions and the almost-sure convergence of the objective function values are also analyzed. To the best of our knowledge, this is a pioneer convergence analysis for RTT algorithms in the presence of missing data and outliers.

Finally, we provide several experiments on both synthetic and real data to illustrate the effectiveness of RACP and its variant in comparison with state-of-the-art algorithms.

Compared to our companion work on tensor tracking in [27], there are several differences between ACP and RACP. First, ACP is not designed for handling sparse corruptions, thus its tracking ability diminishes considerably when observations are corrupted by outliers. By contrast, RACP which is a robust version of ACP is capable of tracking the underlying tensor model as well as detecting sparse corruptions successfully over time.

Technically, the data model and the objective function considered in Section II are different from that in [27] due to the presence of sparse outliers. A ℓ_1-norm regularization term and a truncated sliding window are particularly introduced in this work, which leads to several different technical specifications in optimization methodology and convergence analysis. More concretely, we here derive an ADMM solver to estimate the tensor dictionary coefficients and sparse outliers while ACP adopts a randomized least-squares method for this task.

Next, we propose to use a truncated window of a flexible size L_t varying from 1 to t, instead of using an exponential one as in [27]. In the adaptive signal processing literature, it is well known that the exponential window is only useful for stationary and slowly time-varying environments where the underlying low-rank model is either static or changes slowly with time [45]. To enhance the tracking ability of RACP in more complicated scenarios (e.g., fast time-varying or abrupt changes at some points), the use of a truncated window is preferable. Accordingly, a more elaborate recursive rule for updating each tensor factor is designed up to row-wise level which can further support parallel and distributed processing and implementations. It also helps accelerate the tracking process, e.g., we can ignore or skip the update of some factor rows without affecting the others if the corresponding observations are seriously disrupted by strong corruptions.

C. Paper Organization & Notations

The rest of this paper is organized as follows. Section II formulates the RTT problem of interest. Section III presents the proposed RACP algorithm and its convergence analysis is established in Section IV. Section V provides experiments to evaluate the performance of RACP. Section VI concludes the paper. For clarity, the frequently used notations are summarized in Table I.

TABLE I: Notational conventions.

$x, \mathbf{x}, \mathbf{X}, \boldsymbol{\mathcal{X}}$	scalar, vector, matrix, and tensor
$x_{i_1\dots i_N}$ or $[\boldsymbol{\mathcal{X}}]_{i_1\dots i_N}$	$(i_1,\dots,i_N)$-th entry of $\boldsymbol{\mathcal{X}}$
$\mathbf{x} = \mathrm{vec}(\mathbf{X})$	vectorization of $\mathbf{X}$
$\mathbf{X} = \mathrm{diag}(\mathbf{x})$	diagonal matrix $\mathbf{X}$ with $\mathbf{x}$ on the main diagonal
$\mathbf{X}(i,:), \mathbf{X}(:,j)$	i-th row and j-th column of $\mathbf{X}$
$\mathbf{X}^\top$, $\mathbf{X}^{-1}$ $\mathbf{X}^{\#}$	transpose, inverse, and pseudo-inverse of $\mathbf{X}$
$\underline{\mathbf{X}}^{(n)}$	mode-n unfolding of $\boldsymbol{\mathcal{X}}$
$\mathbf{U}^{(n)}$	n-th loading factor/matrix
$\circ$, $\odot$, $\circledast$	outer, Khatri-Rao, and Hadamard product
$\boldsymbol{\mathcal{X}} \boxplus_n \boldsymbol{\mathcal{Y}}$	concatenation of $\boldsymbol{\mathcal{X}}$ with $\boldsymbol{\mathcal{Y}}$ along the dimension n
$\boldsymbol{\mathcal{X}} \times_n \mathbf{U}$	n-mode product of $\boldsymbol{\mathcal{X}}$ with $\mathbf{U}$,
$\boldsymbol{\mathcal{X}} \prod_{n=1}^N \times_n \mathbf{U}^{(n)}$	$\boldsymbol{\mathcal{X}} \times_1 \mathbf{U}^{(1)} \times_2 \cdots \times_N \mathbf{U}^{(N)}$
$\bigodot_{n=1}^N \mathbf{U}^{(n)}$	$\mathbf{U}^{(N)} \odot \mathbf{U}^{(N-1)} \odot \cdots \odot \mathbf{U}^{(1)}$
$\lVert . \rVert$	Euclidean norm

II. Problem Statement

In this study, we consider an incomplete streaming tensor $\boldsymbol{\mathcal{X}}[t] \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N \times t}$ whose slices are serially observed with time. At each time t, $\boldsymbol{\mathcal{X}}[t]$ is particularly obtained by concatenating a new incoming "slice" $\boldsymbol{\mathcal{X}}_t \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N \times 1}$ into the previous $\boldsymbol{\mathcal{X}}[t-1]$ along the time dimension, i.e., $\boldsymbol{\mathcal{X}}[t] = \boldsymbol{\mathcal{X}}[t-1] \boxplus_{N+1} \boldsymbol{\mathcal{X}}_t$. In particular, we assume to observe the tensor slice $\boldsymbol{\mathcal{X}}_t$ satisfying the following model:

$$\boldsymbol{\mathcal{P}}_t \circledast \boldsymbol{\mathcal{X}}_t = \boldsymbol{\mathcal{P}}_t \circledast \big(\boldsymbol{\mathcal{Y}}_t + \boldsymbol{\mathcal{O}}_t + \boldsymbol{\mathcal{N}}_t\big), \tag{1}$$

where $\boldsymbol{\mathcal{P}}_t$ is a binary mask tensor, $\boldsymbol{\mathcal{Y}}_t$ is a low-rank tensor, $\boldsymbol{\mathcal{O}}_t$ is a sparse tensor containing outliers, $\boldsymbol{\mathcal{N}}_t$ is a Gaussian noise tensor, and all these tensors are of the same size with $\boldsymbol{\mathcal{X}}_t$. Specifically, the observation mask $\boldsymbol{\mathcal{P}}_t$ indicates whether the $(i_1, i_2, \dots, i_N)$-th entry of $\boldsymbol{\mathcal{X}}_t$ is observed or missing, i.e.,

$$p_{i_1 i_2 \dots i_N} = \begin{cases} 0, & \text{if } x_{i_1 i_2 \dots i_N} \text{ is missing,} \\ 1, & \text{otherwise.} \end{cases}$$

The low-rank tensor $\boldsymbol{\mathcal{Y}}_t$ is generated according to the following model[1]:

$$\boldsymbol{\mathcal{Y}}_t = \left(\boldsymbol{\mathcal{I}} \prod_{n=1}^{N} \times_n \mathbf{U}^{(n)}\right) \times_{N+1} \mathbf{u}_t^\top, \tag{2}$$

where $\boldsymbol{\mathcal{I}} \in \mathbb{R}^{r \times r \times \dots \times r}$ is an identity tensor, $\mathbf{u}_t \in \mathbb{R}^{r \times 1}$ is a weight vector[2] and $\{\mathbf{U}^{(n)}\}_{n=1}^N$, with $\mathbf{U}^{(n)} \in \mathcal{U}_n \subseteq \mathbb{R}^{I_n \times r}$, are loading factors. For short, we write $\mathcal{D} := \mathcal{U}_1 \times \mathcal{U}_2 \times \dots \times \mathcal{U}_N$ and denote $\mathbf{D} = [(\mathbf{U}^{(1)})^\top, (\mathbf{U}^{(2)})^\top, \dots, (\mathbf{U}^{(N)})^\top]^\top$ the tensor dictionary containing all loading factors.

Next, we define a loss function $\ell(\cdot)$ that not only promotes sparsity but also preserves convexity. For a fixed $\mathbf{D}$ and a tensor slice $\boldsymbol{\mathcal{X}}$ under a binary observation mask $\boldsymbol{\mathcal{P}}$, the loss function w.r.t. $\mathbf{D}$ and $\{\boldsymbol{\mathcal{P}}, \boldsymbol{\mathcal{X}}\}$ is defined as

$$\ell\big(\mathbf{D}, \boldsymbol{\mathcal{P}}, \boldsymbol{\mathcal{X}}\big) = \min_{\mathbf{u}, \boldsymbol{\mathcal{O}}} \tilde{\ell}\big(\mathbf{D}, \boldsymbol{\mathcal{P}}, \boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{O}}, \mathbf{u}\big), \text{ with} \tag{3}$$

$$\tilde{\ell}\big(\mathbf{D}, \boldsymbol{\mathcal{P}}, \boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{O}}, \mathbf{u}\big) = \big\lVert \boldsymbol{\mathcal{O}} \big\rVert_1 + \frac{\rho}{2} \Big\lVert \boldsymbol{\mathcal{P}} \circledast \big(\boldsymbol{\mathcal{X}} - \boldsymbol{\mathcal{O}} - \boldsymbol{\mathcal{H}} \times_{N+1} \mathbf{u}^\top\big) \Big\rVert_F^2,$$

where $\boldsymbol{\mathcal{H}} = \boldsymbol{\mathcal{I}} \prod_{n=1}^N \times_n \mathbf{U}^{(n)}$. The ℓ_1-norm is to promote the sparsity on $\boldsymbol{\mathcal{O}}$ and $\rho > 0$ is a regularized parameter.

Now, given a streaming set of incomplete tensor slices $\{\boldsymbol{\mathcal{P}}_k \circledast \boldsymbol{\mathcal{X}}_k\}_{k=1}^t$, robust tensor tracking (RTT) can be stated as the following optimization problem:

$$\mathbf{D}_t = \underset{\mathbf{D}}{\mathrm{argmin}} \left[f_t(\mathbf{D}) = \frac{1}{L_t} \sum_{k=t-L_t+1}^{t} \lambda^{t-k} \ell\big(\mathbf{D}, \boldsymbol{\mathcal{P}}_k, \boldsymbol{\mathcal{X}}_k\big) \right], \tag{4}$$

where L_t is the length of a sliding window and λ is a forgetting factor. When $L_t = t, \lambda = 1$, the minimization of (4) boils down to its counterpart in batch setting. When $0 < L_t < t$ or $\lambda < 1$, it reduces the impact of past observations, and hence facilitates the tracking ability of RTT estimators in time-varying conditions.

We make some assumptions to support the proposed algorithm in Section III. First, entries of tensor slices $\{\boldsymbol{\mathcal{X}}_t\}_{t\geq 1}$ are Frobenius-norm bounded, i.e., $\lVert \boldsymbol{\mathcal{X}}_t \rVert_F \leq M_x < \infty$ $\forall t$. This prevents arbitrarily large values in observations and ill-conditioned computation. Next, the tensor rank r is assumed to remain unchanged over time. In addition, tensor factors $\{\mathbf{U}_t^{(n)}\}_{n=1}^N$ are bounded and full column rank, i.e., $\mathrm{rank}(\bar{\mathbf{U}}_t^{(n)}) = r < I_n$ and $\lVert \bar{\mathbf{U}}_t^{(n)} \rVert_F \leq \kappa_U < \infty$ $\forall n$. Besides, the variation between two consecutive time instants is small, i.e., $0 \leq \sin\theta(\mathbf{U}_t^{(n)}, \mathbf{U}_{t-1}^{(n)}) \ll 1$ $\forall n, t$, where $\theta(\mathbf{U}_t^{(n)}, \mathbf{U}_{t-1}^{(n)})$ denotes the canonical angle (the largest principal angle) between two subspaces spanning $\mathbf{U}_t^{(n)}$ and $\mathbf{U}_{t-1}^{(n)}$, respectively. This assumption permits the estimation of the outliers and the coefficient vector from the previous estimation with reasonable accuracy. Under these assumptions, our optimization algorithm is capable of accurately estimating tensor factors, but also successfully tracking their variation over time.

[1]Since the CP format can be viewed as a special case of the Tucker format, we have the following equivalence:

$$\boldsymbol{\mathcal{I}} \prod_{n=1}^N \times_n \mathbf{U}^{(n)} \equiv \sum_{i=1}^r \mathbf{U}^{(1)}(:,i) \circ \mathbf{U}^{(2)}(:,i) \circ \dots \circ \mathbf{U}^{(N)}(:,i),$$

where $\mathbf{U}^{(n)}(:,i)$ is the i-th column of $\mathbf{U}^{(n)} \in \mathbb{R}^{I_n \times r}$.

[2]In batch setting, the weight vector $\mathbf{u}_t$ in (2) is seen as the t-th row of the last loading factor $\mathbf{U}^{(N+1)} \in \mathbb{R}^{I_{N+1}[t] \times r}$ of the underlying tensor $\boldsymbol{\mathcal{X}}[t]$.

III. Proposed Methods

In this section, we first propose the robust adaptive CP (RACP) algorithm for the RTT problem in the presence of missing data and outliers. Then, we introduce two simple extensions of RACP in order to deal with the smoothness condition and nonnegative constraints.

A. Proposed RACP Algorithm

Finding the global optimal solution of (4) is difficult since $f_t(\cdot)$ is nonconvex. We here adapt it using the majorization-minimization (MM) framework [43], which has been successfully applied to several signal processing problems in general [44] and online learning problems in particular [46]–[49]. In essence, we decompose it into two main stages: (i) online outlier rejection and (ii) tensor factor tracking.

On the arrival of $\boldsymbol{\mathcal{X}}_t$ at each time t, we first estimate the outlier tensor $\boldsymbol{\mathcal{O}}_t$ and the coefficient vector $\mathbf{u}_t$ based on the old estimation $\mathbf{D}_{t-1}$. Specifically, we solve the following optimization:

$$\{\boldsymbol{\mathcal{O}}_t, \mathbf{u}_t\} = \underset{\boldsymbol{\mathcal{O}},\mathbf{u}}{\operatorname{argmin}}\ \tilde{\ell}\big(\mathbf{D}_{t-1}, \boldsymbol{\mathcal{P}}_t, \boldsymbol{\mathcal{X}}_t, \boldsymbol{\mathcal{O}}, \mathbf{u}\big). \quad (5)$$

From the past statistics $\{\mathbf{D}_k, \boldsymbol{\mathcal{P}}_k, \boldsymbol{\mathcal{X}}_k, \boldsymbol{\mathcal{O}}_k, \mathbf{u}_k\}_{k\geq 1}$, the set of loading factors $\mathbf{D}_t = \{\mathbf{U}_t^{(n)}\}_{n=1}^N$ can be updated by minimizing the following majorizing surrogate $\tilde{f}_t(\cdot)$:

$$\tilde{f}_t(\mathbf{D}) = \frac{1}{L_t} \sum_{k=t-L_t+1}^{t} \lambda^{t-k} \tilde{\ell}\big(\mathbf{D}, \boldsymbol{\mathcal{P}}_k, \boldsymbol{\mathcal{X}}_k, \boldsymbol{\mathcal{O}}_k, \mathbf{u}_k\big), \quad (6)$$

that locally approximates $f_t(\cdot)$. Note that $\tilde{f}_t(\mathbf{D})$ is not only first-order surrogate, but also a majorant function of $f_t(\mathbf{D})$, that is, for all t and $\mathbf{D}$, we always have $f_t(\mathbf{D}) \leq \tilde{f}_t(\mathbf{D})$ and the error function $e_t(\mathbf{D}) = \tilde{f}_t(\mathbf{D}) - f_t(\mathbf{D})$ is Lipschitz continuous. In fact, $\tilde{f}_t(\mathbf{D})$ and $f_t(\mathbf{D})$ converge almost-surely to the same limit, and the solution $\mathbf{D}_t$, which minimizes $\tilde{f}_t(\mathbf{D})$, is exactly that of $f_t(\mathbf{D})$ when $t \to \infty$. The results will be later proven in our convergence analysis.

