Yokohama Publishers
ISSN 1349-8169 ONLINE JOURNAL

Pacific Journal of Optimization
Vol. 17, No. 3, pp. 433–452, 2021

A MOMENTUM BLOCK-RANDOMIZED STOCHASTIC ALGORITHM FOR LOW-RANK TENSOR CP DECOMPOSITION*

Qingsong Wang, Chunfeng Cui† and Deren Han

Abstract: The block-randomized stochastic algorithm has shown its power in handling high-dimensional low-rank tensor canonical polyadic decomposition (CPD). Since computing CPD is computationally expensive, there is great interest in speeding up the convergence. In this paper, we introduce a momentum accelerated version of the block-randomized stochastic gradient descent (SGD) algorithm for low-rank tensor CPD. Both the constant stepsize version and the adaptive stepsize version are considered. Under some mild conditions, we show the global convergence to the stationary point of the constant stepsize algorithm for this optimization problem. Compared with the algorithms without momentum, the preliminary numerical experiments for the synthetic and real data demonstrate that our accelerated algorithms are efficient, and can achieve better performance in terms of objection function value, mean squared error, and structural similarity value.

Key words: *momentum, stochastic gradient, canonical polyadic decomposition, adaptive stepsize*

Mathematics Subject Classification: *15A72, 90C26, 90C30*

1 Introduction

Tensor decomposition is the higher-order analogue of matrix decomposition and is becoming an important tool for data analysis. There are many common tensor decomposition methods, e.g., canonical polyadic decomposition (CPD), Tucker decomposition, tensor-train decomposition, tensor ring decomposition. The CPD is one of the most widely used low-rank tensor decomposition model, and has found many applications in modern research fields, such as topic modeling [1, 29], human brain study [7, 8], hyperspectral imaging [28, 34], medical data analysis [15, 31] and etc.

We consider an Nth order tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ with $\boldsymbol{\mathcal{X}}(i_1, \ldots, i_N)$ denotes its $(i_1, \ldots, i_N)$-th element of the tensor $\boldsymbol{\mathcal{X}}$. The tensor $\boldsymbol{\mathcal{X}}$ can be represented as the sum of R rank-one components:

$$\boldsymbol{\mathcal{X}} = \sum_{r=1}^{R} \boldsymbol{A}_{(1)}(:, r) \circ \boldsymbol{A}_{(2)}(:, r) \circ \cdots \circ \boldsymbol{A}_{(N)}(:, r), \tag{1.1}$$

*This research is supported by the National Natural Science Foundation of China (NSFC) grants 11625105 and 11926358

†Corresponding Author.

© 2021 Yokohama Publishers

where $\circ$ denotes the outer product of vectors, $\boldsymbol{A}_{(n)}(n = 1, \ldots, N)$ is an I_n-by-R matrix denoting the mode n latent factor, and the rth factor (column) in mode n is denoted by $\boldsymbol{A}_{(n)}(:, r)$. Each component of $\boldsymbol{\mathcal{X}}$ is an N-way outer product of N factors, i.e.,

$$\boldsymbol{\mathcal{X}}(i_1, \ldots, i_N) = \sum_{r=1}^{R} \prod_{n=1}^{N} \boldsymbol{A}_{(n)}(i_n, r),$$

for $i_n \in \{1, \ldots, I_n\}$. We have $\text{rank}(\boldsymbol{\mathcal{X}}) \leq R$ when $\boldsymbol{\mathcal{X}}$ can be written as the sum of R rank-one tensors. If there exists a minimal integer R such that the expression in (1.1) satisfied, the right side of (1.1) is called the CPD of the tensor $\boldsymbol{\mathcal{X}}$.

Assume that $\text{rank}(\boldsymbol{\mathcal{X}}) = R$, then the CPD of a tensor can be obtained via minimizing a certain optimization problem with $\boldsymbol{A}_{(n)} \in \mathbb{R}^{I_n \times R}(n = 1, \ldots, N)$ as follows

$$\min_{\{\boldsymbol{A}_{(n)}\}_{n=1}^{N}} \quad f(\boldsymbol{A}_{(1)}, \ldots, \boldsymbol{A}_{(N)}), \tag{1.2}$$

where f is the loss function. A common optimization criterion for CPD is the least squares (LS) fitting criterion [14, 35] which is defined as

$$f\left(\boldsymbol{A}_{(1)}, \ldots, \boldsymbol{A}_{(N)}\right) := \frac{1}{2} \left\| \boldsymbol{\mathcal{X}} - \sum_{r=1}^{R} \boldsymbol{A}_{(1)}(:, r) \circ \ldots \circ \boldsymbol{A}_{(N)}(:, r) \right\|_F^2. \tag{1.3}$$

The problem (1.2) with LS fitting criterion (1.3) will be "ill-posed" in the sense as follows [19]:

1. The loss function (1.3) has continuously many local minima because of the indeterminacy of scaling.
2. From [6] and [9], we know that the loss function (1.3) may not have a global minimum bacause the domain is non-compact.

To solve these issues, the regularization terms for $\boldsymbol{A}_{(n)}$ with $n = 1, \ldots, N$ are considered. Then the objective function is reformulated as

$$\min_{\{\boldsymbol{A}_{(n)}\}_{n=1}^{N}} \quad \Phi(\boldsymbol{A}_{(1)}, \ldots, \boldsymbol{A}_{(N)}) := f(\boldsymbol{A}_{(1)}, \ldots, \boldsymbol{A}_{(N)}) + \sum_{n=1}^{N} h_n(\boldsymbol{A}_{(n)}), \tag{1.4}$$

where $h_n(\boldsymbol{A}_{(n)})$ denotes a structure promoting regularizer on $\boldsymbol{A}_{(n)}$, such as nonnegative, sparse or low rank terms for $\boldsymbol{A}_{(n)}$. For example, if $\boldsymbol{A}_{(n)} \in \mathcal{A}_n := \{\boldsymbol{A}_{(n)} | \boldsymbol{A}_{(n)} \geq \boldsymbol{0}\}$ is applied, we can write $h_n(\cdot)$ as the indicator function of set $\mathcal{A}_n$:

$$h_n(\boldsymbol{A}) = \mathcal{I}\left(\mathcal{A}_n\right) = \begin{cases} 0, & \boldsymbol{A} \in \mathcal{A}_n, \\ \infty, & \text{otherwise.} \end{cases} \tag{1.5}$$

The matrix unfolding of a tensor has proven very useful in many tensor factorization algorithms. The mode-n unfolding of a tensor $\boldsymbol{\mathcal{X}}$ is a J_n-by-I_n matrix ($J_n = I_1 \times \cdots \times I_{n-1} \times I_{n+1} \times \cdots \times I_N$) which is defined as

$$\boldsymbol{X}_{(n)}(j, i_n) = \boldsymbol{\mathcal{X}}(i_1, \ldots, i_N),$$

where $j = 1 + \sum_{k=1,k\neq n}^{N}(i_k - 1)\bar{J}_k$ with $\bar{J}_k = \prod_{m=1,m\neq n}^{k-1} I_m$ [17]. Then the CPD representation of (1.1) can be rewritten as

$$\boldsymbol{X}_{(n)} = \boldsymbol{H}_{(n)}\boldsymbol{A}_{(n)}^T,$$

where $\boldsymbol{H}_{(n)} \in \mathbb{R}^{J_n \times R}$ is defined as

$$\boldsymbol{H}_{(n)} = \boldsymbol{A}_{(N)} \odot \ldots \odot \boldsymbol{A}_{(n+1)} \odot \boldsymbol{A}_{(n-1)} \odot \ldots \odot \boldsymbol{A}_{(1)}.$$

Here, $\odot$ denotes the Khatri-Rao product of matrices. There are many algorithms can be used to solve the CPD problem, such as [14, 21, 23, 35]. Many of these algorithms leverage data sparsity to scale up CPD by using the zero elements in large scale tensor. The alternating least square (ALS) method can significantly simplify the cost of computation. Specifically, ALS solves the following subproblem cyclically for $n = 1, ..., N$:

$$\boldsymbol{A}_{(n)} = \arg\min_{\boldsymbol{A}} \quad \frac{1}{2}\left\|\boldsymbol{X}_{(n)} - \boldsymbol{H}_{(n)}\boldsymbol{A}^T\right\|_F^2. \tag{1.6}$$

When the value of I_n $(n = 1, \ldots, N)$ above is large, computing the gradient of the loss function $f(\boldsymbol{A}_{(1)}, \ldots, \boldsymbol{A}_{(N)})$ is often prohibitively expensive, rendering most traditional deterministic first-order optimization algorithms ineffective. Over the years, randomized optimization algorithms [5, 27] have become increasingly popular due to their efficiency and simplicity. A number of stochastic optimization based CPD algorithms have been proposed in the literature [2, 4, 30]. We apply the random sampling rule to the tensor data and use the sampled piece to update the latent factors. A set of mode-n fibers for a certain n at each iteration are sampled [2]. A direct way is to solve the least squares subproblems (1.6) for all the modes in [2] are exactly following a Gauss-Seidel manner in each iteration. But this exact method is time-consuming when the dimension of subproblem (1.6) is large.

Recently, the work in [11] proposed a new stochastic algorithmic framework for computing the CPD of large-scale dense tensors, i.e., the Block-Randomized SGD (BrasCPD). The BrasCPD algorithm is a combination of randomized block coordinate descent (BCD) [3, 22] and stochastic proximal gradient method [12, 13]. It admits smaller per-iteration memory and computational complexities, particularly in high-rank cases. Furthermore, it is flexible in terms of incorporating regularization terms and constraints on the latent factors. The stepsize used in BrasCPD is a constant, thus it is time-consuming (need many numerical experiments) to choose a proper stepsize to get a good numerical performance when implementing this stochastic algorithm. Due to this disadvantage, the adaptive stepsize method is also proposed in [11], namely AdaCPD, which combines with the Adagrad algorithm [10] and BrasCPD algorithm.