In what follows, we propose two solvers for minimizing (5) and (6) efficiently.

Stage 1: Online Outlier Rejection

To estimate $\boldsymbol{\mathcal{O}}_t$ and $\mathbf{u}_t$, we recast (5) into the following standard matrix-vector form:

$$\{\mathbf{o}_t, \mathbf{u}_t\} = \underset{\mathbf{o},\mathbf{u}}{\operatorname{argmin}}\ \|\mathbf{o}\|_1 + \frac{\rho}{2}\Big\|\mathbf{P}_t\big(\mathbf{x}_t - \mathbf{o} - \mathbf{H}_{t-1}\mathbf{u}\big)\Big\|_2^2, \quad (7)$$

where $\mathbf{o}_t = \operatorname{vec}(\boldsymbol{\mathcal{O}}_t)$, $\mathbf{x}_t = \operatorname{vec}(\boldsymbol{\mathcal{X}}_t)$, the observation mask matrix $\mathbf{P}_t = \operatorname{diag}(\operatorname{vec}(\boldsymbol{\mathcal{P}}_t))$, and $\mathbf{H}_{t-1}$ is of a Khatri-Rao structure, i.e., $\mathbf{H}_{t-1} = \bigodot_{n=1}^N \mathbf{U}_{t-1}^{(n)}$.

Since both terms of (7) are convex, it can be efficiently solved by several methods with convergence guarantees. Here, we use an ADMM solver to minimize (7) due to its simple interpretation and moderate convergence rate [50]. At the i-th iteration, we particularly read

$$\mathbf{u}^i = \big(\mathbf{H}_{t-1}^\top \mathbf{P}_t \mathbf{H}_{t-1}\big)^{\#} \mathbf{H}_{t-1}^\top \mathbf{P}_t \big(\mathbf{x}_t - \mathbf{o}^{i-1} - \mathbf{z}^{i-1}/\rho\big), \quad (8)$$

$$\mathbf{r}^i = \alpha \mathbf{P}_t\big(\mathbf{x}_t - \mathbf{H}_{t-1}\mathbf{u}^i\big) + (1-\alpha)\mathbf{o}^{i-1} \quad (9)$$

$$\mathbf{o}^i = \mathcal{S}_{1/\rho}\big(\mathbf{r}^i - \mathbf{z}^{i-1}/\rho\big), \quad (10)$$

$$\mathbf{z}^i = \mathbf{z}^{i-1} + \rho\big(\mathbf{o}^i - \mathbf{r}^i\big), \quad (11)$$

where $\mathcal{S}(\cdot)$ is the soft-thresholding operator of the ℓ_1-norm defined as $\mathcal{S}_\varepsilon(x) = \max(0, x-\varepsilon) - \max(0, -x-\varepsilon)$ and $\alpha \in [1.5, 1.8]$ is a relaxation parameter. The procedure is stopped when residuals are small, i.e., $\|\mathbf{P}_t(\mathbf{x}_t - \mathbf{H}_{t-1}\mathbf{u}^i - \mathbf{o}^i)\|_2 \leq \epsilon^{res}$ and $\|\mathbf{o}^i - \mathbf{r}^i\|_2 \leq \epsilon^{out}$ where $\epsilon^{res}, \epsilon^{out} > 0$ are predefined accuracy parameters or when the procedure reaches the maximum number of iterations.

After the sparse outlier $\boldsymbol{\mathcal{O}}_t$ is detected, we reduce the effect of $\boldsymbol{\mathcal{O}}_t$ on the tracking process by the following outlier removal

$$\boldsymbol{\mathcal{P}}_t \circledast \widehat{\boldsymbol{\mathcal{X}}}_t = \boldsymbol{\mathcal{P}}_t \circledast (\boldsymbol{\mathcal{X}}_t - \boldsymbol{\mathcal{O}}_t). \quad (12)$$

In some cases, we can skip the corrupted entries in $\boldsymbol{\mathcal{X}}_t$ by re-updating the observation mask $\boldsymbol{\mathcal{P}}_t$ as

$$p_{i_1 i_2 \dots i_N} = \begin{cases} 0, & \text{if } x_{i_1 \dots i_N} \text{ is missing or outlier,} \\ 1, & \text{otherwise.} \end{cases} \quad (13)$$

Here, the removal step (12) still holds under the new binary mask $\boldsymbol{\mathcal{P}}_t$. This approach stems from the following observations. In the context of subspace tracking (ST), rejecting outliers can facilitate the tracking ability of ST estimators since only "clean" measurements involve the process [49]. Our next stage for estimating the tensor basis can indeed boil down to the ST problem with missing data, so the outlier rejection mechanism of (13) can improve performance. Please see Fig. 7 for an illustration that the outlier rejection mechanism can help improve the convergence rate of RACP when the fraction of corrupted entries is not too large. We refer to the mechanism (13) as a heuristic modification of the standard outlier removal (12) in RACP.

Stage 2: Estimation of factors $\{\mathbf{U}_t^{(n)}\}_{n=1}^N$

The optimization (6) can be effectively solved by using the block-coordinate descent (BCD) technique. The main idea is to minimize alternately the surrogate $\tilde{f}_t(\cdot)$ w.r.t. each factor $\mathbf{U}_t^{(n)}$ while fixing the remaining factors (hereafter denoted as $\tilde{f}_t(\mathbf{U}_t^{(n)}, .)$ for short), that is,

$$\mathbf{U}_t^{(n)} = \underset{\mathbf{U}^{(n)}}{\operatorname{argmin}}\, \tilde{f}_t\big(\mathbf{U}^{(n)}, .\big). \quad (15)$$

Minimization (15) is equivalent to

$$\underset{\mathbf{U}^{(n)}}{\operatorname{argmin}} \sum_{k=t-L_t+1}^{t} \lambda^{t-k} \Big\| \underline{\mathbf{P}}_k^{(n)} \circledast \Big(\widehat{\underline{\mathbf{X}}}_k^{(n)} - \mathbf{U}^{(n)}\big(\mathbf{W}_k^{(n)}\big)^\top\Big)\Big\|_F^2, \quad (16)$$

where $\widehat{\underline{\mathbf{X}}}_k^{(n)}$ and $\underline{\mathbf{P}}_k^{(n)}$ are the mode-n unfoldings of $\widehat{\boldsymbol{\mathcal{X}}}_k$ and $\boldsymbol{\mathcal{P}}_k$ respectively, and $\mathbf{W}_k^{(n)}$ is given by[3]

$$\mathbf{W}_k^{(n)} = \left(\bigodot_{i=1}^{n-1} \mathbf{U}_t^{(i)}\right) \odot \left(\bigodot_{i=n+1}^{N} \mathbf{U}_{t-1}^{(i)}\right) \odot \mathbf{u}_k^\top. \quad (17)$$

The minimization of (16) can be decomposed into sub-problems for each row $\mathbf{u}_m^{(n)}$ of $\mathbf{U}^{(n)}$, $m = 1, 2, \dots, I_n$, as

$$\underset{\mathbf{u}_m^{(n)}}{\operatorname{argmin}} \sum_{k=t-L_t+1}^{t} \lambda^{t-k} \Big\| \underline{\mathbf{P}}_{k,m}^{(n)} \Big(\big(\hat{\underline{\mathbf{x}}}_{k,m}^{(n)}\big)^\top - \mathbf{W}_k^{(n)}\big(\mathbf{u}_m^{(n)}\big)^\top\Big)\Big\|_F^2, \quad (18)$$

[3]In practice, we can use the non-linear Jacobi iteration scheme to update (17) as $\mathbf{W}_k^{(n)} = \big(\bigodot_{i=1, i\neq n}^N \mathbf{U}_{t-1}^{(i)}\big) \odot \mathbf{u}_k^\top$. This scheme can be useful for parallel and/or distributed processing.

Algorithm 1: Robust Adaptive CP Decomposition (RACP)

Input:
- Tensor slices $\{\boldsymbol{\mathcal{P}}_t \circledast \boldsymbol{\mathcal{X}}_t\}_{t=1}^{\infty}$, $\boldsymbol{\mathcal{X}}_t \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$,
- CP rank r, forgetting factor $\lambda \in (0,1]$,
- Predefined parameters: penalty $\rho > 0$, precision ϵ^{res}, $\epsilon^{out} > 0$, maximum iteration K, relaxation $\alpha \in [1.5, 1.8]$, and $\delta > 0$.

Output: Loading factors $\{\mathbf{U}_t^{(n)}\}_{n=1}^{N}$.

Initialization:
- $\{\mathbf{U}_0^{(n)}\}_{n=1}^{N}$ is initialized randomly,
- $\{\mathbf{S}_0^{(n)}\}_{n=1}^{N} = \delta \mathbf{I}_{r_{\text{CP}}}$.

for $t = 1, 2, \dots$ **do**

Stage 1: Online Outlier Rejection

$\mathbf{H}_{t-1} = \bigodot_{n=1}^{N} \mathbf{U}_{t-1}^{(n)}$

$\mathbf{o}^0, \mathbf{z}^0, \mathbf{u}^0 \leftarrow \mathbf{0}$

for $i = 1, 2, \dots, K$ **do**

$\mathbf{u}^i = \big(\mathbf{H}_{t-1}^\top \mathbf{P}_t \mathbf{H}_{t-1}\big)^{\#} \mathbf{H}_{t-1}^\top \mathbf{P}_t \big(\mathbf{x}_t - \mathbf{o}^{i-1} - \mathbf{z}^{i-1}/\rho\big)$

$\mathbf{r}^i = \alpha \mathbf{P}_t \big(\mathbf{x}_t - \mathbf{H}_{t-1}\mathbf{u}^i\big) + (1-\alpha)\mathbf{o}^{i-1}$

$\mathbf{o}^i = \mathcal{S}_{1/\rho}\big(\mathbf{r}^i - \mathbf{z}^{i-1}/\rho\big)$

$\mathbf{z}^i = \mathbf{z}^{i-1} + \rho(\mathbf{o}^i - \mathbf{r}^i)$

if stopping criteria are met **break**

end

Outlier Removal (Re-update of $\boldsymbol{\mathcal{P}}_t$ in (13) is optional)

$$\boldsymbol{\mathcal{P}}_t \circledast \widehat{\boldsymbol{\mathcal{X}}}_t = \boldsymbol{\mathcal{P}}_t \circledast (\boldsymbol{\mathcal{X}}_t - \boldsymbol{\mathcal{O}}_t)$$

Stage 2: Estimation of $\{\mathbf{U}_t^{(n)}\}_{n=1}^{N}$

for $n = 1, 2, \dots, N$ **do**

$\mathbf{W}_t^{(n)} = \Big(\bigodot_{i=1}^{n-1} \mathbf{U}_t^{(i)}\Big) \odot \Big(\bigodot_{i=n+1}^{N} \mathbf{U}_{t-1}^{(i)}\Big) \odot \mathbf{u}_k^\top$

$\widetilde{\mathbf{W}}_t^{(n)} = \Big[\big(\mathbf{W}_t^{(n)}\big)^\top \quad \big(\mathbf{W}_{t-L_t}^{(n)}\big)^\top \Big]^\top$

for $m = 1, 2 \dots, I_n$ **do**

$\widetilde{\underline{\mathbf{P}}}_{t,m}^{(n)} = \begin{bmatrix} \underline{\mathbf{P}}_{t,m}^{(n)} & \mathbf{0} \\ \mathbf{0} & -\lambda^{L_t} \underline{\mathbf{P}}_{t-L_t,m}^{(n)} \end{bmatrix}$

$\tilde{\underline{\mathbf{x}}}_{t,m}^{(n)} = \Big[\hat{\underline{\mathbf{x}}}_{t,m}^{(n)} \quad \hat{\underline{\mathbf{x}}}_{t-L_t,m}^{(n)} \Big]$

$\mathbf{S}_{t,m}^{(n)} = \lambda \mathbf{S}_{t-1,m}^{(n)} + \big(\widetilde{\mathbf{W}}_t^{(n)}\big)^\top \widetilde{\underline{\mathbf{P}}}_{t,m}^{(n)} \widetilde{\mathbf{W}}_t^{(n)}$

$\mathbf{V}_{t,m}^{(n)} = \big(\mathbf{S}_{t,m}^{(n)}\big)^{-1} \big(\widetilde{\mathbf{W}}_t^{(n)}\big)^\top$

$\boldsymbol{\delta}\tilde{\underline{\mathbf{x}}}_{t,m}^{(n)} = \widetilde{\underline{\mathbf{P}}}_{t,m}^{(n)} \Big(\big(\tilde{\underline{\mathbf{x}}}_{t,m}^{(n)}\big)^\top - \widetilde{\mathbf{W}}_t^{(n)} \big(\mathbf{u}_{t-1,m}^{(n)}\big)^\top \Big)$

$\mathbf{u}_{t,m}^{(n)} = \mathbf{u}_{t-1,m}^{(n)} + \big(\boldsymbol{\delta}\tilde{\underline{\mathbf{x}}}_{t,m}^{(n)}\big)^\top \big(\mathbf{V}_{t,m}^{(n)}\big)^\top$

end

end

Stage 3: (Optional) Normalization and Re-estimation of $\mathbf{u}_t$

Column-wise Normalization:

$$\big[\mathbf{U}_t^{(n)}\big]_{:,r} = \frac{\big[\mathbf{U}_t^{(n)}\big]_{:,r}}{\big\|\big[\mathbf{U}_t^{(n)}\big]_{:,r}\big\|_2^2}.$$

Re-estimation of $\mathbf{u}_t$:

$$\mathbf{u}_t = \big(\mathbf{H}_t^\top \mathbf{P}_t \mathbf{H}_t\big)^{\#} \mathbf{H}_t^\top \mathbf{P}_t (\mathbf{x}_t - \mathbf{o}_t)$$

where $\mathbf{H}_t = \bigodot_{n=1}^{N} \mathbf{U}_t^{(n)}$

end

where $\hat{\underline{\mathbf{x}}}_{k,m}^{(n)}$ is the m-th row of $\widehat{\underline{\mathbf{X}}}_k^{(n)}$, and the row-mask matrix is given by $\underline{\mathbf{P}}_{k,m}^{(n)} = \operatorname{diag}\big(\underline{\mathbf{P}}_k^{(n)}(m,:)\big)$. Here, we introduce an efficient recursive least-squares (RLS) solver to minimize (18) effectively (see Algorithm 1 and the Appendix for its derivation).

Stage 3 (Optional): Normalization and re-estimation of $\mathbf{u}_t$
In order to avoid numerical problems, we can perform the column-wise normalization on the updated factors $\{\mathbf{U}_t^{(n)}\}_{n=1}^{N}$. In addition, given the already estimated factors, the weight vector $\mathbf{u}_t$ in Step 1 can be re-updated to achieve a better estimation as follows

$$\mathbf{u}_t = \big(\mathbf{H}_t^\top \mathbf{P}_t \mathbf{H}_t\big)^{\#} \mathbf{H}_t^\top \mathbf{P}_t \hat{\mathbf{x}}_t, \tag{19}$$

where $\mathbf{H}_t = \bigodot_{n=1}^{N} \mathbf{U}_t^{(n)}$. This step is useful for the early stage of tracking and fast time-varying environments [19], [20].

B. Extensions of the RACP algorithm

In the following, we present two simple modifications of RACP when smoothness and nonnegativity are imposed on the loading factors.

1) Smoothness Condition: In many applications, it is a common assumption that the underlying data or model are smooth [51]. Here, we incorporate a smoothing regularization matrix on the loading factors to control the smoothness of the solution as well as to avoid biases and singular/ill-posed computation. This regularization adds a small bias against large terms to the updating rules.

On the arrival of $\boldsymbol{\mathcal{X}}_t$, the outliers $\boldsymbol{\mathcal{O}}_t$ and the coefficient vector $\mathbf{u}_t$ are derived from the following minimization:

$$\begin{aligned} \{\boldsymbol{\mathcal{O}}_t, \mathbf{u}_t\} &= \underset{\boldsymbol{\mathcal{O}}, \mathbf{u}}{\operatorname{argmin}} \ \big\|\boldsymbol{\mathcal{O}}\big\|_1 + \frac{\gamma}{2}\big\|\mathbf{B}\mathbf{u}\big\|_2^2 \\ \text{s.t.} & \ \big\|\boldsymbol{\mathcal{P}}_t \circledast \big(\boldsymbol{\mathcal{X}}_t - \boldsymbol{\mathcal{O}} - \boldsymbol{\mathcal{H}}_{t-1} \times_{N+1} \mathbf{u}\big)\big\|_F^2 = 0, \end{aligned} \tag{20}$$

where $\boldsymbol{\mathcal{H}}_{t-1} = \boldsymbol{\mathcal{I}} \prod_{n=1}^{N} \times_n \mathbf{U}_{t-1}^{(n)}$ and $\gamma > 0$ is a small penalty parameter and $\mathbf{B}$ a chosen banded matrix. More concretely, the vector $\mathbf{u}_t$ is obtained by minimizing the following problem:

$$\mathbf{u}_t = \underset{\mathbf{u}}{\operatorname{argmin}} \ \frac{\gamma}{2}\big\|\mathbf{B}\mathbf{u}\big\|_2^2 + \frac{\rho}{2}\big\|\mathbf{P}_t(\mathbf{x}_t - \mathbf{o} - \mathbf{H}_{t-1}\mathbf{u})\big\|_2^2. \tag{21}$$

Accordingly, we replace the update rule for $\mathbf{u}$ in (8) with

$$\mathbf{u}^i = \Big(\mathbf{H}_{t-1}^\top \mathbf{P}_t \mathbf{H}_{t-1} + \frac{\gamma}{\rho}\mathbf{B}^\top\mathbf{B}\Big)^{\#} \mathbf{H}_{t-1}^\top \mathbf{P}_t \big(\mathbf{x}_t - \mathbf{o}^i\big), \tag{22}$$

Instead of (18), the m-th row $\mathbf{u}_{t,m}^{(n)}$ of $\mathbf{U}_t^{(n)}$ is derived from

$$\begin{aligned} \underset{\mathbf{u}_m^{(n)}}{\operatorname{argmin}} \sum_{k=t-L_t+1}^{t} \lambda^{t-k} \Big\| \underline{\mathbf{P}}_{k,m}^{(n)} \Big(\big(\hat{\underline{\mathbf{x}}}_{k,m}^{(n)}\big)^\top - \mathbf{W}_k^{(n)} \big(\mathbf{u}_m^{(n)}\big)^\top \Big) \Big\|_2^2 \\ + \frac{\gamma}{2} \Big\| \mathbf{B} \big(\mathbf{u}_m^{(n)}\big)^\top \Big\|_2^2, \end{aligned} \tag{23}$$

In particular, $\mathbf{u}_{t,m}^{(n)}$ is the solution of the following equation:

$$\begin{aligned} &\sum_{k=t-L_t+1}^{t} \lambda^{t-k} \big(\mathbf{W}_k^{(n)}\big)^\top \underline{\mathbf{P}}_{k,m}^{(n)} \big(\hat{\underline{\mathbf{x}}}_{k,m}^{(n)}\big)^\top \\ &= \Big(\sum_{k=t-L_t+1}^{t} \lambda^{t-k} \big(\mathbf{W}_k^{(n)}\big)^\top \underline{\mathbf{P}}_{k,m}^{(n)} \mathbf{W}_k^{(n)} + \frac{\gamma}{2}\mathbf{B}^\top\mathbf{B} \Big) \big(\mathbf{u}_m^{(n)}\big)^\top. \end{aligned} \tag{24}$$

Therefore, the recursive rule for updating $\mathbf{u}_{t,m}^{(n)}$ becomes

$$\mathbf{u}_{t,m}^{(n)} = \mathbf{u}_{t-1,m}^{(n)} + \big(\boldsymbol{\delta}\tilde{\underline{\mathbf{x}}}_{t,m}^{(n)}\big)^\top \big(\bar{\mathbf{V}}_{t,m}^{(n)}\big)^\top, \tag{25}$$

where

$$\bar{\mathbf{V}}_{t,m}^{(n)} = \Big(\mathbf{S}_{t,m}^{(n)} + \frac{\gamma}{2}\mathbf{B}^\top\mathbf{B}\Big)^{-1} \big(\widetilde{\mathbf{W}}_t^{(n)}\big)^\top.$$

2) Nonnegative Constraint: It is known that nonnegative tensor factorization (NTF) offers interesting properties, e.g., the resulting expression appears to be purely additive and the loading factors are "sparse" in general [52].

One of the simplest ways is to project the estimates (i.e., $\mathbf{u}_t$ and $\{\mathbf{U}_t^{(n)}\}_{n=1}^N$) on their nonnegative orthant at the end of each step of RACP, as introduced by Nguyen *et al.* in [53]. This approach offers a low complexity and yields a reasonable performance in some cases. However, it may not be optimal nor guarantee convergence in general. In this task, we aim to customize the updates of $\mathbf{u}_t$ and $\{\mathbf{U}_t^{(n)}\}_{n=1}^N$ in order to deal with nonnegativity at each time t.

In step 1, we replace the exact LS solution (8) with the minimizer of the following nonnegative least-squares (NNLS) problem:

$$\mathbf{u}^i = \underset{\mathbf{u}}{\operatorname{argmin}} \; \left\| \mathbf{P}_t\big(\mathbf{x}_t - \mathbf{o}^i - \mathbf{H}_{t-1}\mathbf{u}\big) \right\|_2^2 \text{ s.t. } [\mathbf{u}]_j \geq 0 \;\; \forall j. \quad (26)$$

Here, we can apply any provable NNLS algorithm to solve (26). The reader is referred to [54], [55] for good surveys on numerical methods for NNLS. In this work, we adopt the widely-used algorithm of Lawson and Hanson [55] which is implemented as the function `lsqnonneg` in MATLAB.

In step 2, the m-th row of $\mathbf{U}_t^{(n)}$ can be derived by minimizing the following constrained version of (18):

$$\begin{aligned} \mathbf{u}_{t,m}^{(n)} = \underset{\mathbf{u}_m^{(n)}}{\operatorname{argmin}} \sum_{k=t-L_t+1}^{t} \lambda^{t-k} \left\| \underline{\mathbf{P}}_{k,m}^{(n)} \Big(\big(\hat{\underline{\mathbf{x}}}_{k,m}^{(n)}\big)^\top - \mathbf{W}_k^{(n)} \big(\mathbf{u}_m^{(n)}\big)^\top \Big) \right\|_2^2, \\ \text{s.t. } \left[\mathbf{u}_m^{(n)}\right]_j \geq 0 \quad \forall j. \qquad (27) \end{aligned}$$

In order to solve (27), we apply the projected gradient method (i.e. proximal gradient on indicator function [56]). More concretely, the iterative procedure for updating $\mathbf{u}_{t,m}^{(n)}$ is given by[4]

$$\mathbf{u}_l = \left[\left(\mathbf{I}_r - \frac{\mathbf{S}_{t,m}^{(n)}}{\|\mathbf{S}_{t,m}^{(n)}\|_2} \right) \mathbf{u}_{l-1} - \frac{\mathbf{d}_{t,m}^{(n)}}{\|\mathbf{S}_{t,m}^{(n)}\|_2} \right]_+, \qquad (28)$$

where l denotes the iteration index. We refer to this modification of RACP as NRACP.

IV. Performance Analysis

In this section, we present a theoretical convergence analysis for the proposed RACP method in Algorithm 1 while assuming that the underlying tensor dictionary $\mathbf{D}$ does not change over time. Inspired by the recent results of our companion studies on robust subspace tracking [49] and tensor tracking [27], we establish a unified theoretical approach to analyse the convergence of the objective values $\{f_t(\mathbf{D}_t)\}_{t=1}^\infty$ as well as the solutions $\{\mathbf{D}_t\}_{t=1}^\infty$ generated by RACP.

[4]Projected gradient descent has a form of $\mathbf{u}_j = \big[\mathbf{u}_{j-1} - \eta_j \nabla \tilde{f}_t(\mathbf{u}_{j-1})\big]_+$, where $\nabla \tilde{f}_t\big(\mathbf{u}_m^{(n)}\big) = \mathbf{S}_{t,m}^{(n)} \mathbf{u}_m^{(n)} - \mathbf{d}_{t,m}^{(n)}$. In practice, we can set the value of the step-size η_j to $1/\mathcal{L}$ where $\mathcal{L}$ is the Lipschitz constant of $\nabla \tilde{f}_t\big(\mathbf{u}_m^{(n)}\big)$. In this work, it is easy to show that $\mathcal{L} = \|\mathbf{S}_{t,m}^{(n)}\|_2$.

A. Assumptions

In order to facilitate the convergence analysis, we make the following assumptions[5]:

(A1): Low-rank components $\{\boldsymbol{\mathcal{Y}}_t\}_{t\geq1}$ of the observed tensor slices $\{\boldsymbol{\mathcal{X}}_t\}_{t\geq1}$ are assumed to be deterministic and bounded. Entries of noise tensors $\{\boldsymbol{\mathcal{N}}_t\}_{t\geq1}$ are zero-mean, independently and identically distributed (i.i.d.) with a small finite covariance, and bounded. Entries of $\boldsymbol{\mathcal{X}}_t$ are Frobenius-norm bounded, i.e., $\|\boldsymbol{\mathcal{X}}_t\|_F \leq M_x < \infty$, for all t.

(A2): The tensor factors $\{\mathbf{U}^{(n)}\}_{n=1}^N$ remain unchanged over time, i.e., the tensor dictionary $\mathbf{D}$ is fixed. The loading factors are Frobenius-norm bounded and the tensor rank r is fixed.

(A3): Observation masks $\{\boldsymbol{\mathcal{P}}_t\}_{t\geq1}$ are independent of $\{\boldsymbol{\mathcal{X}}_t\}_{t\geq1}$, and their entries follow a uniform distribution. The number of observed entries of $\boldsymbol{\mathcal{X}}_t$ should be larger than the lower bound $\mathcal{O}\big(rL\log(L)\big)$, where $L = I_1 I_2 \dots I_N$. Every row of the mode-n unfolding $\underline{\mathbf{X}}_t^{(n)}$ of $\boldsymbol{\mathcal{X}}_t$ is observed in at least r entries, for $n = 1, 2, \dots, N$. In addition, each observed entry of $\boldsymbol{\mathcal{X}}_t$ is corrupted by outliers independently of others, i.e., the index of outliers is also uniformly random.

(A4): The surrogate function $\tilde{f}_t(\cdot)$ is m-strongly multi-block convex, i.e., its second-order derivative w.r.t. each factor is positive-definite, $\nabla_n^2 \tilde{f}_t\big(\mathbf{U}^{(n)}, .\big) \geq m\mathbf{I} > \mathbf{0}$ with $m > 0$.

Among these assumptions, (A1) and (A2) are common for analysing the convergence of online learning algorithms, such as [24], [46], [49]. Indeed, (A1) holds in many situations, e.g., real data, such as audio, image and video data, are often bounded. (A2) is a strong assumption as it requires the tensor dictionary to be constant with time. Also, the bound in (A2) prevents arbitrarily large values in $\mathbf{U}^{(n)}$ and ill-conditioned computation. Along with (A1), it is interpreted as the simplest possible data model in (robust) tensor tracking where tensor slices are assumed to be generated from a stationary process. Theoretically, stationary processes are often "easier" to model and analyse than nonstationary ones as their statistical properties remain constant over time. Accordingly, stationarity has become a common assumption underlying many statistical procedures in general and tracking tools in particular to study their convergence and asymptotic behavior. In this work, a novel theoretical approach is established to analyse the convergence behavior of RACP in stationary environments. We leave the convergence analysis of RACP under a nonstationary model where the tensor dictionary is time-varying to a future work. Assumption (A3) is also common, under which the index of missing entries is uniformly random. Moreover, with respect to the imputation of missing values and recovery of low-rank components, the uniform randomness allows the sequence of binary masks $\{\boldsymbol{\mathcal{P}}_t\}_{t\geq1}$ to admit stable recovery [57]. The next two constraints of (A3) are fundamental conditions to prevent the underdetermined imputation problem [58]–[60]. The last constraint of (A3) plays a similar role as the first one but accounting for sparse outliers. Assumption (A4) allows us to derive several nice results in the convergence analysis. In fact, as the Hessian matrix of $\tilde{f}_t(\cdot)$ w.r.t. each factor is

[5]The four assumptions (A1)-(A4) are used for the purpose of convergence analysis only. The proposed RACP algorithm can work well in many other scenarios (see Section V for an illustration).

already positive semidefinite, (A4) can be achieved with a good initialization $\mathbf{D}_0$ or by simply adding a convex regularization term to $\tilde{\ell}(\cdot)$ or $\tilde{f}_t(\cdot)$.

B. Main Results

Given the assumptions of (A1)-(A4), our main theoretical result can be stated in the following theorem:

Theorem 1. *Given (A1)-(A4), $L_t = t$ and let $\mathbf{D}_t$ be the solution generated by Algorithm 1 at each time t. When $t \to \infty$,*

- $f_t(\mathbf{D}_t) - \tilde{f}_t(\mathbf{D}_t) \overset{a.s.}{\to} 0$;
- $\nabla f_t(\mathbf{D}_t) \overset{a.s.}{\to} 0$.

Accordingly, $\mathbf{D}_t$ is almost surely a stationary point of $f_t(.)$ when t tends to infinity.

The proof of this theorem follows intermediately Proposition 1 and Lemmas 1 and 2, to be stated shortly. We detail their proofs in our Supplementary Material attached to this manuscript.