In order to get better numerical results, we consider the momentum gradient descent which was first proposed in the 1960s [24], and this idea has been applied to many applications, such as federated learning [18], Q-learning [33] and recommender systems [25] and so on. It combines the current gradient with a history of the previous step to accelerate the convergence of the algorithm. The full momentum update for minimizing $f(x)$ with $x \in \mathbb{R}^n$ is:

$$x^{k+1} = x^k - \alpha\nabla f(x^k) + \beta(x^k - x^{k-1}),$$

where α is the stepsize and β is a hyperparameter (typically $\beta \in [0, 1]$, although not limited to it), which scales down the previous step. Due to the momentum term $\beta(x^k - x^{k-1})$, the method avoids zigzagging for ill-conditioned problems, which leads to significant efficiency in practice. In this paper, we propose two momentum based block-randomized stochastic proximal gradient methods under constant and adaptive stepsize framework, namely mBrasCPD

and mAdaCPD, respectively, which are the momentum version of BrasCPD and AdaCPD in [11], respectively.

The rest of this paper is organized as follows. We present the mBrasCPD and mAdaCPD algorithms in details and provide convergence analysis of the mBrasCPD algorithm in Section 2 and Section 3, respectively. Some numerical experiments for the two proposed algorithms (mBrasCPD and mAdaCPD) compared with the other two algorithms (BrasCPD [11] and AdaCPD [11]) for the synthetic data and the real dataset are presented in Section 4. Finally, we draw a conclusion in Section 5.

2 Algorithms

In this section, we consider two momentum block-randomized SGD algorithms for CPD optimization problem (1.4), i.e., constant stepsize version and adaptive version. We name this two algorithms as mBrasCPD (Algorithm 1) and mAdaCPD (Algorithm 2), respectively.

2.1 mBrasCPD

We first consider the constant stepsize version of momentum block-randomized SGD algorithms (mBrasCPD). Our idea is to apply stochastic algorithm while exploiting the tensor fiber structure. The work in [11] considers a doubly stochastic procedure when update $\boldsymbol{A}_{(n)}$. Firstly, a mode index $n \in \{1, \ldots, N\}$ is randomly sampled at iteration k. Then we randomly sample a set of mode-n fibers that is indexed by $\mathcal{F}_n \subset \{1, \ldots, J_n\}$ with $|\mathcal{F}_n| = B$. Note that a mode-n fiber of $\boldsymbol{\mathcal{X}}$ (see Figure 1 for an example with sample size $|\mathcal{F}_n| = 6$.) is a row of the mode-n unfolding $\boldsymbol{X}_{(n)}$.

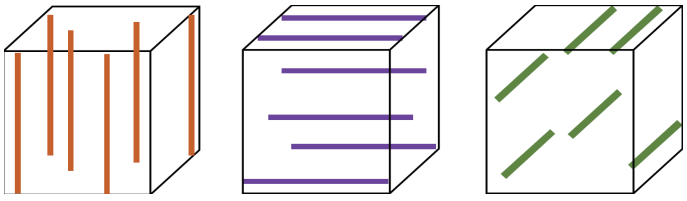


Figure 1: From left to right: the mode-1, 2, and 3 block randomized samples.

Let $\boldsymbol{G}^{(k)} \in \mathbb{R}^{(I_1+\cdots+I_N)\times R}$ be the gradient of $f(\boldsymbol{A}_{(1)}, \ldots, \boldsymbol{A}_{(N)})$ defined in (1.3) such that

$$\boldsymbol{G}^{(k)} = [\boldsymbol{G}^{(k)}_{(1)}, \ldots, \boldsymbol{G}^{(k)}_{(N)}]^T.$$

Here we have

$$\begin{aligned} \boldsymbol{G}^{(k)}_{(n)} &= \frac{1}{|\mathcal{F}_n|}\left(\boldsymbol{A}^{(k)}_{(n)}\boldsymbol{H}^{\top}_{(n)}(\mathcal{F}_n)\,\boldsymbol{H}_{(n)}(\mathcal{F}_n) - \boldsymbol{X}^{\top}_{(n)}(\mathcal{F}_n)\,\boldsymbol{H}_{(n)}(\mathcal{F}_n)\right), \\ \boldsymbol{G}^{(k)}_{(n')} &= \boldsymbol{0}, \quad n' \neq n, \end{aligned} \tag{2.1}$$

where

$$\boldsymbol{X}_{(n)}(\mathcal{F}_n) = \boldsymbol{X}_{(n)}(\mathcal{F}_n, :), \quad \boldsymbol{H}_{(n)}(\mathcal{F}_n) = \boldsymbol{H}_{(n)}(\mathcal{F}_n, :).$$

The latent variables $\boldsymbol{A}_{(n)}$ in [11] are updated by

$$\begin{aligned} \boldsymbol{A}^{(k+1)}_{(n)} &= \arg\min_{\boldsymbol{A}_{(n)}} \frac{1}{2\alpha^{(k)}}\left\|\boldsymbol{A}_{(n)} - \left(\boldsymbol{A}^{(k)}_{(n)} - \alpha^{(k)}\boldsymbol{G}^{(k)}_{(n)}\right)\right\|^2_F + h_n\left(\boldsymbol{A}_{(n)}\right), \\ \boldsymbol{A}^{(k+1)}_{(n')} &= \boldsymbol{A}^{(k)}_{(n')}, \quad n' \neq n. \end{aligned} \tag{2.2}$$

It shows that $\boldsymbol{G}_{(n)}^{(r)}$ is a gradient estimator for the function $f(\boldsymbol{A}_{(1)},\ldots,\boldsymbol{A}_{(N)})$ with the mode-n variable $\boldsymbol{A}_{(n)}$, and the update is an iteration of the proximal stochastic gradient algorithm with a minibatch size $|\mathcal{F}_n| = B$ for solving the subproblem (1.6). From [11], it shows that the most resource-consuming update process $\boldsymbol{H}_{(n)}^T\boldsymbol{X}_{(n)}$ compared with other algorithms [14, 35] can be avoided. We also know the computing cost of $\boldsymbol{X}_{(n)}^\top(\mathcal{F}_n,:)\,\boldsymbol{H}_{(n)}(\mathcal{F}_n,:)$ is only $\mathcal{O}(|\mathcal{F}_n|RI_n)$ which is significant smaller than $\mathcal{O}(I_1\ldots I_NR)$ for computing $\boldsymbol{X}_{(n)}^\top\boldsymbol{H}_{(n)}$. The sample size $|\mathcal{F}_n| = B$ can be controlled in practice and can get lower complexity than other exact methods [2, 14, 35] for solving the subproblem (1.6).

Although a doubly stochastic algorithm is applied to the problem (1.4), sometimes we need a long time to get a good numerical result because the computational cost of CP decomposition is expensive. If this stochastic algorithm can be accelerated, then we can further get better results in less time. In this paper, the momentum idea in [24] is applied to the above framework of updating $\boldsymbol{A}_{(n)}$. We update $\boldsymbol{A}_{(n)}$ by

$$\begin{aligned}\tilde{\boldsymbol{A}}_{(n)}^{(k)} =& \boldsymbol{A}_{(n)}^{(k)} + \beta^{(k)}\left(\boldsymbol{A}_{(n)}^{(k)} - \boldsymbol{A}_{(n)}^{(k-1)}\right),\\ \boldsymbol{A}_{(n)}^{k+1} =& \underset{\boldsymbol{A}_{(n)}}{\arg\min}\; h_n\left(\boldsymbol{A}_{(n)}\right) + \left\langle \boldsymbol{G}_{(n)}^{(k)}, \boldsymbol{A}_{(n)} - \boldsymbol{A}_{(n)}^{(k)}\right\rangle + \frac{1}{2\alpha^{(k)}}\left\|\boldsymbol{A}_{(n)} - \tilde{\boldsymbol{A}}_{(n)}^{(k)}\right\|_F^2.\end{aligned}$$

Then we equivalently rewrite the above update as follows

$$\begin{aligned}\boldsymbol{A}_{(n)}^{k+1} =& \underset{\boldsymbol{A}_{(n)}}{\arg\min}\; h_n\left(\boldsymbol{A}_{(n)}\right) + \left\langle \boldsymbol{G}_{(n)}^{(k)}, \boldsymbol{A}_{(n)} - \boldsymbol{A}_{(n)}^{(k)}\right\rangle\\ &+ \frac{1}{2\alpha^{(k)}}\left\|\boldsymbol{A}_{(n)} - \left(\boldsymbol{A}_{(n)}^{(k)} + \beta^{(k)}\left(\boldsymbol{A}_{(n)}^{(k)} - \boldsymbol{A}_{(n)}^{(k-1)}\right)\right)\right\|_F^2\\ =& \arg\min_{\boldsymbol{A}_{(n)}}\; h_n\left(\boldsymbol{A}_{(n)}\right) + \frac{1}{2\alpha^{(k)}}\left\|\boldsymbol{A}_{(n)} - \left(\boldsymbol{A}_{(n)}^{(k)} - \alpha^{(k)}\boldsymbol{G}_{(n)}^{(k)} + \beta^{(k)}\left(\boldsymbol{A}_{(n)}^{(k)} - \boldsymbol{A}_{(n)}^{(k-1)}\right)\right)\right\|_F^2.\end{aligned}$$

Thus we can update $\boldsymbol{A}_{(n)}$ by

$$\begin{aligned}\boldsymbol{A}_{(n)}^{(k+1)} =& \arg\min_{\boldsymbol{A}_{(n)}}\; \frac{1}{2\alpha^{(k)}}\left\|\boldsymbol{A}_{(n)} - \left(\boldsymbol{A}_{(n)}^{(k)} - \alpha^{(k)}\boldsymbol{G}_{(n)}^{(k)} + \beta^{(k)}\left(\boldsymbol{A}_{(n)}^{(k)} - \boldsymbol{A}_{(n)}^{(k-1)}\right)\right)\right\|_F^2\\ &+ h_n\left(\boldsymbol{A}_{(n)}\right),\\ \boldsymbol{A}_{(n')}^{(k+1)} =& \boldsymbol{A}_{(n')}^{(k)}, \quad n' \neq n.\end{aligned} \tag{2.3}$$

If $h_n(\cdot)$ is a closed proper convex function and its proximal operator can be computed easily, then update (2.3) can be solved by applying the proximal operator of $h_n(\cdot)$, which is denoted as

$$\boldsymbol{A}_{(n)}^{(k+1)} = \mathrm{Prox}_{\alpha^{(k)}h_n}\left(\boldsymbol{A}_{(n)}^{(k)} - \alpha^{(k)}\boldsymbol{G}_{(n)}^{(k)} + \beta^{(k)}\left(\boldsymbol{A}_{(n)}^{(k)} - \boldsymbol{A}_{(n)}^{(k-1)}\right)\right). \tag{2.4}$$

Now we describe the algorithmic framework of the mBrasCPD for the optimization problem (1.4) as follows.