Proposition 1 (Key Properties)**.** *Given (A1)-(A4), $L_t = t$, and denote the error function $e_t := \tilde{f}_t - f_t$. If $\{\mathbf{D}_t, \boldsymbol{\mathcal{O}}_t, \mathbf{u}_t\}_{t=1}^{\infty}$ is a sequence of variables generated by Algorithm 1, then*

(a) *Boundedness: $\{\mathbf{D}_t, \boldsymbol{\mathcal{O}}_t, \mathbf{u}_t\}_{t=1}^{\infty}$ are uniformly bounded;*
(b) *Forward Monotonicity: $\tilde{f}_t(\mathbf{D}_{t-1}) \geq \tilde{f}_t(\mathbf{D}_t)$;*
(c) *Backward Monotonicity: $\tilde{f}_t(\mathbf{D}_t) \leq \tilde{f}_t(\mathbf{D}_{t+1})$;*
(d) *Stability of estimates: $\|\mathbf{D}_t - \mathbf{D}_{t-1}\|_F = \mathcal{O}(1/t)$;*
(e) *Stability of errors: $|e_t(\mathbf{D}_t) - e_{t-1}(\mathbf{D}_{t-1})| = \mathcal{O}(1/t)$.*

Proof Sketch. Part (a) can be derived from applying the same arguments of Proposition 1 in our companion work on tensor tracking [27]. Part (b) and (c) are trivial due to the proposed iteration scheme. Part (d) can be obtained by exploiting the Lipschitz continuity and multi-block convexity of the surrogate function $\tilde{f}_t(.)$. We indicate Part (e) by using Part (d) and the Lipschitz continuity of $f_t(.)$ and $\tilde{f}_t(.)$. □

Lemma 1 (Almost sure convergence)**.** *The sequence of $\{\tilde{f}_t(\mathbf{D}_t)\}_{t=1}^{\infty}$ converges almost surely as $t \to \infty$. The sequence of objective values $\{f_t(\mathbf{D}_t)\}_{t=1}^{\infty}$ converges to the same limit of its surrogate $\{\tilde{f}_t(\mathbf{D}_t)\}_{t=1}^{\infty}$, i.e.,*

$$f_t(\mathbf{D}_t) \to \tilde{f}_t(\mathbf{D}_t) \quad a.s. \tag{29}$$

Proof Sketch. We first prove that

$$\sum_{t=1}^{\infty} \mathbb{E}\Big[\delta_t \mathbb{E}\big[\tilde{f}_{t+1}(\mathbf{D}_{t+1}) - \tilde{f}_t(\mathbf{D}_t) \big| \mathcal{F}_t\big]\Big] < \infty, \tag{30}$$

where $\mathcal{F}_t = \{\mathbf{D}_\tau, \boldsymbol{\mathcal{O}}_\tau, \mathbf{u}_\tau\}_{0<\tau\leq t}$ records all past estimates of RACP at time t and the indicator function δ_t is defined as

$$\delta_t \triangleq \begin{cases} 1 & \text{if} \quad \mathbb{E}\big[\tilde{f}_{t+1}(\mathbf{D}_{t+1}) - \tilde{f}_t(\mathbf{D}_t) \big| \mathcal{F}_t\big] > 0, \\ 0 & \text{otherwise.} \end{cases} \tag{31}$$

Thanks to the quasi-martingale convergence theorem [61, page 51], (30) implies that $\{\tilde{f}_t(\mathbf{D}_t)\}_{t=1}^{\infty}$ converges almost surely as $t \to \infty$.

We next prove $\{f_t(\mathbf{D}_t)\}_{t=1}^{\infty}$ and $\{\tilde{f}_t(\mathbf{D}_t)\}_{t=1}^{\infty}$ converge to the same limit by showing

$$\sum_{t=1}^{\infty} \frac{\tilde{f}_t(\mathbf{D}_t) - f_t(\mathbf{D}_t)}{t+1} < \infty. \tag{32}$$

Since $\sum_{t=1}^{\infty} \frac{1}{t+1} = \infty$ and $\big|e_t(\mathbf{D}_t) - e_{t-1}(\mathbf{D}_{t-1})\big| = \mathcal{O}(1/t)$, we obtain $\sum_{t=1}^{\infty} \tilde{f}_t(\mathbf{D}_t) - f_t(\mathbf{D}_t) < \infty$, or

$$\tilde{f}_t(\mathbf{D}_t) \to f_t(\mathbf{D}_t) \; a.s., \tag{33}$$

thanks to [46, Lemma 3]. □

Lemma 2 (Local convergence)**.** *When $t \to \infty$, $\mathbf{D}_t$ converges almost surely to a stationary point of $\tilde{f}_\infty(.) = \lim_{t\to\infty} \tilde{f}_t(.)$:*

$$\nabla \tilde{f}_\infty(\mathbf{D}_t) \to \nabla f_\infty(\mathbf{D}_t) \to 0 \quad a.s. \tag{34}$$

Proof Sketch. We first indicate that

$$\lim_{t\to\infty} \operatorname{tr}\big[(\mathbf{D}_t - \mathbf{D}_{t+1})^\top \nabla \tilde{f}_{t+1}(\mathbf{D}_{t+1})\big] = 0, \tag{35}$$

by showing $\sum_{t=1}^{\infty} \Big|\operatorname{tr}\big[(\mathbf{D}_t - \mathbf{D}_{t+1})^\top \nabla \tilde{f}_{t+1}(\mathbf{D}_{t+1})\big]\Big| < \infty$.

Next, we prove that the following inequality

$$\begin{aligned} \operatorname{tr}\big[(\mathbf{D}_t - \mathbf{D}_{t+1})^\top \nabla \tilde{f}_{t+1}\big(\mathbf{D}_{t+1}\big)\big] &\leq c_1 \big\|\mathbf{D}_{t+1} - \mathbf{D}_t\big\|_F^2 \\ &+ c_2 \operatorname{tr}\big[(\mathbf{D} - \mathbf{D}_t)^\top \nabla \tilde{f}_{t+1}\big(\mathbf{D}_t\big)\big], \end{aligned} \tag{36}$$

holds for all $\mathbf{D} \in \mathcal{D}$ where c_1 and c_2 are positive constants.

Then, we use proof by contradiction to indicate that

$$\big(\nabla \tilde{f}_\infty(\mathbf{D}_\infty)\big)^\top (\mathbf{D} - \mathbf{D}_\infty) \geq \mathbf{0}, \;\; \forall \mathbf{D} \in \mathcal{D}. \tag{37}$$

Accordingly, $\mathbf{D}_\infty$ is a stationary point of $\tilde{f}_\infty(.)$.

In order to prove $\nabla \tilde{f}_t\big(\mathbf{D}_t\big) \overset{a.s.}{\to} \nabla f_t\big(\mathbf{D}_t\big)$ as $t \to \infty$, we first exploit that $f_t(\mathbf{D} + a_t\mathbf{V}) \leq \tilde{f}_t(\mathbf{D} + a_t\mathbf{V})$ $\forall \mathbf{D}, \mathbf{V} \in \mathcal{D}$ and a_t, and then take its Taylor expansion at $t \to \infty$ to yield

$$\begin{aligned} f_\infty\big(\mathbf{D}_\infty\big) &+ \operatorname{tr}\big[a_t \mathbf{V}^\top \nabla f_\infty\big(\mathbf{D}_\infty\big)\big] + \boldsymbol{o}\big(a_t\mathbf{V}\big) \\ &\leq \tilde{f}_\infty\big(\mathbf{D}_\infty\big) + \operatorname{tr}\big[a_t \mathbf{V}^\top \nabla \tilde{f}_\infty\big(\mathbf{D}_\infty\big)\big] + \boldsymbol{o}\big(a_t\mathbf{V}\big). \end{aligned} \tag{38}$$

As indicated in Lemma 1, $\tilde{f}_\infty\big(\mathbf{D}_\infty\big) = f_\infty\big(\mathbf{D}_\infty\big)$ and thus $\operatorname{tr}\big[a_t \mathbf{V}^\top \nabla f_\infty\big(\mathbf{D}_\infty\big)\big] \leq \operatorname{tr}\big[a_t \mathbf{V}^\top \nabla \tilde{f}_\infty\big(\mathbf{D}_\infty\big)\big]$. Since the above inequality must hold for all $\mathbf{V} \in \mathcal{D}$ and a_t, we obtain

$$\nabla \tilde{f}_\infty\big(\mathbf{D}_\infty\big) = \nabla f_\infty\big(\mathbf{D}_\infty\big). \tag{39}$$

Together with (37), we can conclude that $\mathbf{D}_\infty$ is a stationary point of the objective function $f_t(.)$ as $t \to \infty$. □

C. Discussion

Our analysis follows the same framework to derive the convergence of adaptive/incremental algorithms for online matrix/tensor factorization problems as in [24], [25], [27], [46], [48], [49]. Therefore, our main theoretical result is somewhat similar to their results. However, there are several points that make our convergence analysis different from theirs.

First, [46] is devoted to the problem of online dictionary learning and sparse coding. The authors dealt with a LASSO-like cost function and required a preliminary uniqueness condition on the sparse coding. The condition is important to ensure that the solution generated in the sparse coding stage is unique, and to derive the Lipschitz property of the cost function. Particularly, they suggested an elastic-net regularized term for enforcing the condition. Since the problem formulation of RTT is different, our convergence analysis does not involve such issues. Moreover, the missing data distinguish our work from theirs.

The studies in [48] and [49] consider the problem of robust online PCA/subspace tracking which can handle data corruptions (i.e., outliers and/or missing entries). These studies are designed for tracking the time-variant subspace – an object different from ours – which leads to some differences from our analysis. In particular, their main goal is to develop provable algorithms for minimizing the expected cost function in an online manner, and then indicate that their algorithm converges to a stationary point or global optimum under certain conditions. Our optimization, however, minimizes an exponential weighted cost function constructed on the latest data streams (i.e., tensor slices). Moreover, [48] does not require the solution derived from the subspace update stage to be necessarily optimal, but full column rank only at each time t (see [48, Theorem 1]). However, it is a sufficient condition that is highly leveraged in our analysis. In addition, our object is a set of multiple loading factors, instead of a single subspace matrix as in [48], [49].

The studies most related to ours are those in [24], [25], [27], which also investigate the tensor tracking problem. However, they consider only outlier-free streaming tensors. By contrast, we here provide a more unified convergence analysis that is able to deal with both missing data and outliers. Also, our results are stronger than those of [24], [25], which are limited to the case of third-order streaming tensors with $\lambda = 1$.

V. Experiments

In this section, we provide several experiments on both synthetic and real data to demonstrate the effectiveness of RACP and its variant. In particular, the performance of our method is evaluated in comparison with the-state-of-the-art algorithms with respect to the following aspects: (i) impact of outliers, (ii) impact of missing data, and (iii) tracking ability in noisy and time-varying environments.

A. Experiment Setup

At $t = 0$, we randomly initialize $\mathbf{U}_0^{(n)} \in \mathbb{R}^{I_n \times r}$ whose entries are i.i.d. from a normal distribution $\mathcal{N}(0,1)$, $n = 1,2,\dots,N$. When $t \geq 1$, $\mathbf{U}_t^{(n)}$ is varied according to the following model:

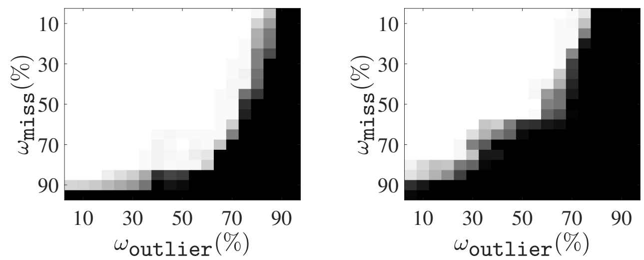
$$\mathbf{U}_t^{(n)} = \mathbf{U}_{t-1}^{(n)} + \epsilon \mathbf{N}_t^{(n)}, \tag{40}$$

where $\mathbf{N}_t^{(n)}$ is a Gaussian noise matrix (with zero-mean and unit-variance), and ϵ is a positive time-varying factor used to control the variation of $\mathbf{U}^{(n)}$ between t and $t-1$.

The t-th slice $\boldsymbol{\mathcal{X}}_t$ is then generated under the data model

$$\boldsymbol{\mathcal{X}}_t = \boldsymbol{\mathcal{P}}_t \circledast \left(\boldsymbol{\mathcal{I}} \prod_{n=1}^{N} \times_n \mathbf{U}_t^{(n)} \times_{N+1} \mathbf{u}_t^\top + \boldsymbol{\mathcal{O}}_t + \boldsymbol{\mathcal{N}}_t \right), \tag{41}$$

where $\boldsymbol{\mathcal{P}}_t$ is a binary observation mask according to a Bernoulli distribution with probability of observing data $1 - \omega_{\texttt{miss}}$, $\boldsymbol{\mathcal{N}}_t$ is a Gaussian noise tensor with i.i.d. entries $\mathcal{N}(0, \sigma_n^2)$, $\boldsymbol{\mathcal{O}}_t$ is a sparse outlier tensor whose entries are drawn uniformly from the range $[0, A_{\texttt{outlier}}]$ and the indices of outliers also follow a Bernoulli distribution with probability $\omega_{\texttt{outlier}}$, and $\mathbf{u}_t \in \mathbb{R}^{r \times 1}$ is a standard normal random vector.



(a) $r = 2$ (b) $r = 5$

Fig. 1: Effect of data corruptions (outliers and missing values) on performance of RACP. White color denotes perfect estimation (i.e., $\mathrm{RE}(\hat{\mathbf{D}}, \mathbf{D}) \leq 0.01$), black color denotes failure (i.e., $\mathrm{RE}(\hat{\mathbf{D}}, \mathbf{D}) \geq 0.5$), and gray color is in between.

To evaluate the estimation accuracy, we use the metric

$$e\big(\mathbf{Z}_{\texttt{estimate}}, \mathbf{Z}_{\texttt{true}}\big) = \frac{\|\mathbf{Z}_{\texttt{estimate}} - \mathbf{Z}_{\texttt{true}}\|_F}{\|\mathbf{Z}_{\texttt{true}}\|_F}, \tag{42}$$

where $\mathbf{Z}_{\texttt{estimate}}$ (resp. $\mathbf{Z}_{\texttt{true}}$) refers to the estimation (resp. ground truth). Due to the permutation and scaling indeterminacy of CP decomposition, the estimation $\hat{\mathbf{U}}_t^{(n)}$ of $\mathbf{U}_t^{(n)}$ at each time t will be permuted and scaled such that it matches $\mathbf{U}_t^{(n)}$ before measuring the error metric (42). In particular, we derive the ordered and scaled version $\hat{\mathbf{U}}_{re-t}^{(n)}$ of $\hat{\mathbf{U}}_t^{(n)}$ from

$$\hat{\mathbf{U}}_{re-t}^{(n)} = \hat{\mathbf{U}}_t^{(n)} \big(\mathbf{P}^{(n)}\big)^\top \big(\mathbf{Q}^{(n)}\big)^{-1}, \quad n = 1,2,\dots,N, \tag{43}$$

where the permutation matrix $\mathbf{P}^{(n)} \in \mathbb{R}^{r \times r}$ and the diagonal matrix $\mathbf{Q}^{(n)} \in \mathbb{R}^{r \times r}$ are obtained by

$$\big\{\mathbf{P}^{(n)}, \mathbf{Q}^{(n)}\big\} = \underset{\mathbf{P},\mathbf{Q}}{\operatorname{argmin}} \big\| \hat{\mathbf{U}}_t^{(n)} - \mathbf{U}_t^{(n)} \mathbf{P}\mathbf{Q} \big\|_F^2. \tag{44}$$

The relative errors are then computed as

$$\mathrm{RE}\big(\hat{\mathbf{D}}_t, \mathbf{D}_t\big) = \frac{1}{N} \sum_{n=1}^{N} e\big(\hat{\mathbf{U}}_{re-t}^{(n)}, \mathbf{U}_t^{(n)}\big), \tag{45}$$

$$\mathrm{RE}\big(\hat{\boldsymbol{\mathcal{X}}}_t, \boldsymbol{\mathcal{X}}_t\big) = e(\hat{\boldsymbol{\mathcal{X}}}_t, \boldsymbol{\mathcal{X}}_t), \tag{46}$$

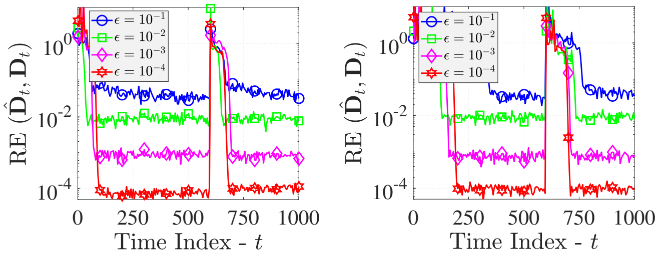
where $\hat{\boldsymbol{\mathcal{X}}}_t$ is a reconstructed version of the true slice $\boldsymbol{\mathcal{X}}_t$ derived from the recent updated loading factors.

B. Robustness of RACP

We first investigated the robustness of RACP against gross data corruptions. Specifically, we changed the density of outliers and missing data, and then measured the relative error between the ground truth and RACP's estimation.

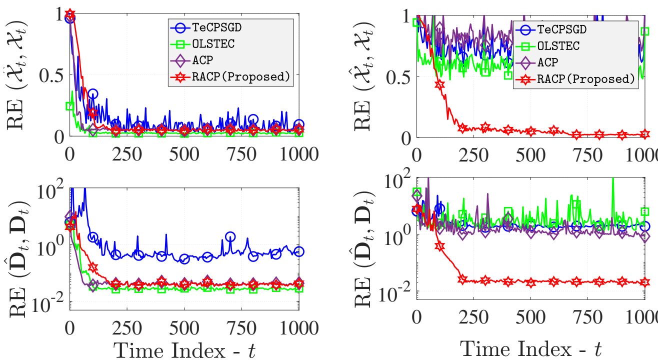
In this task, we used a synthetic 4[th]-order streaming tensor of size $20 \times 20 \times 20 \times 1000$ and the CP rank was set at $r = 2$ and $r = 5$. The noise level σ_n and the time-varying factor ϵ were fixed at 10^{-3} and 10^{-2}, respectively. We consider the case where the underlying data were corrupted by strong outliers with $A_{\texttt{outlier}} = 10$. The fraction of outliers ($\omega_{\texttt{outlier}}$) and missing data ($\omega_{\texttt{miss}}$) were varied in the range $[5\%, 95\%]$. Throughout our experiments, the forgetting factor λ was fixed at 0.5 while the window length was $L_t = t$.

Phase transitions w.r.t. the pair of $\{\omega_{\texttt{outlier}}, \omega_{\texttt{miss}}\}$ are shown in Fig. 1. The results indicate that there is a large region in which our estimation was successful. Particularly, RACP



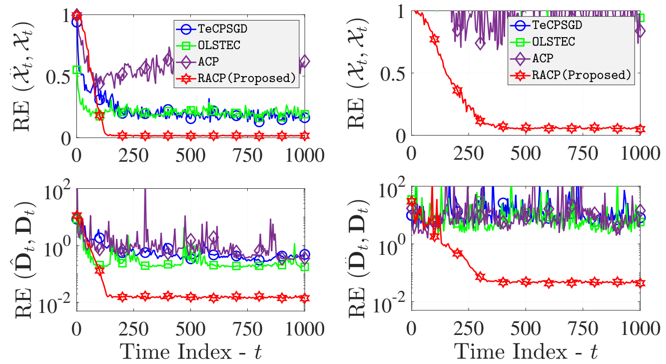
(a) 4[th]-order: $20 \times 20 \times 20 \times 1000$ (b) 5[th]-order: $20 \times 20 \times 20 \times 20 \times 1000$

Fig. 2: Performance of RACP in time-varying environments.



(a) $A_{\texttt{outlier}} = 1$ (small) (b) $A_{\texttt{outlier}} = 10$ (strong)

Fig. 3: Impact of outlier intensity ($A_{\texttt{outlier}}$) on performance of adaptive CP algorithms; $\omega_{\texttt{miss}} = 10\%$, $\omega_{\texttt{outlier}} = 20\%$, $\sigma = 10^{-2}$, $\varepsilon = 10^{-2}$.



(a) $\omega_{\texttt{outlier}} = 10\%$ (b) $\omega_{\texttt{outlier}} = 50\%$

Fig. 4: Impact of outlier density ($\omega_{\texttt{outlier}}$) on performance of adaptive CP algorithms: $\omega_{\texttt{miss}} = 10\%$, $\sigma = 10^{-2}$, $\varepsilon = 10^{-2}$, $A_{\texttt{outlier}} = 10$.

worked well when the number of "clean" data is large enough. In the presence of huge data corruptions (e.g., $\omega_{\texttt{outlier}} \geq 70\%$ and/or $\omega_{\texttt{miss}} \geq 70\%$), the proposed algorithm failed to track the underlying tensor model.

Next, we evaluated the tracking ability of RACP in time-varying environments. The two synthetic rank-5 tensors of size $20 \times 20 \times 20 \times 1000$ and $20 \times 20 \times 20 \times 20 \times 1000$ were used in this task. The fraction of missing entries and sparse outliers were both set to 5%. The outlier intensity $A_{\texttt{outlier}}$ and the noise factor σ_n were fixed at 10 and 10^{-4}, respectively. The value of the time-varying factor ϵ was varied from $[10^{-4}, 10^{-1}]$. An abrupt change was created at $t = 600$ to assess how fast RACP converges. We can see from Fig. 2 that RACP's convergence rate is not much affected by the value of ϵ but that its estimation accuracy is.

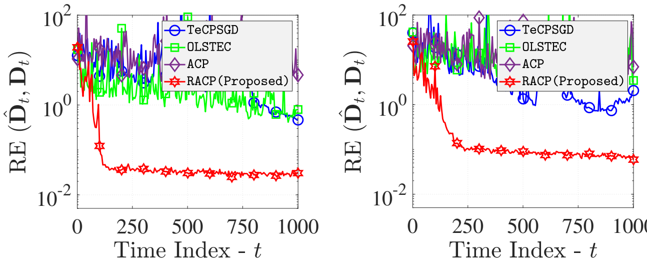
To demonstrate the effectiveness of the proposed algorithm, we compared the performance of RACP with the state-of-the-art adaptive CP decompositions, namely TeCPSGD [24], OLSTEC [25], and ACP [27]. To have a fair comparison, the algorithm parameters were set by default as suggested by their authors. These algorithms are dependent on a forgetting factor; we set its value at 0.7, 0.001, and 0.5 for OLSTEC, TeCPSGD, and ACP, respectively. The penalty parameter was set at 10^{-3} and 10^{-1} for OLSTEC and TeCPSGD, respectively.

Since OLSTEC and TeCPSGD are only capable of tracking third-order streaming tensors, we here used a synthetic streaming tensor of size $20 \times 20 \times 1000$ and its rank was fixed at 5. The noise level and time-varying factor were both kept at 10^{-2}.

Performance comparison results are shown in Figs. 3 and 4. Fig. 3 illustrates the impact of the outlier intensity on the performance of the four adaptive CP algorithms in the presence of 10% missing data and 20% outliers. When the outlier intensity is small, all algorithms were able to track the underlying tensor model over time, as shown in Fig. 3(a). TeCPSGD yielded a worse estimation than the other three adaptive CP algorithms. In the presence of strong outliers, the state-of-the-art adaptive CP algorithms failed to update the tensor basis and recover the corrupted tensor slice. By contrast, our RACP algorithm still worked well, as shown in Fig. 3(b). Fig. 4 illustrates the impact of the outlier density on the performance of RACP against the three adaptive CP algorithms when the missing density $\omega_{\texttt{miss}} = 10\%$ and outlier intensity $A_{\texttt{outlier}} = 10$. We can see that RACP outperformed OLSTEC, TeCPSGD, and ACP in all testing cases. Similar to the case study of strong outliers, the state-of-the-art adaptive algorithms were unable to track the streaming tensors when the number of outliers was large.

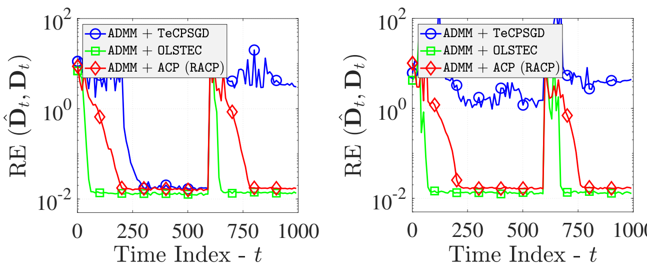
We next investigated the performance of RACP when loading factors are not normal in comparison with other adaptive CP algorithms. In particular, the initial factors $\{\mathbf{U}_0^{(n)}\}_{n=1}^N$ were sampled from a uniform distribution on the $(0, 1)$ interval instead of a Gaussian one. The time-varying model (40) was replaced with $\mathbf{U}_t^{(n)} = \mathbf{U}_{t-1}^{(n)} + \epsilon \mathbf{N}_t^{(n)}$ where $\mathbf{N}_t^{(n)}$ was also an i.i.d. uniform random matrix from 0 to 1. The parameter specifications were kept as in the previous experiment. Results are illustrated in Fig. 5. We can see that the proposed RACP algorithm still tracks the loading factors successfully over time while the state-of-the-art CP algorithms failed.

The experimental results in Figs. 3, 4, and 5 suggest that the outlier rejection step (e.g. Step 1 in RACP) using the ADMM solver plays an important role in the tracking process when observations are corrupted by sparse outliers. Therefore, we next evaluated the effectiveness of the proposed outlier rejection by applying the ADMM solver to other trackers: TeCPSGD and OLSTEC. We here reused the experiment setup above and created an abrupt change at $t = 600$. We can see from Fig. 6 that the combination of the ADMM solver and OLSTEC resulted in the best convergence rate and estimation accuracy. This is probably due to the effectiveness of the



(a) $\omega_{\texttt{miss}} = \omega_{\texttt{outlier}} = 5\%$ (b) $\omega_{\texttt{miss}} = \omega_{\texttt{outlier}} = 20\%$

Fig. 5: Non-Gaussian loading factors.



(a) $\omega_{\texttt{miss}} = \omega_{\texttt{outlier}} = 5\%$ (b) $\omega_{\texttt{miss}} = \omega_{\texttt{outlier}} = 20\%$

Fig. 6: Outlier rejection with different trackers.

second-order estimator in slowly time-varying environments. Our RACP provided a reasonable performance compared to that of OLSTEC, while the TeCPSGD tracker did not work well. It should be noted that OLSTEC is designed for only 3$^{\text{rd}}$-order streaming tensors and that its computational complexity is very high. Our tracker is much faster and capable of dealing with higher-order streaming tensors. We refer readers to our companion work in [27] for further comparisons of ACP against TeSGD and OLSTEC.

Lastly, we conducted a performance comparison between the original RACP and its variant in which the step of re-updating $\boldsymbol{\mathcal{P}}_t$ defined as in (13) was used. We reused the two rank-5 tensors of size $20 \times 20 \times 20 \times 1000$ and $20 \times 20 \times 20 \times 20 \times 1000$. The fraction of missing entries was fixed at 10%. We set the outlier density and intensity to 10% and 10, respectively. The noise and time-varying factors were kept at 10^{-2} and an abrupt change at $t = 600$ was also created as in previous experiments. The results are illustrated in Fig. 7. It can be seen that the outlier rejection mechanism can help improve the convergence rate of RACP.

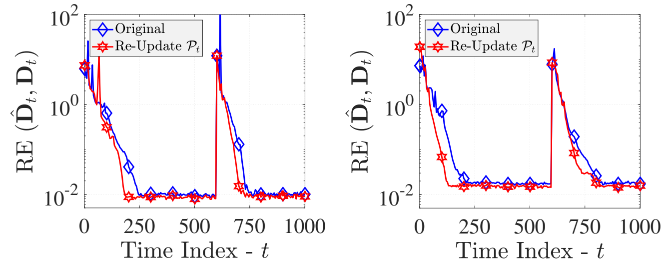
C. Nonnegative RACP

We reused the experiment setup in Section V-A, but the time variation of $\mathbf{U}^{(n)} \geq 0$ was modified as

$$\mathbf{U}_t^{(n)} = \operatorname{abs}\left(\mathbf{U}_{t-1}^{(n)} + \epsilon \mathbf{N}_t^{(n)}\right), \tag{47}$$

where $\operatorname{abs}(\cdot)$ denotes the absolute value, $\mathbf{N}_t^{(n)}$ is a Gaussian noise matrix with i.i.d. entries, and ϵ is to control the variation.

We first investigated the performance of NRACP against time-varying environments. A synthetic rank-5 nonnegative tensor of size $50 \times 50 \times 50 \times 1000$ was used in this task. We consider the case where 10% of the measurements are corrupted by outliers with $A_{\texttt{outlier}} = 10$ and the noise level is $\sigma_n = 10^{-3}$. An abrupt change at $t = 600$ was created to evaluate how fast



(a) 4$^{\text{th}}$-order: $20 \times 20 \times 20 \times 1000$ (b) 5$^{\text{th}}$-order: $20 \times 20 \times 20 \times 20 \times 1000$

Fig. 7: Convergence rate of RACP and its modification with the re-update of $\boldsymbol{\mathcal{P}}_t$ as defined in (13): $\omega_{\texttt{miss}} = 10\%$, $\omega_{\texttt{outlier}} = 10\%$, $A_{\texttt{outlier}} = 10$, $\sigma = 10^{-2}$, and $\varepsilon = 10^{-2}$.

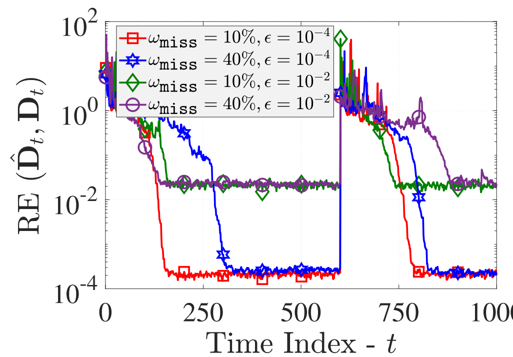


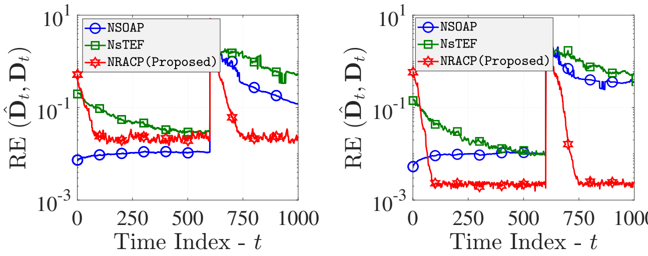
Fig. 8: Incomplete observations & time-varying scenarios: Performance of NRACP on a synthetic rank-5 tensor of size $50 \times 50 \times 50 \times 500$; $\sigma_n = 10^{-3}$, $A_{\texttt{outlier}} = 10$, $\omega_{\texttt{outlier}} = 10\%$.

NRACP converges. The results are shown in Fig. 8. We can see that the relative error between the estimation and ground truth converged to an error floor. Furthermore, the missing density $\omega_{\texttt{miss}}$ impacted only the convergence rate of NRACP. Specifically, the lower the missing density $\omega_{\texttt{miss}}$ was, the faster NRACP converged.