Algorithm 1 mBrasCPD: momentum based block-randomized SGD for the optimization problem (1.4)

Input: N-way tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$; rank R; sample size B; initialization $\{\boldsymbol{A}_{(n)}^{(0)}\}$ and $\{\boldsymbol{A}_{(n)}^{(-1)}\}$, where $\boldsymbol{A}_{(n)}^{(-1)} = \boldsymbol{A}_{(n)}^{(0)}$, setting stepsize $\{\alpha^{(k)}\}$ and momentum parameter $\{\beta^{(k)}\}$.

1: $k \leftarrow 0$;
2: **repeat**
3: sample n uniformly from $\{1, \ldots, N\}$;
4: sample $\mathcal{F}_n$ uniformly from $\{1, \ldots, J_n\}$ with $|\mathcal{F}_n| = B$;
5: compute the stochastic gradient $\boldsymbol{G}^{(k)}$ from (2.1);
6: update $\boldsymbol{A}_{(n)}^{(k+1)}$ and $\boldsymbol{A}_{(n')}^{(k+1)}$ from (2.3);
7: $k \leftarrow k + 1$;
8: **until** some stopping criterion is reached;

Output: $\{\boldsymbol{A}_{(n)}^{(k)}\}_{n=1}^{N}$

When $\beta^{(k)} = 0$ in Algorithm 1, the mBrasCPD algorithm would be reduced to the BrasCPD algorithm [11].

2.2 mAdaCPD

An obvious disadvantage of the mBrasCPD algorithm is that we need to conduct a lot of numerical experiments to find a proper stepsize. Furthermore, it can be seen from the numerical experiment in Section 4 that the convergence speed of this algorithm is relatively slow. Thus we need a better way to select the stepsize. Recently, a series of algorithms were proposed in the machine learning community for adaptive stepsize scheduling when training deep neural networks, such as AdaGrad [10], RMSProp[1], Adam [16] and etc. The idea of these adaptive algorithms can be understood as adaptively assign different learning rates to each parameter. To be more precise, it adapts the learning rate to the parameters and performs larger updates for rare parameters and smaller updates of frequent parameters.

Now, we consider an adaptive stepsize version of mBrasCPD algorithm in this subsection, namely, mAdaCPD. The adaptive stepsize is defined as

$$\left[\boldsymbol{\eta}_{(n)}^{(k)}\right]_{i,r} = \frac{\eta}{\left(b + \sum_{t=1}^{k} \left[\boldsymbol{G}_{(n)}^{(t)}\right]_{i,r}^2\right)^{\frac{1}{2}+\varepsilon}}, \tag{2.5}$$

where $\eta, b, \varepsilon > 0$. Setting $b = \varepsilon = 0$ does not hurt the numerical performance in practice. Then we can apply the adaptive rule of stepsize to mBrasCPD algorithm (Algorithm 1), i.e., the 6-th line in Algorithm 1 is replaced by (2.5) and

$$\begin{aligned} \boldsymbol{A}_{(n)}^{(k+1)} &= \operatorname{Prox}_{\boldsymbol{\eta}_{(n)}^{(k)} h_n}\left(\boldsymbol{A}_{(n)}^{(k)} - \boldsymbol{\eta}_{(n)}^{(k)} \circledast \boldsymbol{G}_{(n)}^{(k)} + \beta^{(k)}\left(\boldsymbol{A}_{(n)}^{(k)} - \boldsymbol{A}_{(n)}^{(k-1)}\right)\right), \\ \boldsymbol{A}_{(n')}^{(k+1)} &= \boldsymbol{A}_{(n')}^{(k)}, \quad n' \neq n, \end{aligned} \tag{2.6}$$

where $\circledast$ denotes the element-wise product. Now the mAdaCPD algorithm (Algorithm 2) framework is summarized as follows.

[1] http://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf

Algorithm 2 mAdaCPD : momentum based block-randomized SGD with adaptive stepsize for the optimization problem (1.4)

Input: N-way tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$; rank R; sample size B; initialization $\{\boldsymbol{A}_{(n)}^{(0)}\}$ and $\{\boldsymbol{A}_{(n)}^{(-1)}\}$, where $\boldsymbol{A}_{(n)}^{(-1)} = \boldsymbol{A}_{(n)}^{(0)}$, and momentum parameter $\{\beta^{(k)}\}$.

1: $k \leftarrow 0$;
2: **repeat**
3: sample n uniformly from $\{1, \ldots, N\}$;
4: sample $\mathcal{F}_n$ uniformly from $\{1, \ldots, J_n\}$ with $|\mathcal{F}_n| = B$;
5: compute the stochastic gradient $\boldsymbol{G}^{(k)}$ from (2.1);
6: compute the stepsize $\boldsymbol{\eta}_{(n)}^{(k)}$ from (2.5);
7: update $\boldsymbol{A}_{(n)}^{(k+1)}$ and $\boldsymbol{A}_{(n')}^{(k+1)}$ from (2.6);
8: $k \leftarrow k + 1$;
9: **until** some stopping criterion is reached;

Output: $\{\boldsymbol{A}_{(n)}^{(k)}\}_{n=1}^{N}$

When $\beta^{(k)} = 0$ in Algorithm 2, the mAdaCPD algorithm would be reduced to the AdaCPD algorithm [11].

3 Convergence Analysis

Before introducing the convergence analysis of the mBrasCPD algorithm (Algorithm 1), we present the following lemmas and assumptions.

Lemma 3.1. *[11] For any $\{\boldsymbol{A}_{(1)}, \ldots, \boldsymbol{A}_{(N)}\}$, $\{\bar{\boldsymbol{A}}_{(1)}, \ldots, \bar{\boldsymbol{A}}_{(N)}\}$ and any mode $n \in \{1, \ldots, N\}$, there exists a constant $\bar{L}_{(n)}$ such that*

$$\begin{aligned} f\left(\boldsymbol{A}_{(1)}, \ldots, \boldsymbol{A}_{(N)}\right) \leq & f\left(\bar{\boldsymbol{A}}_{(1)}, \ldots, \bar{\boldsymbol{A}}_{(N)}\right) + \left\langle \nabla_{\bar{\boldsymbol{A}}_{(n)}} f\left(\bar{\boldsymbol{A}}_{(1)}, \ldots, \bar{\boldsymbol{A}}_{(N)}\right), \boldsymbol{A}_{(n)} - \bar{\boldsymbol{A}}_{(n)} \right\rangle \\ & + \frac{\bar{L}_{(n)}}{2} \left\| \boldsymbol{A}_{(n)} - \bar{\boldsymbol{A}}_{(n)} \right\|_F^2, \end{aligned} \tag{3.1}$$

where $f(\cdot)$ is defined as (1.3) *and $\boldsymbol{A}_{(i)} = \bar{\boldsymbol{A}}_{(i)}$ for $i \neq n$.*

The equation (3.1) holds because the objective function $f(\boldsymbol{A}_{(1)}, \ldots, \boldsymbol{A}_{(N)})$ is the quadratic of variable $\boldsymbol{A}_{(n)}$ and has a Lipchitz continuous gradient with Lipschitz constant $\bar{L}_{(n)} \geq \lambda_{\max}(\bar{\boldsymbol{H}}_{(n)}^T \bar{\boldsymbol{H}}_{(n)})$.

Define $\xi^{(k)} \in \{1, \ldots, N\}$ and $\zeta \subseteq \{1, \ldots, J_{\xi^{(k)}}\}$ as the random variables responsible for the mode and fibers in iteration r respectively for mBrasCPD algorithm (Algorithm 1). With these random variables, it holds that

$$\mathrm{P}(\xi^{(k)} = n) = \frac{1}{N}, \quad \mathrm{P}(\zeta^{(k)} = \mathcal{S} | \xi^{(k)} = n) = \frac{1}{M},$$

where $M = \begin{pmatrix} J_n \\ B \end{pmatrix}$, $n \in \{1, \ldots, N\}$ and $\mathcal{S}$ is an any subset of $\{1, \ldots, J_{\xi^{(k)}}\}$ (i.e., $\mathcal{S} \subseteq \{1, \ldots, J_{\xi^{(k)}}\}$) with $|\mathcal{S}| = B$.

Lemma 3.2. *[11] Denote $\mathcal{B}^{(k)}$ as the filtration generated by the random variables $\{\xi^{(1)}, \zeta^{(1)}, \ldots, \xi^{(k-1)}, \zeta^{(k-1)}\}$ such that the $\{\boldsymbol{A}_{(1)}^{(k)}, \ldots, \boldsymbol{A}_{(N)}^{(k)}\}$ in iteration k is determined conditioned*

on $\mathcal{B}^{(k)}$. Then the stochastic gradient $\boldsymbol{G}^{(k)}_{(\xi^{(k)})}$ from (2.1) is an unbiased estimate for the full gradient for $\boldsymbol{A}_{(\xi^{(k)})}$, that is

$$\mathbb{E}_{\zeta^{(k)}}\left[\boldsymbol{G}^{(k)}_{(\xi^{(k)})} \mid \mathcal{B}^{(k)}, \xi^{(k)}\right] = \nabla_{\boldsymbol{A}_{(\xi^{(k)})}} f\left(\boldsymbol{A}^{(k)}_{(1)}, \ldots, \boldsymbol{A}^{(k)}_{(N)}\right). \tag{3.2}$$

The above lemma says that the block stochastic gradient estimator $\boldsymbol{G}^{(k)}_{(n)}$ is an unbiased estimation for the gradient $\nabla_{\boldsymbol{A}_{(n)}} f\left(\boldsymbol{A}^{(k)}_{(1)}, \ldots, \boldsymbol{A}^{(k)}_{(N)}\right)$.

Lemma 3.3. *[20] Let $\{a_t\}$ and $\{b_t\}$ be two nonnegative sequences such that b_t is bounded, if we have $\sum_{t=0}^{\infty} a_t b_t$ converges and $\sum_{t=0}^{\infty} a_t$ diverges, then it holds that*

$$\liminf_{t\to\infty} b_t = 0.$$

The above lemma is used when we analyze the convergence of mBrasCPD algorithm (Algorithm 1) in this paper.

Assumption 3.1. From the Robbins-Monro rule [27], we assume the stepsize $\{\alpha^{(k)}\}_{k\geq 0}$ used in mBrasCPD algorithm (Algorithm 1) satisfy

$$\sum_{k=0}^{\infty} \alpha^{(k)} = \infty.$$

The above assumption is required to guarantee that the steps $\{\alpha^{(k)}\}_{k\geq 0}$ are large enough to eventually overcome any initial conditions or random fluctuations.

Assumption 3.2. The optimal objection value Φ^* of (1.4) is finite.

Assumption 3.3. There exists a sequence $\{\sigma^{(k)}\}_{k\geq 0}$ such that

$$\begin{aligned} &\mathbb{E}_{\zeta^{(k)}}\left[\left\|\boldsymbol{G}^{(k)}_{(\xi^{(k)})} - \nabla_{\boldsymbol{A}_{(\xi^{(k)})}} f\left(\boldsymbol{A}^{(k)}_{(1)}, \ldots, \boldsymbol{A}^{(k)}_{(N)}\right)\right\|^2 \mid \mathcal{B}^{(k)}, \xi^{(k)}\right] \leq (\sigma^{(k)})^2, \\ &\sum_{k=0}^{\infty} \alpha^{(k)} (\sigma^{(k)})^2 < \infty, \end{aligned} \tag{3.3}$$

where $\{\alpha^{(k)}\}_{k\geq 0}$ is the stepsize sequence following Assumption 3.1.

The first inequality in this assumption can make the variance of approximated gradient estimator $\boldsymbol{G}^{(k)}_{(\xi^{(k)})}$ be bounded, which is commonly used in stochastical gradient algorithms for convergence analysis. The second inequality is applied to the convergence analysis of the algorithm in this paper.

In this paper, a solution $\{\boldsymbol{A}^{(k)}_{(1)}, \ldots, \boldsymbol{A}^{(k)}_{(N)}\}$ is a stationary point of (1.4) if $\boldsymbol{P}^{(k)}_{(n)} = 0$ for any n with

$$\begin{aligned} \boldsymbol{P}^{(k)}_{(n)} = \frac{1}{\alpha^{(k)}} \Big(\boldsymbol{A}^{(k)}_{(n)} - \mathrm{Prox}_{\alpha^{(k)} h_n}\Big(\boldsymbol{A}^{(k)}_{(n)} - \alpha^{(k)} \nabla_{\boldsymbol{A}_{(n)}} f\left(\boldsymbol{A}^{(k)}_{(1)}, \ldots, \boldsymbol{A}^{(k)}_{(N)}\right) \\ + \beta^{(k)}\left(\boldsymbol{A}^{(k)}_{(n)} - \boldsymbol{A}^{(k-1)}_{(n)}\right)\Big)\Big). \end{aligned}$$

We knew that this condition with $\beta^{(k)} = 0$ is satisfied in a blockwise fashion [26, 35]. In this paper, we extend the result that $\mathbb{E}\left[\left\|\boldsymbol{P}^{(k)}_{(n)}\right\|^2\right]$ vanishes in the case of $\beta^{(k)} = 0$ [11] to the case of $\beta^{(k)} \neq 0$ for all n as $k \to \infty$.

We have the following convergence property for mBrasCPD algorithm (Algorithm 1).

Lemma 3.4. *Assume that Assumptions 3.1-3.3 hold and $h_n(\cdot)$ is a proper closed convex function. And the sequence $\{\boldsymbol{A}_{(n)}^{(k)}\}_{k\geq 0}$ are generated by mBrasCPD algorithm (Algorithm 1), we have the following inequality holds*

$$\begin{aligned}
&\mathbb{E}\left[\Phi\left(\boldsymbol{A}_{(1)}^{(k+1)},\ldots,\boldsymbol{A}_{(N)}^{(k+1)}\right)\right]-\mathbb{E}\left[\Phi\left(\boldsymbol{A}_{(1)}^{(k)},\ldots,\boldsymbol{A}_{(N)}^{(k)}\right)\right]\\
\leq&\alpha^{(k)}(\sigma^{(k)})^2+\frac{\alpha^{(k-1)}\beta^{(k)}}{2}\mathbb{E}\left[\left\|\boldsymbol{p}_{\left(\xi^{(k)}\right)}^{(k-1)}\right\|^2\right]+\frac{(\alpha^{(k)})^2L-\alpha^{(k)}+\alpha^{(k-1)}\beta^{(k)}}{2}\mathbb{E}\left[\left\|\boldsymbol{p}_{\left(\xi^{(k)}\right)}^{(k)}\right\|^2\right],
\end{aligned}$$

where $\boldsymbol{p}_{\left(\xi^{(k)}\right)}^{(k)}=\frac{1}{\alpha^{(k)}}\left(\boldsymbol{A}_{\left(\xi^{(k)}\right)}^{(k+1)}-\boldsymbol{A}_{\left(\xi^{(k)}\right)}^{(k)}\right)$ and the definition of $\sigma^{(k)}$ can be referred to Assumption 3.3.

Proof. From the update of $\boldsymbol{A}_{(n)}^{(k+1)}$ in BrasCPD framework (Algorithm 1), we know that

$$\begin{aligned}
&\boldsymbol{A}_{(n)}^{(k+1)}\\
=&\arg\min_{\boldsymbol{A}_{(n)}}\ h_n\left(\boldsymbol{A}_{(n)}\right)+\frac{1}{2\alpha^{(k)}}\left\|\boldsymbol{A}_{(n)}-\left(\boldsymbol{A}_{(n)}^{(k)}-\alpha^{(k)}\boldsymbol{G}_{(n)}^{(k)}+\beta^{(k)}\left(\boldsymbol{A}_{(n)}^{(k)}-\boldsymbol{A}_{(n)}^{(k-1)}\right)\right)\right\|_F^2\\
=&\arg\min_{\boldsymbol{A}_{(n)}}\ h_n\left(\boldsymbol{A}_{(n)}\right)+\left\langle\boldsymbol{G}_{(n)}^{(k)}-\frac{\beta^{(k)}}{\alpha^{(k)}}\left(\boldsymbol{A}_{(n)}^{(k)}-\boldsymbol{A}_{(n)}^{(k-1)}\right),\boldsymbol{A}_{(n)}-\boldsymbol{A}_{(n)}^{(k)}\right\rangle\\
&\qquad+\frac{1}{2\alpha^{(k)}}\left\|\boldsymbol{A}_{(n)}-\boldsymbol{A}_{(n)}^{(k)}\right\|^2
\end{aligned}\tag{3.4}$$

for a randomly selected n. For a given $\xi^{(k)}$, by setting $\boldsymbol{A}_{(n)}=\boldsymbol{A}_{\left(\xi^{(k)}\right)}^{(k)}$ and $\boldsymbol{A}_{(n)}=\boldsymbol{A}_{\left(\xi^{(k)}\right)}^{(k+1)}$ in above equation respectively, we have the following inequality holds

$$\begin{aligned}
&h_{\xi^{(k)}}\left(\boldsymbol{A}_{\left(\xi^{(k)}\right)}^{(k+1)}\right)-h_{\xi^{(k)}}\left(\boldsymbol{A}_{\left(\xi^{(k)}\right)}^{(k)}\right)\\
\leq&-\left\langle\boldsymbol{G}_{\left(\xi^{(k)}\right)}^{(k)}-\frac{\beta^{(k)}}{\alpha^{(k)}}\left(\boldsymbol{A}_{\left(\xi^{(k)}\right)}^{(k)}-\boldsymbol{A}_{\left(\xi^{(k)}\right)}^{(k-1)}\right),\boldsymbol{A}_{\left(\xi^{(k)}\right)}^{(k+1)}-\boldsymbol{A}_{\left(\xi^{(k)}\right)}^{(k)}\right\rangle-\frac{1}{2\alpha^{(k)}}\left\|\boldsymbol{A}_{\left(\xi^{(k)}\right)}^{(k+1)}-\boldsymbol{A}_{\left(\xi^{(k)}\right)}^{(k)}\right\|^2.
\end{aligned}\tag{3.5}$$

From the block Lipschitz continuity of the quadratic function $f(\boldsymbol{A}_{(1)},\ldots,\boldsymbol{A}_{(N)})$ in Lemma 3.1, we have

$$\begin{aligned}
&f\left(\boldsymbol{A}_{(1)}^{(k+1)},\ldots,\boldsymbol{A}_{(N)}^{(k+1)}\right)-f\left(\boldsymbol{A}_{(1)}^{(k)},\ldots,\boldsymbol{A}_{(N)}^{(k)}\right)\\
\leq&\left\langle\nabla_{\boldsymbol{A}_{\left(\xi^{(k)}\right)}}f\left(\boldsymbol{A}_{(1)}^{(k)},\ldots,\boldsymbol{A}_{(N)}^{(k)}\right),\boldsymbol{A}_{\left(\xi^{(k)}\right)}^{(k+1)}-\boldsymbol{A}_{\left(\xi^{(k)}\right)}^{(k)}\right\rangle+\frac{L_{\left(\xi^{(k)}\right)}^{(k)}}{2}\left\|\boldsymbol{A}_{\left(\xi^{(k)}\right)}^{(k+1)}-\boldsymbol{A}_{\left(\xi^{(k)}\right)}^{(k)}\right\|^2.
\end{aligned}\tag{3.6}$$