Next, we studied the robustness of NRACP against the noise variance in comparison with NSOAP [53] and NsTEF [62]. Since both algorithms are only feasible for third-order tensors without corruptions (outliers and missing values), we used a synthetic outlier-free tensor of size $50 \times 50 \times 1000$ and rank 5 for this task. The time-varying factor ϵ was set at 10^{-3}. Both NRACP and NsTEF used random initialization while the first 50 temporal slices were used to construct a good initialization tensor for NSOAP. Performance comparison results are illustrated in Fig. 9. At a low SNR, NSOAP provided a better estimation accuracy than NRACP and NsTEF. However, the proposed NRACP outperformed NSOAP and NsTEF at the high SNR, see Fig. 9(b). In the presence of abrupt changes, the convergence rate of NRACP was fast while NSOAP and NsTEF failed to track the change.

D. Real Datasets

To demonstrate the use of RACP with real-world datasets, we consider the following tasks: (i) tracking the online low-rank approximation of real-world data streams, (ii) multichannel EEG analysis, and (iii) video background modeling and fore-

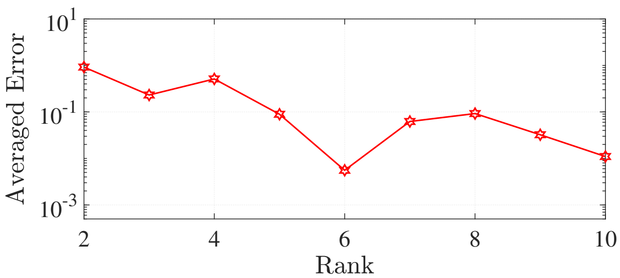


(a) $\sigma_n = 10^{-1}, \epsilon = 10^{-3}$ (b) $\sigma_n = 10^{-3}, \epsilon = 10^{-3}$

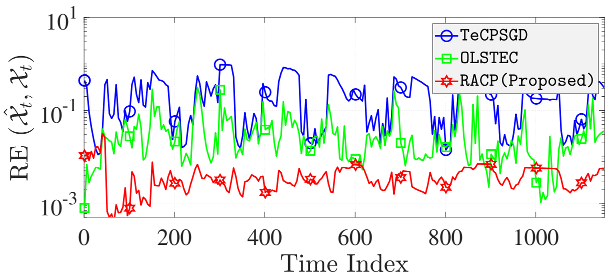
Fig. 9: Nonnegative adaptive CP decompositions: Outlier-free, full observations and an abrupt change at $t = 600$.

Dataset		Data size	Tasks
Intel Berkeley Lab		$54 \times 4 \times 1152$	Tracking the online low-rank approximation & online data completion
Internet Traffic		$12 \times 12 \times 48384$	
Taxi Trip Record		$265 \times 265 \times 3672$	
Video	Hall	$176 \times 144 \times 3584$	Background modeling & foreground detection
	Lobby	$128 \times 160 \times 1546$	
	Highway	$240 \times 320 \times 1700$	
EEG	ERPWAVELAB	$28 \times 64 \times 4392$	Multichannel EEG analysis & anomaly EEG detection
	Epileptic data	$19 \times 500 \times 6929$	

TABLE II: Real datasets under the study.

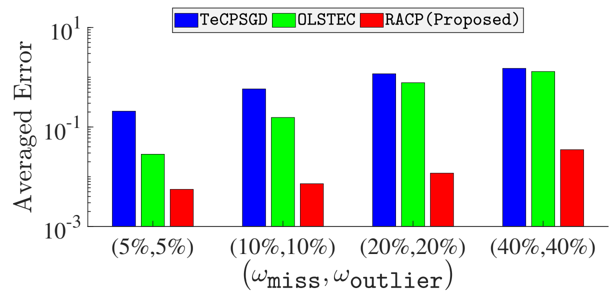


(a) Performance of RACP with different values of tensor rank

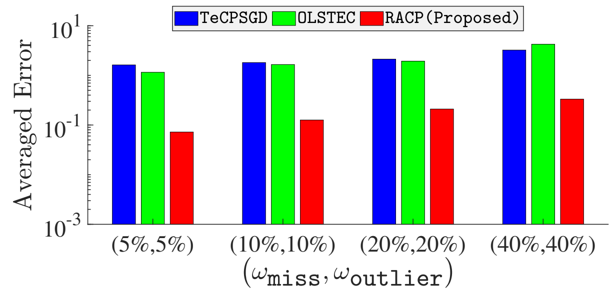


(b) Performance of adaptive CP algorithms with tensor rank $r = 6$

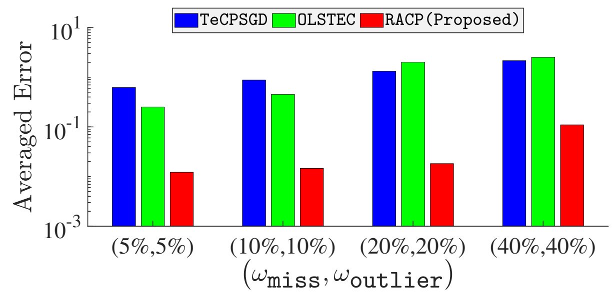
Fig. 10: Experimental results on the Intel Berkeley Lab data.



(a) Intel Berkeley Lab: Estimated rank $r = 6$



(b) Internet Traffic: Estimated rank $r = 3$



(c) Taxi Trip Record: Estimated rank $r = 8$

Fig. 11: Completion accuracy of adaptive CP algorithms on real-world data streams.

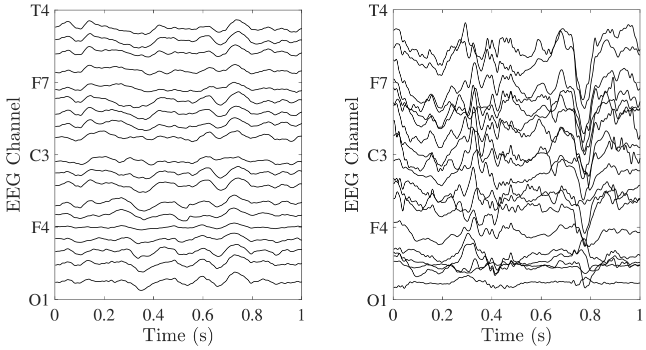
ground detection. See Tab. II for a summary of the real datasets used in this paper.

Task 1: Tracking the online low-rank approximation and online data completion

Datasets: In this task, we used three real datasets: Intel Berkeley Lab[6], Internet Traffic[7], and Taxi Trip Record[8]. The first dataset is a collection of timestamped topology information gathered from 54 positions (sensors) in the Intel Berkeley Research Lab. Specifically, these sensors collected: temperature (in degree Celsius), humidity (ranging from 0% to 100%), light (in Lux), and voltage (in volt, ranging from 2 to 3). Accordingly, we represent the sensor data by a three-order tensor of size $54 \times 4 \times 1152$ (i.e., *sensor* $\times$ *measurement* $\times$ *time*). The second dataset is the link traffic data which was collected from the Internet2 backbone network Abilene. The Abilene backbone is relatively small with 12 routers, 15 links, and 144 flow entries in each traffic matrix of size 12×12. We concatenated all these traffic matrices into a tensor of size $12 \times 12 \times 48384$. The third dataset describes yellow taxi trip records in the pairs of 265 pick-up and drop-off sites in New York. Each trip record contains several attributes, such as pick-up/drop-off times and locations, elapsed trip distance, rate type, and payment method. In this work, we specifically constructed a third-order tensor of size $265 \times 265 \times 3672$ (i.e., *origin* $\times$ *destination* $\times$ *time*).

Experiments & Results: Following the same experiment setup as in subsection V-A, data corruptions were generated as follows. The locations of missing entries and sparse outliers are randomly generated with probabilities $\omega_{\texttt{miss}}$ and $\omega_{\texttt{outlier}}$, respectively. Outlier values are drawn uniformly from the range $[0, \max(\boldsymbol{\mathcal{X}})]$ where $\max(\boldsymbol{\mathcal{X}})$ is the largest absolute

[6]Intel Berkeley Lab: http://db.csail.mit.edu/labdata/labdata.html
[7]Internet Traffic: https://roughan.info/project/traffic_matrix/
[8]Taxi Record: https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page



(a) Normal data (b) Complex partial seizure

Fig. 12: Epileptic EEG Dataset.

value in the underlying data $\boldsymbol{\mathcal{X}}$. In this experiment, we chose the value of $\omega_{\texttt{miss}}$ and $\omega_{\texttt{outlier}}$ among the range $\{5\%, 10\%, 20\%, 40\%\}$. As the true rank is unknown, we first varied its value from 2 to 10 and then chose the "best" one based on the averaged reconstruction error, see Fig. 10(a) for an example. We compared the performance of RACP against the two adaptive CP algorithms TeCPSGD [24] and OLSTEC [25]. Both algorithms are dependent on the forgetting factor λ, and its value was set at 0.98, 0.001, and 0.7, respectively. The penalty parameter μ was set at 1 for both TeCPSGD and OLSTEC. The experimental result in Fig. 11 indicates that RACP outperforms TeCPSGD and OLSTEC.

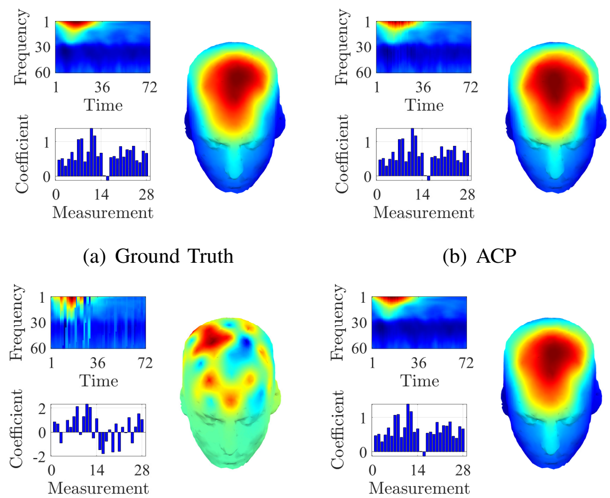
Task 2: Multichannel EEG Analysis

Datasets: In this task, we used two public electroencephalogram (EEG) datasets: ERPWAVELAB[9] and Epileptic EEG Data[10]. The former dataset contains wavelet-transformed versions of EEG signals that were collected from 14 subjects during the hand stimulation (i.e., proprioceptive pulls of the left and right hands) for inter-trial phase coherence analysis. In particular, these EEG signals were recorded using an electrode system of 64 channels with 28 measurements per subject. The continuous wavelet transform was then applied to represent these signals in the time-frequency domain. The latter dataset includes 20 EEG recordings of 6 patients diagnosed with epilepsy at the American university of Beirut medical center. The EEG data were recorded by using a system of 21 channels with a sampling rate of 500Hz. The dataset includes 3895 normal segments and 3850 abnormal segments in which there are 3034 partial seizures, 705 electrographic seizures, and 111 video-detected seizures with no visual change over EEG. Figs. 12(a) and 12(b) illustrate EEG normal waveforms and complex partial seizures. In what follows, we consider two common problems in multichannel EEG analysis: (i) incomplete multichannel EEG analysis from partial observations and (ii) anomaly EEG detection.

Incomplete Multichannel EEG Analysis: Here, we used the ERPWAVELAB dataset and followed the same experimental setup as in [27], [63], [64] to demonstrate the use of RACP with real EEG signals. We constructed an EEG tensor of size $28 \times 64 \times 4392$ (i.e., *measurement*$\times$*channel*$\times$*time-frequency*). To generate incomplete observations, signals from some channels at each time were randomly assumed to be missing. As suggested in [63], [64], we set the tensor rank at $r = 3$. The performance of RACP was compared with two adaptive CP algorithms NL-PETRELS [63] and ACP [27]. We fixed the forgetting factor λ at 0.999 and 0.5 for NL-PETRELS and ACP, respectively. As NL-PETRELS requires a warm start, we ran the batch CP-WOPT algorithm [64] with the first 1500 tensor slices, whereas random initialization was used for ACP and RACP. In this experiment, we aimed to factorize the EEG tensor into three basis components w.r.t. spatial domain, time-frequency domain, and measurement mode. As there is no real ground truth, we used the results (i.e., CP factors) derived from applying the batch CP-ALS algorithm to the EEG tensor with full observations as benchmarks. Experimental results are shown in Tab. III and Fig. 13. They indicate that RACP outperforms NL-PETRELS and provides a slightly better estimation than ACP, especially in the presence of highly incomplete observations (e.g., ≥ 40 channels are missing).

Missing channels	NL-PETRELS	ACP	RACP (Proposed)
1/64	0.051	0.063	0.056
10/64	0.062	0.025	0.023
20/64	0.077	0.011	0.014
30/64	0.121	0.097	0.086
40/64	0.891	0.132	0.119
50/64	1.325	1.137	0.982

TABLE III: Averaged errors of adaptive CP algorithms for multichannel EEG analysis from incomplete observations.



(a) Ground Truth (b) ACP

(c) NL-PETRELS (d) RACP (Proposed)

Fig. 13: First component of EEG factors when 40/60 EEG channels are missing.

Anomaly EEG Detection: We demonstrate the use of RACP to detect abnormal activities in the brain (i.e., epileptic seizures) with the epileptic EEG dataset. Here, we adopted a simple but effective way to predict abnormalities in multidimensional data streams [65], i.e. by modeling the abnormality of a tensor

[9] ERPWAVELAB: http://www.erpwavelab.org/

[10] Epileptic EEG Data: https://data.mendeley.com/datasets/5pc2j46cbc/1

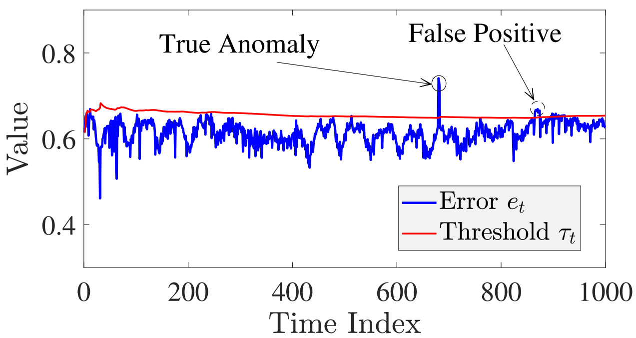


Fig. 14: The error e_t over time with $\alpha = 1.5$ and $L_t = t$. Normal data which are inaccurately labelled as abnormal are referred to as "false positive".

(streaming) slice $\mathcal{Y}_t$ by its recovery error

$$e_t = \left\| \mathcal{P}_t \circledast \left(\mathcal{Y}_t - \mathcal{Y}_t \prod_{n=1}^{N} \times_n \mathbf{U}_t^{(n)} \mathbf{U}_t^{(n)\#} \right) \right\|_F \Big/ \left\| \mathcal{Y}_t \right\|_F, \quad (48)$$

where $\{\mathbf{U}_t^{(n)}\}_{n=1}^N$ is the set of solutions generated by RACP at time t. It is also worth noting that the error e_t is relatively proportional to the norm of the outlier $\mathcal{O}_t$. We label $\mathcal{Y}_t$ based on the following rule

$$e_t \underset{\text{normal}}{\overset{\text{abnormal}}{\gtrless}} \tau_t = \text{mean}\left(\{e\}_{L_t}\right) + \alpha\,\text{std}\left(\{e\}_{L_t}\right), \quad (49)$$

where $\{e\}_{L_t}$ denotes the set of e_τ with $t - L_t < \tau \leq t$.

We followed the method in our companion work on epileptic spike detection [66] to obtain the time-frequency representation of multichannel EEG segments (including normal data and seizures), and hence the corresponding EEG tensors of size $19 \times 20 \times 500$ (i.e., *channel* $\times$ *scale* $\times$ *time*).[11] The resulting tensors were then concatenated into a huge tensor of which the last mode is being streamed. We used the first 100 tensors of normal data to obtain a warm start and the estimated rank of 9. Experimental results are shown in Fig. 14 (the error e_t over time) and Tab. IV (prediction accuracy versus the value of α). The results indicate that it is highly potential to detect anomalies in EEG signals by monitoring the approximation error. Subsequent investigations (e.g., type of wavelet, dominant scales, and mother function) are necessary to obtain a better prediction.

Task 3: Video background modeling & foreground detection

Datasets: Three real video sequences were used in this task, including Hall, Lobby, and Highway, see Fig. 15 for an illustration.[12] The Hall video is a set of 3584 images taken in an airport hall, and the image resolution is 176×144. This video shows a busy hall of an airport with many people coming in and out of the ground. The Lobby video contains 1546 images of 128×160 pixels. This image sequence was captured in an office lobby where background changes were specifically caused by switching on/off lights. The Highway video contains

[11] As indicated in the EEG dataset description report, the data of two channels Cz and Pz were omitted. Thus, we have 19 EEG channels left and each channel contains 500 samples. Also, 20 wavelet scales are chosen in the range $[4, 8]$.

[12] Video Sequences: http://jacarini.dinf.usherbrooke.ca.

Value of α	Sensitivity	Specificity	Accuracy
0.1	42.21%	53.02%	47.57%
0.5	59.74%	66.48%	63.09%
1	72.80%	74.38%	73.59%
1.5	81.58%	85.16%	83.36%
2	50.16%	53.54%	51.83%

TABLE IV: Anomaly EEG detection results. Sensitivity and specificity measure the percentage of abnormal and normal data detected correctly, respectively. Accuracy indicates the overall performance.

a) Hall b) Lobby c) Highway

Fig. 15: Three video sequences used in this paper.

1700 traffic images and each frame is of 240×320 pixels. It consists of two lanes of vehicles approaching the camera on a highway. These videos are naturally represented by three-way tensors of size *pixel* $\times$ *pixel* $\times$ *frame*. Accordingly, the adaptivity (streaming) can be done along the last dimension "frame".

Background Modeling: We first measured the video background modeling ability of RACP in comparison with a robust subspace tracking algorithm PETRELS-ADMM [49], and two adaptive CP algorithms (TeCPSGD [24] and OLSTEC [25]). These algorithmic parameters were kept as in task 1, except for the penalty parameter μ which was set at 0.1. The CP rank and subspace rank were set at 10. We consider the scenario where 50% of pixels were randomly assumed to be missing. Experimental results are illustrated in Fig. 16. It can be seen that the two robust algorithms PETRELS-ADMM and RACP were able to recover the video background, with the proposed RACP providing a slightly better estimation than PETRELS-ADMM. The two adaptive CP algorithms TeCPSGD and OLSTEC seem to fail when the video frame contains moving objects, probably because they do not account for sparse outliers.

Foreground Detection: Next, we investigated the ability of RACP in video foreground detection. We also compare the performance of RACP with three notable foreground detection algorithms, namely GRASTA [67], OSTD [34] and PETRELS-ADMM [49]. To have a fair comparison, the algorithm parameters were set by default as suggested by their authors. The penalty parameter ρ and constant step-size scale C were, respectively, set at 1.8 and 2 in GRASTA. The forgetting factor in PETRELS-ADMM is fixed at $\lambda = 0.98$, while OSTD is a parameter-free algorithm. It can be seen from Fig. 17 that the proposed RACP was capable of detecting moving objects in video streams and provided a competitive performance as compared to GRASTA, OSTD, and PETRELS-ADMM.

Fig. 16: Qualitative illustration of video background modeling results.

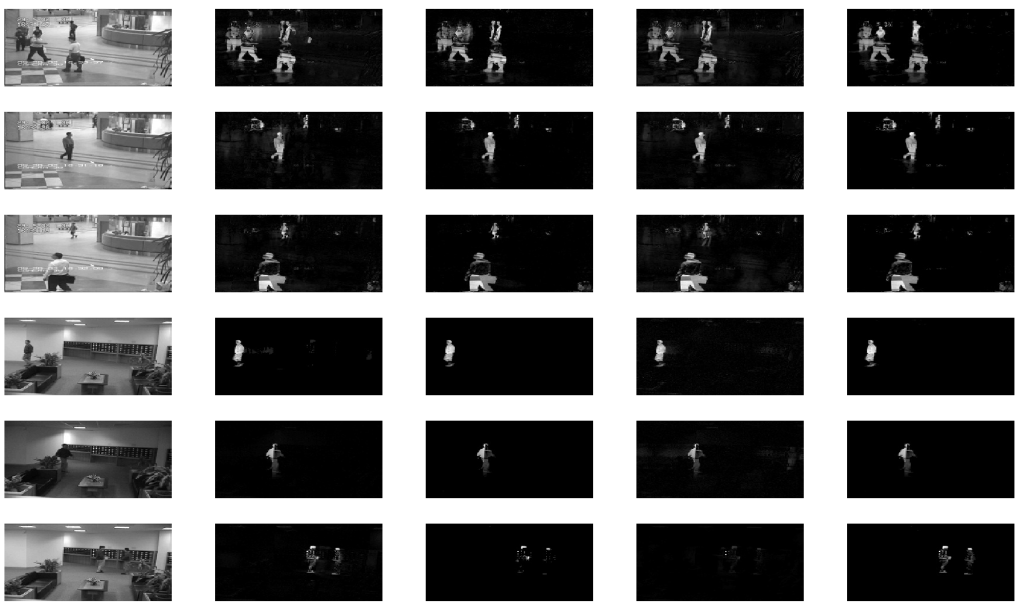


Fig. 17: Qualitative illustration of video foreground detection results.

VI. Conclusions

In this paper, we have addressed the problem of robust tensor tracking in the presence of both missing data and outliers. Under the CP/PARAFAC model, a novel robust adaptive CP decomposition called RACP has been proposed to track the low-rank approximation of streaming tensors from uncertain, noisy, and imperfect measurements. Its convergence analysis has been established to guarantee that the solution generated by RACP converges to a stationary point asymptotically. Experimental results indicate that RACP is capable of estimating the tensor factors as well as tracking their variations over time with high accuracy, and that RACP outperformed the state-of-the-art adaptive CP algorithms in both simulated and real data tests.

Appendix: Derivations of tensor factor tracking

The optimal solution of (18) can be derived by setting its derivative to zero

$$\begin{aligned}\sum_{k=t-L_t+1}^{t} & \lambda^{t-k}\big(\mathbf{W}_k^{(n)}\big)^\top \underline{\mathbf{P}}_{k,m}^{(n)}\big(\hat{\underline{\mathbf{x}}}_{k,m}^{(n)}\big)^\top \\ &= \sum_{k=t-L_t+1}^{t} \lambda^{t-k}\big(\mathbf{W}_k^{(n)}\big)^\top \underline{\mathbf{P}}_{k,m}^{(n)} \mathbf{W}_k^{(n)}\big(\mathbf{u}_m^{(n)}\big)^\top.\end{aligned} \tag{50}$$

Instead of solving (50) directly, we propose a more elegant recursive way to obtain $\mathbf{u}_{t,m}^{(n)}$ as follows.

First, let us denote the left hand side of (50) by $\mathbf{d}_{t,m}^{(n)}$, and the expression $\sum_{k=t-L_t+1}^{t} \lambda^{t-k}\big(\mathbf{W}_k^{(n)}\big)^\top \underline{\mathbf{P}}_{k,m}^{(n)} \mathbf{W}_k^{(n)}$ by $\mathbf{S}_{t,m}^{(n)}$.

Accordingly, (50) becomes

$$\mathbf{S}_{t,m}^{(n)}\big(\mathbf{u}_{t,m}^{(n)}\big)^\top = \mathbf{d}_{t,m}^{(n)}. \tag{51}$$

Interestingly, both $\mathbf{d}_{t,m}^{(n)}$ and $\mathbf{S}_{t,m}^{(n)}$ can be updated recursively:

$$\mathbf{d}_{t,m}^{(n)} = \lambda \mathbf{d}_{t-1,m}^{(n)} + \big(\widetilde{\mathbf{W}}_t^{(n)}\big)^\top \underline{\widetilde{\mathbf{P}}}_{t,m}^{(n)}\big(\underline{\tilde{\mathbf{x}}}_{t,m}^{(n)}\big)^\top, \tag{52}$$

$$\mathbf{S}_{t,m}^{(n)} = \lambda \mathbf{S}_{t-1,m}^{(n)} + \big(\widetilde{\mathbf{W}}_t^{(n)}\big)^\top \underline{\widetilde{\mathbf{P}}}_{t,m}^{(n)}\widetilde{\mathbf{W}}_t^{(n)}. \tag{53}$$

where

$$\widetilde{\mathbf{W}}_t^{(n)} = \big[\big(\mathbf{W}_t^{(n)}\big)^\top \;\; \big(\mathbf{W}_{t-L_t}^{(n)}\big)^\top\big]^\top, \tag{54}$$

$$\underline{\tilde{\mathbf{x}}}_{t,m}^{(n)} = \big[\underline{\hat{\mathbf{x}}}_{t,m}^{(n)} \;\; \underline{\hat{\mathbf{x}}}_{t-L_t,m}^{(n)}\big], \tag{55}$$

$$\underline{\widetilde{\mathbf{P}}}_{t,m}^{(n)} = \begin{bmatrix} \underline{\mathbf{P}}_{t,m}^{(n)} & \mathbf{0} \\ \mathbf{0} & -\lambda^{L_t}\underline{\mathbf{P}}_{t-L_t,m}^{(n)} \end{bmatrix}. \tag{56}$$

Therefore, we can rewrite (51) as

$$\begin{aligned}
\mathbf{S}_{t,m}^{(n)}\big(\mathbf{u}_{t,m}^{(n)}\big)^\top &= \lambda \mathbf{d}_{t-1,m}^{(n)} + \big(\widetilde{\mathbf{W}}_t^{(n)}\big)^\top \underline{\widetilde{\mathbf{P}}}_{t,m}^{(n)}\;\big(\underline{\tilde{\mathbf{x}}}_{t,m}^{(n)}\big)^\top \\
&= \lambda \mathbf{S}_{t-1,m}^{(n)}\big(\mathbf{u}_{t-1,m}^{(n)}\big)^\top + \big(\widetilde{\mathbf{W}}_t^{(n)}\big)^\top \underline{\widetilde{\mathbf{P}}}_{t,m}^{(n)}\;\big(\underline{\tilde{\mathbf{x}}}_{t,m}^{(n)}\big)^\top \\
&= \mathbf{S}_{t,m}^{(n)}\big(\mathbf{u}_{t-1,m}^{(n)}\big)^\top + \big(\widetilde{\mathbf{W}}_t^{(n)}\big)^\top \underline{\widetilde{\mathbf{P}}}_{t,m}^{(n)}\Big(\big(\underline{\tilde{\mathbf{x}}}_{t,m}^{(n)}\big)^\top - \widetilde{\mathbf{W}}_t^{(n)}\big(\mathbf{u}_{t-1,m}^{(n)}\big)^\top\Big).
\end{aligned}$$

Multiplying both sides by $\big(\mathbf{S}_{t,m}^{(n)}\big)^{-1}$ results in

$$\mathbf{u}_{t,m}^{(n)} = \mathbf{u}_{t-1,m}^{(n)} + \big(\boldsymbol{\delta}\underline{\tilde{\mathbf{x}}}_{t,m}^{(n)}\big)^\top\big(\mathbf{V}_{t,m}^{(n)}\big)^\top, \tag{57}$$

where

$$\boldsymbol{\delta}\underline{\tilde{\mathbf{x}}}_{t,m}^{(n)} = \underline{\widetilde{\mathbf{P}}}_{t,m}^{(n)}\Big(\big(\underline{\tilde{\mathbf{x}}}_{t,m}^{(n)}\big)^\top - \widetilde{\mathbf{W}}_t^{(n)}\big(\mathbf{u}_{t-1,m}^{(n)}\big)^\top\Big), \tag{58a}$$

$$\mathbf{V}_{t,m}^{(n)} = \big(\mathbf{S}_{t,m}^{(n)}\big)^{-1}\big(\widetilde{\mathbf{W}}_t^{(n)}\big)^\top. \tag{58b}$$

Collecting all rows $\mathbf{u}_{t,m}^{(n)}$ together, $m = 1, 2, \ldots, I_n$, a simplified version of (57) for updating the whole $\mathbf{U}_t^{(n)}$ can be given by[13]

$$\mathbf{U}_t^{(n)} = \mathbf{U}_{t-1}^{(n)} + \boldsymbol{\Delta}\widetilde{\mathbf{X}}_t^{(n)}\big(\mathbf{V}_t^{(n)}\big)^\top. \tag{59}$$

Here, the error matrix $\boldsymbol{\Delta}\widetilde{\mathbf{X}}_t^{(n)} \in \mathbb{R}^{I_n\times 2J_n}$ with $J_n = \prod_{i\neq n} I_i$ is defined as

$$\boldsymbol{\Delta}\widetilde{\mathbf{X}}_t^{(n)} = \underline{\widetilde{\mathbf{P}}}_t^{(n)} \circledast \Big(\widetilde{\mathbf{X}}_t^{(n)} - \mathbf{U}_{t-1}^{(n)}\big(\widetilde{\mathbf{W}}_t^{(n)}\big)^\top\Big), \tag{60}$$

with $\widetilde{\mathbf{X}}_t^{(n)} = \big[\widehat{\mathbf{X}}_t^{(n)} \;\; \widehat{\mathbf{X}}_{t-L_t,m}^{(n)}\big] \in \mathbb{R}^{I_n\times 2J_n}$ and the coefficient matrix $\mathbf{V}_t^{(n)} \in \mathbb{R}^{r\times 2J_n}$ is computed as

$$\mathbf{V}_t^{(n)} = \big(\mathbf{S}_t^{(n)}\big)^{-1}\big(\widetilde{\mathbf{W}}_t^{(n)}\big)^\top, \tag{61}$$

where the matrix $\mathbf{S}_t^{(n)} \in \mathbb{R}^{r\times r}$ is recursively updated as follows

$$\mathbf{S}_t^{(n)} = \lambda \mathbf{S}_{t-1}^{(n)} + \big(\widetilde{\mathbf{W}}_t^{(n)}\big)^\top \widetilde{\mathbf{W}}_t^{(n)}. \tag{62}$$

In this way, we can skip several operations and save a memory storage of $\mathcal{O}\big(\sum_{n=1}^N (I_n - 1)(I_n r + r^2)\big)$. Specifically, the cost of computing (62) is $\mathcal{O}\big(r^2\prod_{i=1,i\neq n}^N I_i\big)$ flops. The computation of (61) also requires a cost of $\mathcal{O}\big(r^2\prod_{i=1,i\neq n}^N I_i\big)$ flops because $\mathbf{S}_t^{(n)}$ is of size $r\times r$ and its inverse computation is not expensive and independent of the tensor dimension. The error matrix $\boldsymbol{\Delta}\hat{\mathbf{X}}_t^{(n)}$ in (60) can be derived from Step 1 by reshaping the residual vector $\mathbf{P}_t(\mathbf{x}_t - \mathbf{o}_t - \mathbf{H}_{t-1}\mathbf{u}_t)$. The most expensive step is the product $\boldsymbol{\Delta}\widetilde{\mathbf{X}}_t^{(n)}\big(\mathbf{V}_t^{(n)}\big)^\top$ which costs $r\prod_{i=1}^N I_i$ flops while the addition operator in (59) requires only rI_n flops. Therefore, the overall cost of updating $\mathbf{U}_t^{(n)}$ in a naive way is $\mathcal{O}\big(r\prod_{i=1}^N I_i\big)$ flops. Note that $\boldsymbol{\Delta}\widetilde{\mathbf{X}}_t^{(n)}\big(\mathbf{V}_t^{(n)}\big)^\top$ can be divided into two parts $\mathbf{Z}_t^{(n)} = \boldsymbol{\Delta}\widetilde{\mathbf{X}}_t^{(n)}\widetilde{\mathbf{W}}_t^{(n)}$ and $\mathbf{Z}_t^{(n)}\big(\mathbf{S}_t^{(n)}\big)^{-\top}$. Here, $\boldsymbol{\Delta}\widetilde{\mathbf{X}}_t^{(n)}\widetilde{\mathbf{W}}_t^{(n)}$ can be referred to as "matricized tensor times Khatri-Rao product" (MTTKRP) [68], [69]. Fortunately, Phan *et al.* in [69] proposed a clever reorganization of MTTKRP which can accelerate the computation and reduce the overall cost of (59) to $\mathcal{O}\big(r^2\prod_{i=1,i\neq n}^N I_i\big)$ flops. We detail the derivation of the simplified version of (59) in the supplementary document (Appendix A).