Combing these two inequalities, i.e. (3.5) and (3.6), it shows that

$$\begin{aligned}
&\Phi\left(\boldsymbol{A}_{(1)}^{(k+1)},\ldots,\boldsymbol{A}_{(N)}^{(k+1)}\right)-\Phi\left(\boldsymbol{A}_{(1)}^{(k)},\ldots,\boldsymbol{A}_{(N)}^{(k)}\right)\\
\leq&\left\langle\nabla_{\boldsymbol{A}_{\left(\xi^{(k)}\right)}}f\left(\boldsymbol{A}_{(1)}^{(k)},\ldots,\boldsymbol{A}_{(N)}^{(k)}\right)-\boldsymbol{G}_{\left(\xi^{(k)}\right)}^{(k)}+\frac{\beta^{(k)}}{\alpha^{(k)}}\left(\boldsymbol{A}_{\left(\xi^{(k)}\right)}^{(k)}-\boldsymbol{A}_{\left(\xi^{(k)}\right)}^{(k-1)}\right),\boldsymbol{A}_{\left(\xi^{(k)}\right)}^{(k+1)}-\boldsymbol{A}_{\left(\xi^{(k)}\right)}^{(k)}\right\rangle\\
&+\left(\frac{L_{\left(\xi^{(k)}\right)}^{(k)}}{2}-\frac{1}{2\alpha^{(k)}}\right)\left\|\boldsymbol{A}_{\left(\xi^{(k)}\right)}^{(k+1)}-\boldsymbol{A}_{\left(\xi^{(k)}\right)}^{(k)}\right\|^2.
\end{aligned}\tag{3.7}$$

Denote $\boldsymbol{p}^{(k)}_{\left(\xi^{(k)}\right)} = \frac{1}{\alpha^{(k)}}\left(\boldsymbol{A}^{(k+1)}_{\left(\xi^{(k)}\right)} - \boldsymbol{A}^{(k)}_{\left(\xi^{(k)}\right)}\right)$, then from (3.7) and let

$$L \geq \lambda_{\max}\left(\left(\boldsymbol{H}^{(k)}_{\left(\xi^{(k)}\right)}\right)^T \boldsymbol{H}^{(k)}_{\left(\xi^{(k)}\right)}\right) \geq L^{(k)}_{\left(\xi^{(k)}\right)}$$

for simplicity. Thus we can get

$$\begin{aligned}
&\Phi\left(\boldsymbol{A}^{(k+1)}_{(1)}, \ldots, \boldsymbol{A}^{(k+1)}_{(N)}\right) - \Phi\left(\boldsymbol{A}^{(k)}_{(1)}, \ldots, \boldsymbol{A}^{(k)}_{(N)}\right) \\
\leq &\alpha^{(k)}\left\langle \nabla_{\boldsymbol{A}_{\left(\xi^{(k)}\right)}} f\left(\boldsymbol{A}^{(k)}_{(1)}, \ldots, \boldsymbol{A}^{(k)}_{(N)}\right) - \boldsymbol{G}^{(k)}_{\left(\xi^{(k)}\right)} + \frac{\alpha^{(k-1)}\beta^{(k)}}{\alpha^{(k)}} \boldsymbol{p}^{(k-1)}_{\left(\xi^{(k)}\right)}, \boldsymbol{p}^{(k)}_{\left(\xi^{(k)}\right)} \right\rangle \\
&+ \left(\frac{(\alpha^{(k)})^2 L}{2} - \frac{\alpha^{(k)}}{2}\right)\left\| \boldsymbol{p}^{(k)}_{\left(\xi^{(k)}\right)} \right\|^2 \\
= &\alpha^{(k)}\left\langle \nabla_{\boldsymbol{A}_{\left(\xi^{(k)}\right)}} f\left(\boldsymbol{A}^{(k)}_{(1)}, \ldots, \boldsymbol{A}^{(k)}_{(N)}\right) - \boldsymbol{G}^{(k)}_{\left(\xi^{(k)}\right)}, \boldsymbol{p}^{(k)}_{\left(\xi^{(k)}\right)} - \boldsymbol{P}^{(k)}_{\left(\xi^{(k)}\right)} \right\rangle \\
&+ \left(\frac{(\alpha^{(k)})^2 L}{2} - \frac{\alpha^{(k)}}{2}\right)\left\| \boldsymbol{p}^{(k)}_{\left(\xi^{(k)}\right)} \right\|^2 \\
&+ \alpha^{(k)}\left\langle \nabla_{\boldsymbol{A}_{\left(\xi^{(k)}\right)}} f\left(\boldsymbol{A}^{(k)}_{(1)}, \ldots, \boldsymbol{A}^{(k)}_{(N)}\right) - \boldsymbol{G}^{(k)}_{\left(\xi^{(k)}\right)}, \boldsymbol{P}^{(k)}_{\left(\xi^{(k)}\right)} \right\rangle \\
&+ \alpha^{(k-1)}\beta^{(k)}\left\langle \boldsymbol{p}^{(k-1)}_{\left(\xi^{(k)}\right)}, \boldsymbol{p}^{(k)}_{\left(\xi^{(k)}\right)} \right\rangle.
\end{aligned} \tag{3.8}$$

Take the expectation of the above inequality conditioned on the filtration $\mathcal{B}^{(k)}$ and $\xi^{(k)}$. By Lemma 3.2, we have

$$\mathbb{E}_{\zeta^{(k)}}\left[\left\langle \nabla_{\boldsymbol{A}_{\left(\xi^{(k)}\right)}} f\left(\boldsymbol{A}^{(k)}_{(1)}, \ldots, \boldsymbol{A}^{(k)}_{(N)}\right) - \boldsymbol{G}^{(k)}_{\left(\xi^{(k)}\right)}, \boldsymbol{P}^{(k)}_{\left(\xi^{(k)}\right)} \right\rangle \mid \mathcal{B}^{(k)}, \xi^{(k)}\right] = 0. \tag{3.9}$$

The first term in the right side of (3.8) can be bounded follows from

$$\begin{aligned}
&\mathbb{E}_{\zeta^{(k)}}\left[\left\langle \nabla_{\boldsymbol{A}_{\left(\xi^{(k)}\right)}} f\left(\boldsymbol{A}^{(k)}_{(1)}, \ldots, \boldsymbol{A}^{(k)}_{(N)}\right) - \boldsymbol{G}^{(k)}_{\left(\xi^{(k)}\right)}, \boldsymbol{p}^{(k)}_{\left(\xi^{(k)}\right)} - \boldsymbol{P}^{(k)}_{\left(\xi^{(k)}\right)} \right\rangle \mid \mathcal{B}^{(k)}, \xi^{(k)}\right] \\
\leq &\mathbb{E}_{\zeta^{(k)}}\left[\|\delta^{(k)}\| \, \|\boldsymbol{p}^{(k)}_{\left(\xi^{(k)}\right)} - \boldsymbol{P}^{(k)}_{\left(\xi^{(k)}\right)}\| \mid \mathcal{B}^{(k)}, \xi^{(k)}\right] \\
\leq &\mathbb{E}_{\zeta^{(k)}}\left[\|\delta^{(k)}\|^2 \mid \mathcal{B}^{(k)}, \xi^{(k)}\right] \\
\leq &(\sigma^{(k)})^2,
\end{aligned} \tag{3.10}$$

where the first inequality folllows from the Cauchy-Schwartz inequality and the second inequality is due to the non-expansiveness of the proximal operator of convex function $h_n(\cdot)$.

From the left terms in the right side of (3.8), we have

$$\begin{aligned}
&\mathbb{E}_{\zeta^{(k)}}\left[\alpha^{(k-1)}\beta^{(k)}\left\langle \boldsymbol{p}^{(k-1)}_{(\xi^{(k)})}, \boldsymbol{p}^{(k)}_{(\xi^{(k)})}\right\rangle + \left(\frac{(\alpha^{(k)})^2 L}{2} - \frac{\alpha^{(k)}}{2}\right)\left\|\boldsymbol{p}^{(k)}_{(\xi^{(k)})}\right\|^2 \mid \mathcal{B}^{(k)}, \xi^{(k)}\right] \\
\leq &\mathbb{E}_{\zeta^{(k)}}\left[\frac{\alpha^{(k-1)}\beta^{(k)}}{2}\left(\left\|\boldsymbol{p}^{(k-1)}_{(\xi^{(k)})}\right\|^2 + \left\|\boldsymbol{p}^{(k)}_{(\xi^{(k)})}\right\|^2\right)\right. \\
&\qquad\left. + \left(\frac{(\alpha^{(k)})^2 L}{2} - \frac{\alpha^{(k)}}{2}\right)\left\|\boldsymbol{p}^{(k)}_{(\xi^{(k)})}\right\|^2 \mid \mathcal{B}^{(k)}, \xi^{(k)}\right] \\
= &\mathbb{E}_{\zeta^{(k)}}\left[\frac{\alpha^{(k-1)}\beta^{(k)}}{2}\left\|\boldsymbol{p}^{(k-1)}_{(\xi^{(k)})}\right\|^2\right] + \mathbb{E}\left[\frac{(\alpha^{(k)})^2 L - \alpha^{(k)} + \alpha^{(k-1)}\beta^{(k)}}{2}\left\|\boldsymbol{p}^{(k)}_{(\xi^{(k)})}\right\|^2 \mid \mathcal{B}^{(k)}, \xi^{(k)}\right].
\end{aligned} \tag{3.11}$$

Combining the above inequalities (3.8), (3.9), (3.10) and (3.11), taking the total expectation, it shows that

$$\begin{aligned}
&\mathbb{E}\left[\Phi\left(\boldsymbol{A}^{(k+1)}_{(1)}, \ldots, \boldsymbol{A}^{(k+1)}_{(N)}\right)\right] - \mathbb{E}\left[\Phi\left(\boldsymbol{A}^{(k)}_{(1)}, \ldots, \boldsymbol{A}^{(k)}_{(N)}\right)\right] \\
\leq &\alpha^{(k)}(\sigma^{(k)})^2 + \frac{\alpha^{(k-1)}\beta^{(k)}}{2}\mathbb{E}\left[\left\|\boldsymbol{p}^{(k-1)}_{(\xi^{(k)})}\right\|^2\right] + \frac{(\alpha^{(k)})^2 L - \alpha^{(k)} + \alpha^{(k-1)}\beta^{(k)}}{2}\mathbb{E}\left[\left\|\boldsymbol{p}^{(k)}_{(\xi^{(k)})}\right\|^2\right].
\end{aligned}$$

□

Now we give the main theoretical result of this paper as follows.

Theorem 3.5. *Assume that Assumptions 3.1-3.3 hold, and $h_n(\cdot)$ is a proper closed convex function. Suppose the following condition*[2]*. for $\{\alpha^{(k)}\}$ and $\{\beta^{(k)}\}$ holds*

$$0 \leq \alpha^{(k)}\beta^{(k+1)} < -(\alpha^{(k)})^2 L + \alpha^{(k)} - \alpha^{(k-1)}\beta^{(k)}. \tag{3.12}$$

Then the sequence $\{\boldsymbol{A}^{(k)}_{(n)}\}_{k\geq 0}$ generated by mBrasCPD algorithm satisfies

$$\liminf_{k\to\infty} \mathbb{E}\left[\left\|\boldsymbol{P}^{(k)}_{(n)}\right\|^2\right] = 0, \ \forall n.$$

Proof. From Lemma 3.4, it shows that

$$\begin{aligned}
&\mathbb{E}\left[\Phi\left(\boldsymbol{A}^{(k+1)}_{(1)}, \ldots, \boldsymbol{A}^{(k+1)}_{(N)}\right)\right] - \mathbb{E}\left[\Phi\left(\boldsymbol{A}^{(k)}_{(1)}, \ldots, \boldsymbol{A}^{(k)}_{(N)}\right)\right] \\
\leq &\alpha^{(k)}(\sigma^{(k)})^2 + \frac{\alpha^{(k-1)}\beta^{(k)}}{2}\mathbb{E}\left[\left\|\boldsymbol{p}^{(k-1)}_{(\xi^{(k)})}\right\|^2\right] + \frac{(\alpha^{(k)})^2 L - \alpha^{(k)} + \alpha^{(k-1)}\beta^{(k)}}{2}\mathbb{E}\left[\left\|\boldsymbol{p}^{(k)}_{(\xi^{(k)})}\right\|^2\right].
\end{aligned}$$

Summing up the above inequality from $t = 0$ to $t = k$, we have

$$\begin{aligned}
&\mathbb{E}\left[\Phi\left(\boldsymbol{A}^{(k+1)}_{(1)}, \ldots, \boldsymbol{A}^{(k+1)}_{(N)}\right)\right] - \Phi\left(\boldsymbol{A}^{(0)}_{(1)}, \ldots, \boldsymbol{A}^{(0)}_{(N)}\right) \\
\leq &\sum_{t=0}^{k}\alpha^{(t)}(\sigma^{(t)})^2 + \sum_{t=0}^{k}\frac{\alpha^{(t-1)}\beta^{(t)}}{2}\mathbb{E}\left[\left\|\boldsymbol{p}^{(t-1)}_{(\xi^{(k)})}\right\|^2\right] \\
&+ \sum_{t=0}^{k}\frac{(\alpha^{(t)})^2 L - \alpha^{(t)} + \alpha^{(t-1)}\beta^{(t)}}{2}\mathbb{E}\left[\left\|\boldsymbol{p}^{(t)}_{(\xi^{(k)})}\right\|^2\right].
\end{aligned} \tag{3.13}$$

[2]In fact, the algorithm also holds when the condition $\alpha^{(k)}\beta^{(k+1)} \geq 0$ is not satisfied. It shows that if $\beta^{(k)}$ is lower bounded, the covergence analysis in Theorem 3.5 also holds

Thus we have the following inequality holds

$$\begin{aligned}
&\sum_{t=0}^{k} \frac{-(\alpha^{(t)})^2 L + \alpha^{(t)} - \alpha^{(t-1)}\beta^{(t)}}{2} \mathbb{E}\left[\left\|\boldsymbol{p}^{(t)}_{(\xi^{(k)})}\right\|^2\right] - \sum_{t=0}^{k} \frac{\alpha^{(t-1)}\beta^{(t)}}{2} \mathbb{E}\left[\left\|\boldsymbol{p}^{(t-1)}_{(\xi^{(k)})}\right\|^2\right] \\
\leq &\sum_{t=0}^{k} \alpha^{(t)}(\sigma^{(t)})^2 + \Phi\left(\boldsymbol{A}^{(0)}_{(1)}, \ldots, \boldsymbol{A}^{(0)}_{(N)}\right) - \mathbb{E}\left[\Phi\left(\boldsymbol{A}^{(k+1)}_{(1)}, \ldots, \boldsymbol{A}^{(k+1)}_{(N)}\right)\right] \\
\leq &\sum_{t=0}^{k} \alpha^{(t)}(\sigma^{(t)})^2 + \Phi\left(\boldsymbol{A}^{(0)}_{(1)}, \ldots, \boldsymbol{A}^{(0)}_{(N)}\right) - \Phi^*,
\end{aligned} \tag{3.14}$$

where Φ^* is the optimal objection value of (1.4) from Assumption 3.2. From the condition (3.12), we have

$$-(\alpha^{(t)})^2 L + \alpha^{(t)} - \alpha^{(t-1)}\beta^{(t)} > \alpha^{(t)}\beta^{(t+1)} \geq 0$$

satisfied, then we can get from (3.14)

$$\begin{aligned}
&\sum_{t=0}^{k} \frac{-(\alpha^{(t)})^2 L + \alpha^{(t)} - \alpha^{(t-1)}\beta^{(t)}}{2} \mathbb{E}\left[\left\|\boldsymbol{p}^{(t)}_{(\xi^{(k)})}\right\|^2\right] - \sum_{t=0}^{k} \frac{\alpha^{(t-1)}\beta^{(t)}}{2} \mathbb{E}\left[\left\|\boldsymbol{p}^{(t-1)}_{(\xi^{(k)})}\right\|^2\right] \\
= &\sum_{t=0}^{k} \frac{-(\alpha^{(t)})^2 L + \alpha^{(t)} - \alpha^{(t-1)}\beta^{(t)} - \alpha^{(t)}\beta^{(t+1)}}{2} \mathbb{E}\left[\left\|\boldsymbol{p}^{(t)}_{(\xi^{(k)})}\right\|^2\right] \\
&+ \frac{\alpha^{(k)}\beta^{(k+1)}}{2} \mathbb{E}\left[\left\|\boldsymbol{p}^{(k)}_{(\xi^{(k)})}\right\|^2\right] - \frac{\alpha^{(-1)}\beta^{(0)}}{2} \mathbb{E}\left[\left\|\boldsymbol{p}^{(-1)}_{(\xi^{(k)})}\right\|^2\right] \\
\leq &\sum_{t=0}^{k} \alpha^{(t)}(\sigma^{(t)})^2 + \Phi\left(\boldsymbol{A}^{(0)}_{(1)}, \ldots, \boldsymbol{A}^{(0)}_{(N)}\right) - \Phi^*.
\end{aligned} \tag{3.15}$$

Thus we have

$$\begin{aligned}
&\sum_{t=0}^{k} \frac{-(\alpha^{(t)})^2 L + \alpha^{(t)} - \alpha^{(t-1)}\beta^{(t)} - \alpha^{(t)}\beta^{(t+1)}}{2} \mathbb{E}\left[\left\|\boldsymbol{p}^{(t)}_{(\xi^{(k)})}\right\|^2\right] + \frac{\alpha^{(k)}\beta^{(k+1)}}{2} \mathbb{E}\left[\left\|\boldsymbol{p}^{(k)}_{(\xi^{(k)})}\right\|^2\right] \\
\leq &\sum_{t=0}^{k} \alpha^{(t)}(\sigma^{(t)})^2 + \Phi\left(\boldsymbol{A}^{(0)}_{(1)}, \ldots, \boldsymbol{A}^{(0)}_{(N)}\right) - \Phi^* + \frac{\alpha^{(-1)}\beta^{(0)}}{2} \mathbb{E}\left[\left\|\boldsymbol{p}^{(-1)}_{(\xi^{(k)})}\right\|^2\right].
\end{aligned}$$

By Assumptions 3.2 and 3.3, taking $k \to \infty$ on the above inequality, it shows that the right side of the above inequality is bounded. Combined with Lemma 3.3, we conclude that

$$\liminf_{k\to\infty} \mathbb{E}\left[\left\|\boldsymbol{p}^{(k)}_{(\xi^{(k)})}\right\|^2\right] = 0. \tag{3.16}$$

And we observe that

$$\begin{aligned}
&\frac{1}{2}\mathbb{E}\left[\left\|\boldsymbol{P}^{(k)}_{\left(\xi^{(k)}\right)}\right\|^2\right]\\
\leq&\mathbb{E}\left[\left\|\boldsymbol{p}^{(k)}_{\left(\xi^{(k)}\right)}\right\|^2\right]+\mathbb{E}\left[\left\|\boldsymbol{p}^{(k)}_{\left(\xi^{(k)}\right)}-\boldsymbol{P}^{(k)}_{\left(\xi^{(k)}\right)}\right\|^2\right]\\
\leq&\mathbb{E}\left[\left\|\boldsymbol{p}^{(k)}_{\left(\xi^{(k)}\right)}\right\|^2\right]\\
&+\mathbb{E}_{\xi^{(k)},\mathcal{B}^{(k)}}\left[\mathbb{E}_{\zeta^{(k)}}\left[\left\|\nabla_{\boldsymbol{A}_{\left(\xi^{(k)}\right)}}f\left(\boldsymbol{A}^{(k)}_{(1)},\ldots,\boldsymbol{A}^{(k)}_{(N)}\right)-\boldsymbol{G}^{(k)}_{\left(\xi^{(k)}\right)}\right\|^2\mid\mathcal{B}^{(k)},\xi^{(k)}\right]\right]\\
\leq&\mathbb{E}\left[\left\|\boldsymbol{p}^{(k)}_{\left(\xi^{(k)}\right)}\right\|^2\right]+(\sigma^{(k)})^2,
\end{aligned}\tag{3.17}$$

where the last inequality is obtained via applying the non-expansiveness of the proximal operator property again. From Assumption 3.3 and Lemma 3.3, it holds that

$$\liminf_{k\to\infty}\ (\sigma^{(k)})^2=0.\tag{3.18}$$

Combined with (3.16), (3.17) and (3.18), we have

$$\liminf_{k\to\infty}\ \mathbb{E}\left[\left\|\boldsymbol{P}^{(k)}_{\left(\xi^{(k)}\right)}\right\|^2\right]=0.$$