[13]To enable the recursive rules of (57) and (59), $\mathbf{S}_{0,m}^{(n)}$ and $\mathbf{S}_0^{(n)}$ can be initialized by $\delta\mathbf{I}_r$ where $\delta > 0$, for $n = 1, 2, \ldots, N$.

REFERENCES

[1] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," *SIAM Rev.*, vol. 51, no. 3, pp. 455–500, 2009.

[2] E. Acar and B. Yener, "Unsupervised multiway data analysis: A literature survey," *IEEE Trans Knowl. Data Eng.*, vol. 21, no. 1, pp. 6–20, 2008.

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

[4] N. D. Sidiropoulos, L. D. Lathauwer, X. Fu, K. Huang, E. E. Papalexakis, and C. Faloutsos, "Tensor decomposition for signal processing and machine learning," *IEEE Trans. Signal Process.*, vol. 65, no. 13, pp. 3551–3582, 2017.

[5] R. Bro, "PARAFAC. Tutorial and applications," *Chemometr. Intell. Lab. Syst.*, vol. 38, no. 2, pp. 149–172, 1997.

[6] D. Nion and N. D. Sidiropoulos, "Tensor algebra and multidimensional harmonic retrieval in signal processing for MIMO radar," *IEEE Trans. Signal Process.*, vol. 58, no. 11, pp. 5693–5705, 2010.

[7] Z. Zhou, J. Fang, L. Yang, H. Li, Z. Chen, and R. S. Blum, "Low-rank tensor decomposition-aided channel estimation for millimeter wave MIMO-OFDM systems," *IEEE J. Sel. Areas Commun.*, vol. 35, no. 7, pp. 1524–1538, 2017.

[8] E. Kofidis, "A tensor-based approach to joint channel estimation/data detection in flexible multicarrier MIMO systems," *IEEE Trans. Signal Process.*, vol. 68, pp. 3179–3193, 2020.

[9] E. Acar, A.-B. Canan, H. Bingol, R. Bro, and B. Yener, "Multiway analysis of epilepsy tensors," *Bioinformatics*, vol. 23, no. 13, pp. i10–i18, 2007.

[10] C.-F. V. Latchoumane, F.-B. Vialatte, J. Solé-Casals, M. Maurice, S. R. Wimalaratna, N. Hudson, J. Jeong, and A. Cichocki, "Multiway array decomposition analysis of EEGs in Alzheimer's disease," *J. Neurosci. Methods*, vol. 207, no. 1, pp. 41–50, 2012.

[11] M. De Vos, A. Vergult, L. De Lathauwer, W. De Clercq, S. Van Huffel, P. Dupont, A. Palmini, and W. Van Paesschen, "Canonical decomposition of ictal scalp EEG reliably detects the seizure onset zone," *NeuroImage*, vol. 37, no. 3, pp. 844–854, 2007.

[12] V. A. Miguel, J. E. Cohen, R. Cabral Farias, J. Chanussot, and P. Comon, "Nonnegative tensor CP decomposition of hyperspectral data," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 5, pp. 2577–2588, 2016.

[13] X. Liu, S. Bourennane, and C. Fossati, "Denoising of hyperspectral images using the PARAFAC model and statistical performance analysis," *IEEE Trans. Geosci. Remote Sens.*, vol. 50, no. 10, pp. 3717–3724, 2012.

[14] J. Xue, Y. Zhao, W. Liao, and J. C.-W. Chan, "Nonlocal low-rank regularized tensor decomposition for hyperspectral image denoising," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 7, pp. 5174–5189, 2019.

[15] J. Qiu, Q. Wu, G. Ding, Y. Xu, and S. Feng, "A survey of machine learning for big data processing," *EURASIP J. Adv. Signal Process.*, vol. 2016, no. 1, pp. 1–16, 2016.

[16] T. Akidau, S. Chernyak, and R. Lax, *Streaming systems: the what, where, when, and how of large-scale data processing*. O'Reilly Media, 2018.

[17] T. Kolajo, O. Daramola, and A. Adebiyi, "Big data stream analysis: A systematic literature review," *J. Big Data*, vol. 6, no. 1, pp. 1–30, 2019.

[18] E. E. Papalexakis, U. Kang, C. Faloutsos, N. D. Sidiropoulos, and A. Harpale, "Large scale tensor decompositions: Algorithmic developments and applications." *IEEE Data Eng. Bull.*, vol. 36, no. 3, pp. 59–66, 2013.

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

[20] M. Vandecappelle, N. Vervliet, and L. De Lathauwer, "Nonlinear least squares updating of the canonical polyadic decomposition," in *Eur. Signal Process. Conf.*, 2017, pp. 663–667.

[21] S. Zhou, N. X. Vinh, J. Bailey, Y. Jia, and I. Davidson, "Accelerating online CP decompositions for higher order tensors," in *ACM Int. Conf. Knowl. Discover. Data Min.*, 2016, pp. 1375–1384.

[22] S. Smith, K. Huang, N. D. Sidiropoulos, and G. Karypis, "Streaming tensor factorization for infinite data sources," in *SIAM Int. Conf. Data Min.*, 2018, pp. 81–89.

[23] S. Rambhatla, X. Li, and J. Haupt, "Provable online CP/PARAFAC decomposition of a structured tensor via dictionary learning," in *Adv. Neural Inf. Process. Syst.*, 2020, pp. 1–12.

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

[25] H. Kasai, "Fast online low-rank tensor subspace tracking by CP decomposition using recursive least squares from incomplete observations," *Neurocomput.*, vol. 347, pp. 177–190, 2019.

[26] L. T. Thanh, K. Abed-Meraim, N. L. Trung, and A. Hafiane, "A fast randomized adaptive CP decomposition for streaming tensors," in *IEEE Int. Conf. Acoust. Speech Signal Process.*, 2021, pp. 2910–2914.

[27] ——, "Tracking dynamic low-rank approximations of incomplete high-order streaming tensors," *Techrxiv*, 2022.

[28] Z. Zhang and C. Hawkins, "Variational bayesian inference for robust streaming tensor factorization and completion," in *IEEE Int. Conf. Data Min.*, 2018, pp. 1446–1451.

[29] M. Najafi, L. He, and S. Y. Philip, "Outlier-robust multi-aspect streaming tensor completion and factorization." in *IJCAI Int. Joint Conf. Artif. Intell.*, 2019, pp. 3187–3194.

[30] L. Dongjin and S. Kijung, "Robust factorization of real-world tensor streams with patterns, missing values, and outliers," in *IEEE Int. Conf. Data Eng.*, 2021, pp. 840–851.

[31] C. C. Holt, "Forecasting seasonals and trends by exponentially weighted moving averages," *Int. J. Forecast.*, vol. 20, no. 1, pp. 5–10, 2004.

[32] W. Hu, X. Li, X. Zhang, X. Shi, S. Maybank, and Z. Zhang, "Incremental tensor subspace learning and its applications to foreground segmentation and tracking," *Int. J. Comput. Vis.*, vol. 91, no. 3, pp. 303–327, 2011.

[33] J. Li, G. Han, J. Wen, and X. Gao, "Robust tensor subspace learning for anomaly detection," *Int. J. Mach. Learn. Cybern.*, vol. 2, no. 2, pp. 89–98, 2011.

[34] A. Sobral, S. Javed, S. K. Jung, T. Bouwmans, and E. Zahzah, "Online stochastic tensor decomposition for background subtraction in multispectral video sequences," in *IEEE Int. Conf. Comput. Vis.*, 2015, pp. 946–953.

[35] I. Kajo, N. Kamel, and Y. Ruichek, "Incremental tensor-based completion method for detection of stationary foreground objects," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 29, no. 5, pp. 1325–1338, 2019.

[36] P. Li, J. Feng, X. Jin, L. Zhang, X. Xu, and S. Yan, "Online robust low-rank tensor modeling for streaming data analysis," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 4, pp. 1061–1075, 2019.

[37] D. G. Chachlakis, M. Dhanaraj, A. Prater-Bennette, and P. P. Markopoulos, "Dynamic L1-norm Tucker tensor decomposition," *IEEE J. Sel. Topics Signal Process.*, vol. 15, no. 3, pp. 587–602, 2021.

[38] D. Goldfarb and Z. Qin, "Robust low-rank tensor recovery: Models and algorithms," *SIAM J. Matrix Anal. Appl.*, vol. 35, no. 1, pp. 225–253, 2014.

[39] Q. Gu, H. Gui, and J. Han, "Robust tensor decomposition with gross corruption," in *Adv. Neural Inf. Process. Syst.*, 2014, pp. 1422–1430.

[40] B. Huang, C. Mu, D. Goldfarb, and J. Wright, "Provable models for robust low-rank tensor completion," *Pac. J. Optim.*, vol. 11, no. 2, pp. 339–364, 2015.

[41] A. Wang, Z. Jin, and G. Tang, "Robust tensor decomposition via t-SVD: Near-optimal statistical guarantee and scalable algorithms," *Signal Process.*, vol. 167, p. 107319, 2020.

[42] J. M. Ortega and W. C. Rheinboldt, *Iterative Solution of Nonlinear Equations in Several Variables*, 1970.

[43] J. Mairal, "Incremental majorization-minimization optimization with application to large-scale machine learning," *SIAM J. Optim.*, vol. 25, no. 2, pp. 829–855, 2015.

[44] Y. Sun, P. Babu, and D. P. Palomar, "Majorization-minimization algorithms in signal processing, communications, and machine learning," *IEEE Trans. Signal Process.*, vol. 65, no. 3, pp. 794–816, 2016.

[45] A. Tulay and H. Simon, *Adaptive Signal Processing: Next Generation Solutions*, 2010.

[46] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online learning for matrix factorization and sparse coding," *J. Mach. Learn. Res.*, vol. 11, no. Jan, pp. 19–60, 2010.

[47] N. Guan, D. Tao, Z. Luo, and B. Yuan, "Online nonnegative matrix factorization with robust stochastic approximation," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 7, pp. 1087–1099, 2012.

[48] J. Feng, H. Xu, and S. Yan, "Online robust PCA via stochastic optimization," in *Adv. Neural Inf. Process. Syst.*, 2013, pp. 404–412.

[49] L. T. Thanh, N. V. Dung, N. L. Trung, and K. Abed-Meraim, "Robust subspace tracking with missing data and outliers: Novel algorithm with convergence guarantee," *IEEE Trans. Signal Process.*, vol. 69, pp. 2070–2085, 2021.

[50] 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, 2011.

[51] J. S. Simonoff, *Smoothing Methods in Statistics*, 2012.

[52] 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*, 2009.

[53] N. V. Dung, K. Abed-Meraim, and N. L. Trung, "Second-order optimization based adaptive PARAFAC decomposition of three-way tensors," *Digit. Signal Process.*, vol. 63, pp. 100–111, 2017.

[54] D. Chen and R. J. Plemmons, "Nonnegativity constraints in numerical analysis," in *The Birth of Numerical Analysis*, 2009, pp. 109–139.

[55] C. L. Lawson and R. J. Hanson, *Solving Least Squares Problems*, 1995.

[56] N. Parikh and S. Boyd, "Proximal algorithms," *Found. Trends Opt.*, vol. 1, no. 3, pp. 127–239, 2014.

[57] S. Chatterjee, "A deterministic theory of low rank matrix completion," *IEEE Trans. Inf. Theory*, vol. 66, no. 12, pp. 8046–8055, 2020.

[58] D. L. Pimentel-Alarcon, N. Boston, and R. D. Nowak, "A characterization of deterministic sampling patterns for low-rank matrix completion," *IEEE J. Sel. Topics Signal Process.*, vol. 10, no. 4, pp. 623–636, 2016.

[59] M. Ashraphijuo and X. Wang, "Fundamental conditions for low-CP-rank tensor completion," *J. Mach. Learn. Res.*, vol. 18, no. 1, pp. 2116–2145, 2017.

[60] M. Ashraphijuo, V. Aggarwal, and X. Wang, "Deterministic and probabilistic conditions for finite completability of low-Tucker-Rank tensor," *IEEE Trans. Inf. Theory*, vol. 65, no. 9, pp. 5380–5400, 2019.

[61] M. Metivier, *Semimartingales: A Course on Stochastic Processes*, 1984.

[62] V. Vigneron, A. Kodewitz, M. N. da Costa, A. M. Tome, and E. Langlang, "Non-negative sub-tensor ensemble factorization (NsTEF) algorithm. A new incremental tensor factorization for large datasets." *Signal Process.*, vol. 144, pp. 77–86, 2018.

[63] N. L. Trung, T. M. Chinh, N. V. Dung, and K. Abed-Meraim, "A nonlinear tensor tracking algorithm for analysis of incomplete multi-channel EEG data," in *Int. Symp. Med. Inf. Commun. Technol.*, 2018, pp. 1–6.

[64] E. Acar, D. M. Dunlavy, T. G. Kolda, and M. Morup, "Scalable tensor factorizations for incomplete data," *Chemometr. Intell. Lab. Syst.*, vol. 106, no. 1, pp. 41–56, 2011.

[65] J. Sun, D. Tao, S. Papadimitriou, P. S. Yu, and C. Faloutsos, "Incremental tensor analysis: Theory and applications," *ACM Trans. Knowl. Discov. Data*, vol. 2, no. 3, pp. 1–37, 2008.

[66] L. T. Thanh, N. T. A. Dao, N. V. Dung, N. L. Trung, and K. Abed-Meraim, "Multi-channel EEG epileptic spike detection by a new method of tensor decomposition," *J. Neural Eng.*, vol. 17, no. 1, p. 016023, 2020.

[67] J. He, L. Balzano, and A. Szlam, "Incremental gradient on the Grassmannian for online foreground and background separation in subsampled video," in *IEEE Conf. Comput. Vis. Pattern Recogn.*, 2012, pp. 1568–1575.

[68] B. W. Bader and T. G. Kolda, "Efficient MATLAB computations with sparse and factored tensors," *SIAM J. Sci. Comput.*, vol. 30, no. 1, pp. 205–231, 2008.

[69] A.-H. Phan, P. Tichavsky, 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, 2013.