Moreover, from the sampling rule (3.2) used for the mBrasCPD algorithm in this paper, we have

$$\begin{aligned}
\mathbb{E}\left[\left\|\boldsymbol{P}^{(k)}_{\left(\xi^{(k)}\right)}\right\|^2\right]=&\mathbb{E}_{\xi^{(k)},\mathcal{B}^{(k)}}\left[\mathbb{E}_{\zeta^{(k)}}\left[\left\|\boldsymbol{P}^{(k)}_{\left(\xi^{(k)}\right)}\right\|^2\mid\mathcal{B}^{(k)},\xi^{(k)}\right]\right]\\
=&\mathbb{E}_{\mathcal{B}^{(k)}}\left[\mathbb{E}_{\xi^{(k)}}\left[\left\|\boldsymbol{P}^{(k)}_{\left(\xi^{(k)}\right)}\right\|^2\mid\mathcal{B}^{(k)}\right]\right]\\
=&\mathbb{E}_{\mathcal{B}^{(k)}}\left[\sum_{n=1}^{N}\frac{1}{N}\left\|\boldsymbol{P}^{(k)}_{(n)}\right\|^2\right].
\end{aligned}$$

For any n, we have

$$\liminf_{k\to\infty}\ \mathbb{E}\left[\left\|\boldsymbol{P}^{(k)}_{(n)}\right\|^2\right]=0.$$

□

Remark 3.6. In Theorem 3.5, the condition

$$0\leq\alpha^{(k)}\beta^{(k+1)}<-(\alpha^{(k)})^2L+\alpha^{(k)}-\alpha^{(k-1)}\beta^{(k)}$$

is a little complicated, and it is difficult to be verified in the numerical experiments. However, there are several ways to simplify it. For example, if we fix $\beta^{(k)}=\beta\in[0,1]$ for any k, then this condition can be reduced to $-(\alpha^{(k)})^2L+\alpha^{(k)}-\alpha^{(k-1)}\beta-\alpha^{(k)}\beta>0$. Moreover, if we fix $\alpha^{(k)}=\alpha$ for any k, then we have $-\alpha L+1-2\beta>0$, i.e., $\alpha<\frac{1-2\beta}{L}$ with $\beta\in[0,\frac{1}{2})$ holds, then the convergence of the mBrasCPD algorithm is also satisfied.

Remark 3.7. The convergence analysis for mAdaCPD algorithm (Algorithm 2) is difficult. For example, the adaptive stepsize (2.5) is dependent on $\boldsymbol{G}_{(n)}^{(k)}$, then the decreased property of the objective function can not be obtained like the result of Lemma 3.4. Then it is beyond the scope of this paper. We hope to discuss it in future research work.

4 Numerical Experiments

In this section, we use some synthetic and real data numerical experiments to show the efficiency of the proposed algorithms. All of the experiments are implemented in MATLAB R2019b x64 on a PC with an Intel(k) Core(TM) i5-8265U CPU @ 1.60 GHz 1.80GHz processor and 8GB memory.

The numerical experiment performance is measured by the cost function value (1.3) (denotes by "NRE") and the mean squared error (MSE) which is defined as

$$\text{MSE} = \min_{\pi(r)\in\{1,\ldots,R\}} \frac{1}{R}\sum_{r=1}^{R}\left\| \frac{\boldsymbol{A}_{(n)}(:,\pi(r))}{\left\|\boldsymbol{A}_{(n)}(:,\pi(r))\right\|_2} - \frac{\bar{\boldsymbol{A}}_{(n)}(:,r)}{\left\|\bar{\boldsymbol{A}}_{(n)}(:,r)\right\|_2}\right\|^2,$$

where $\bar{\boldsymbol{A}}_{(n)}$ denotes the estimate of original matrix $\boldsymbol{A}_{(n)}$ and $\pi(r)$ satisfy $\{\pi(1),\ldots,\pi(R)\} = \{1,\ldots,R\}$ which is used to fix the intrinsic column permutation in CPD.

The nonnegative constraints ($\boldsymbol{A}_{(n)} \geq \boldsymbol{0}$) are applied to all the synthetic and real dataset experiments in this section for simplicity. Our algorithms are the accelerated version of BrasCPD and AdaCPD, thus we only compare with these two algorithms in the numerical experiments.

Theoretically, the stopping criterion of the proposed and compared algorithms is $\|\boldsymbol{P}_{(n)}^{(k)}\| \leq \epsilon$ (ϵ is small, for example $\epsilon = 1e-5$) with $n \in \{1,\ldots,N\}$. However, this stopping criterion is time-consuming to be verified in practice if I_n and N are large, because it takes a long time to calculate the full gradient. Thus all the algorithms are stopped after 50 iterations in numerical experiments for simplicity.

We do not compare the CPU time of the algorithms, because in this paper, we only compare our proposed mBrasCPD and mAdaCPD with their counterparts without momentum, i.e., BrasCPD and AdaCPD, respectively. Further, all four algorithms have similar algorithmic complexity because the major computational cost is to compute the gradient and implement the proximal operator, while computing the stepsize and other computations are negligible.

4.1 Synthetic data experiments

Here we conduct experiments to compare some algorithms for low rank tensor factorization in details on synthetic data. We use the third-order tensors (i.e., $N = 3$) whose latent factors are drawn from i.i.d. uniform distribution between 0 and 1. In order to illustrate the efficiency of the algorithm in this paper, large and dense tensors are generated.

We let $\alpha = 0.05$ to measure the numerical performance for the BrasCPD algorithm [11] and the mBrasCPD algorithm (Algorithm 1), respectively. We let $\beta = 0.2$ for our two algorithms[3] (mBrasCPD and mAdaCPD) in this subsection for simplicity. Four different tensor size ($I_n = 50$, $I_n = 100$, $I_n = 150$ and $I_n = 200$) with three different tensor ranks R are applied to the algorithms which are used in numerical experiments, respectively. And

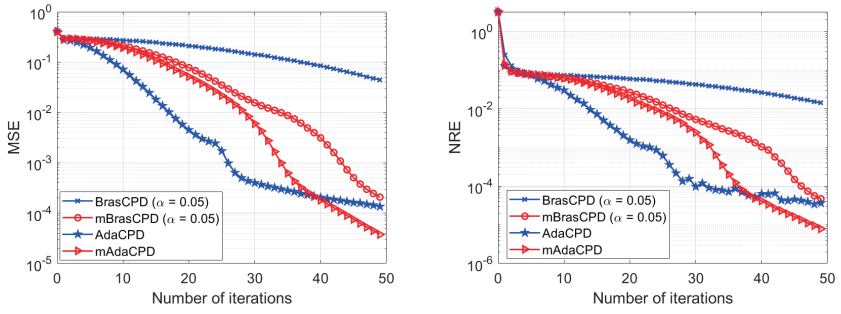
[3]If we want to get better numerical results, different values of α and β need to be properly selected for different data.

we set $|\mathcal{F}_{(n)}| = 20$ for $I_n = 100$ and $I_n = 200$ with $n = 1, 2, 3$, $|\mathcal{F}_{(n)}| = 25$ for $I_n = 50$ and $I_n = 150$ with $n = 1, 2, 3$. At each combination of these hyper-parameters, we repeat the numerical experiments 10 times and use their mean values as the results of these experiments. All the compared algorithms use the same data in each trial.

Table 1: MSE and NRE of the estimated latent factors by the algorithms under different I_1, I_2, I_3 and R. "a" denotes BrasCPD [11] with $\alpha = 0.05$; "b" denotes mBrasCPD with $\alpha = 0.05$ and $\beta = 0.2$; "c" denotes AdaCPD [11]; "d" denotes mAdaCPD with $\beta = 0.2$.

(I_1, I_2, I_3)	R	MSE				NRE			
		a	b	c	d	a	b	c	d
(50,50,50)	10	1.6e-1	1.2e-2	2.2e-4	3.5e-5	1.1e-2	6.2e-3	1.1e-5	1.1e-6
	20	2.0e-1	1.7e-2	1.2e-2	1.0e-3	2.0e-2	1.4e-2	8.6e-4	1.1e-4
	50	2.4e-1	2.2e-1	1.5e-1	1.2e-1	4.6e-2	3.8e-2	3.4e-2	2.5e-2
(100,100,100)	10	6.5e-2	9.4e-4	2.2e-13	1.6e-14	3.0e-3	4.8e-5	7.1e-15	4.8e-16
	30	1.6e-1	7.2e-2	1.3e-4	4.8e-5	1.9e-2	8.3e-3	1.6e-5	4.2e-6
	70	2.4e-1	1.6e-1	1.1e-1	5.6e-2	5.6e-2	4.1e-2	2.9e-2	1.7e-2
(150,150,150)	20	1.9e-2	1.7e-4	2.4e-16	1.2e-16	1.6e-3	9.4e-6	2.0e-20	4.7e-22
	50	1.0e-1	6.5e-3	2.4e-4	2.0e-5	2.0e-2	1.7e-3	4.6e-5	2.6e-6
	150	6.0e-2	5.3e-2	1.7e-1	9.1e-2	1.0e-1	3.4e-2	9.5e-2	5.5e-2
(200,200,200)	20	9.6e-5	3.1e-7	4.4e-15	1.4e-16	6.3e-6	2.0e-8	2.5e-27	8.8e-31
	50	8.9e-3	3.6e-5	4.4e-5	3.5e-8	1.9e-3	5.2e-6	7.5e-6	6.3e-9
	80	4.4e-2	1.5e-4	1.4e-4	3.8e-5	1.4e-2	3.3e-5	3.7e-5	7.9e-6

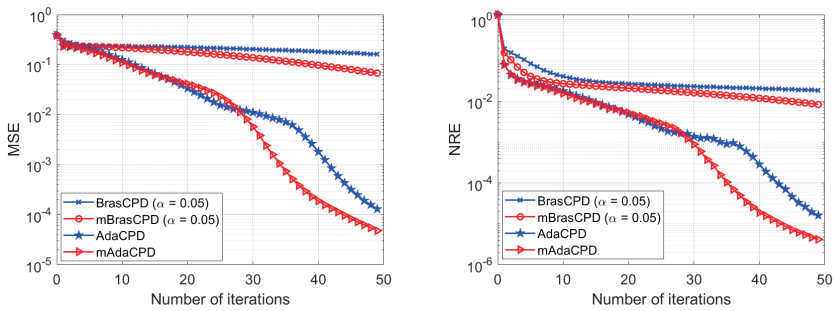
The details of the latent factors result with different parameters (I_1, I_2, I_3 and R) are referred to Table 1[4]. It is obvious from Table 1 that our algorithms (" b " and " d " in this table) can get smaller MSE and NRE values than the other two algorithms (" a " and " c " in this table), respectively. We note that the momentum technique used in this paper has indeed improved the results of numerical experiments. We find that the adaptive stepsize based algorithms have better numerical results compared with constant stepsize algorithms in general. Two numerical results in Table 1 are detailed in Figure 2 and 3, respectively.



(a) (b)

Figure 2: Tensor factorization on synthetic data with $I_1 = I_2 = I_3 = 200$ and $R = 80$.

[4]As the rank of the tensor increases, the solution space generally becomes larger. This reason leads to the fact that when the rank of the tensor is large, its numerical performance is not as good as when the rank of the tensor is small when the number of iteration is small.



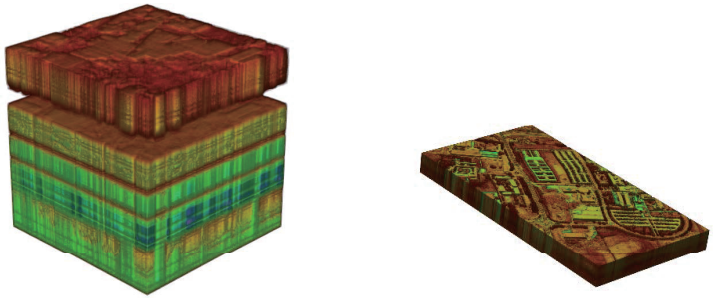
(a) (b)

Figure 3: Tensor factorization on synthetic data with $I_1 = I_2 = I_3 = 100$ and $R = 30$.

4.2 Real data experiments

Hyperspectral images (HSI) are obtained by images of airborne or satellite sensors on a target area, which contains information of objects in tens to hundreds of consecutive and segmented bands from visible light to the infrared spectral region. An HSI is usually stored as a third-order tensor with two spatial coordinates and one spectral coordinate.

Due to the memory limit of our PC, the sub-images of two HSI datasets[5] (the Indian Pines dataset with $145 \times 145 \times 220$ and the Pavia University dataset with $610 \times 340 \times 103$) are used in this subsection. See Figure 4 for details. For HSI numerical experiments, we use the NRE and the structural similarity (SSIM)[6] [32] values to measure the numerical performance of the algorithms.



(a) (b)

Figure 4: (a): The Indian Pines dataset with $145 \times 145 \times 145$. (b): The Pavia University dataset with $610 \times 340 \times 50$.

We repeat the experiments 10 times and 5 times for the Indian Pines dataset and the Pavia University dataset, respectively. And we set $|\mathcal{F}_{(n)}| = 25$ for these two dataset and use the same parameters (α and β) from Subsection 4.1. The details of the results are referred to Table 2. We draw the curve of the NRE values for the experiments of the Indian Pines with $I_1 = I_2 = I_3 = 145$ and $R = 50$ with the iteration of the algorithm in Figure 5.

[5] http://www.ehu.eus/ccwintco/index.php/Hyperspectral_Remote_Sensing_Scenes.
[6] http://www.ece.uwaterloo.ca/~z70wang/research/ssim/

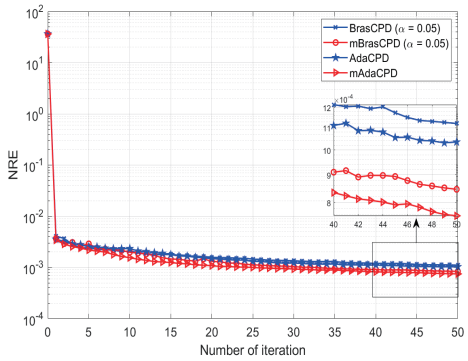


Figure 5: Tensor factorization on the Idian Pines dataset with $I_1 = I_2 = I_3 = 145$ and $R = 50$.

It is obvious from Table 2 that our algorithms ("b" and "d" in this table) can get better NRE and SSIM values than the other two algorithms ("a" and "c" in this table).

Table 2: NRE and SSIM of the estimated latent factors by the algorithms under different R on the HSI data. "a" denotes BrasCPD [11] with $\alpha = 0.05$; "b" denotes mBrasCPD with $\alpha = 0.05$ and $\beta = 0.2$; "c" denotes AdaCPD [11]; "d" denotes mAdaCPD with $\beta = 0.2$.

Dataset	R	NRE				SSIM			
		a	b	c	d	a	b	c	d
Indian Pines	10	1.2e-3	1.1e-3	1.1e-3	1.0e-3	0.8451	0.8552	0.8550	0.8598
	20	1.1e-3	9.5e-4	1.0e-3	9.4e-4	0.8473	0.8637	0.8572	0.8675
$145 \times 145 \times 145$	50	1.1e-3	8.4e-4	1.0e-3	7.5e-4	0.8411	0.8699	0.8483	0.8785
	100	1.0e-3	8.9e-4	9.3e-4	8.3e-4	0.8434	0.8675	0.8525	0.8721
Pavia University	10	3.6e-3	3.4e-3	3.5e-3	3.4e-3	0.5754	0.5880	0.5876	0.5972
	30	3.3e-3	2.4e-3	2.4e-3	1.8e-3	0.6550	0.6736	0.6916	0.7027
$610 \times 340 \times 50$	70	2.1e-3	1.5e-3	1.1e-3	9.9e-4	0.6962	0.7179	0.7705	0.7997
	100	2.2e-3	1.5e-3	8.6e-4	7.8e-4	0.7059	0.7202	0.8220	0.8373

From the synthetic and real data numerical experiments, one can see that the momentum technique can get better results compared with the algorithms under non momentum framework.

5 Conclusion

In this paper, we proposed two momentum block-randomized stochastic gradient descent algorithms under constant and adaptive stepsize framework for low-rank CP tensor factorization problem, respectively. The convergence analysis of the mBrasCPD algorithm for the low-rank tensor CPD problem was given under some mild conditions. Compared with algorithms without momentum, the numerical simulation results for the synthetic and real data demonstrated that our algorithms are more efficient.

Acknowledgment

The authors would like to appreciate two anonymous reviewers for their insightful comments and constructive suggestions to polish this paper in high quality.

References

[1] A. Anandkumar, R. Ge, D. J. Hsu, S. M. Kakade, and M. Telgarsky, Tensor decompositions for learning latent variable models, *J. Mach. Learn. Res.* 15 (2014) 2773–2832.

[2] C. Battaglino, G. Ballard, and T. G. Kolda, A practical randomized CP tensor decomposition, *SIAM J. Matrix Anal. A.* 39 (2018) 876–901.

[3] A. Beck and L. Tetruashvili, On the convergence of block coordinate descent type methods, *SIAM J. Optimiz.* 23 (2013) 2037–2060.

[4] A. Beutel, P. P. Talukdar, A. Kumar, C. Faloutsos, E. E. Papalexakis, and E. P. Xing, Flexifact: Scalable flexible factorization of coupled tensors on hadoop, *In Proceedings of the 2014 SIAM International Conference on Data Mining* (2014) 109–117.

[5] L. Bottou, Large-scale machine learning with stochastic gradient descent, *In 19th International Conference on Computational Statistics*, (2010) 177–186.

[6] J. Chen and Y. Saad, On the tensor SVD and the optimal low rank orthogonal approximation of tensors, *SIAM J. Matrix Anal. A.* 30 (2009) 1709-1734.

[7] Y. Chen, Y. Dai, and D. Han, Fiber orientation distribution estimation using a peaceman-rachford splitting method, *SIAM J. Imaging Sci.* 9 (2016) 573–604.

[8] Y. Chen, Y. Dai, D. Han, and W. Sun, Positive semidefinite generalized diffusion tensor imaging via quadratic semidefinite programming, *SIAM J. Imaging Sci.* 6 (2013) 1531–1552.

[9] V. de Silva and L. Lim, Tensor rank and the ill-posedness of the best low-rank approximation problem, *SIAM J. Matrix Anal. A.* 30 (2008) 1084–1127.

[10] J. C. Duchi, E. Hazan, and Y. Singer, Adaptive subgradient methods for online learning and stochastic optimization, *J. Mach. Learn. Res.* 12 (2011) 2121–2159.

[11] X. Fu, S. Ibrahim, H. Wai, C. Gao, and K. Huang, Block-randomized stochastic proximal gradient for low-rank tensor factorization, *IEEE Trans. Signal Proces.* 68 (2020) 2170–2185.

[12] S. Ghadimi and G. Lan, Stochastic first- and zeroth-order methods for nonconvex stochastic programming, *SIAM J. on Optim.* 23 (2013) 2341–2368.

[13] S. Ghadimi and G. Lan, Accelerated gradient methods for nonconvex nonlinear and stochastic programming, *Math. Program.* 156 (2016) 59–99.

[14] K. Huang, N. D. Sidiropoulos, and A. P. Liavas, A flexible and efficient algorithmic framework for constrained matrix and tensor factorization, *IEEE Trans. Signal Proces.* 64 (2016) 5052–5065.

[15] F. Jiang, D. Han, and X. Zhang, A trust-region-based alternating least-squares algorithm for tensor decompositions, *J. Comput. Math.* 36 (2018) 351–372.

[16] D. P. Kingma and J. Ba, Adam: A method for stochastic optimization, *In 3rd International Conference on Learning Representations*, 2015.

[17] T. G. Kolda and B. W. Bader, Tensor decompositions and applications, *SIAM Revi.* 51 (2009) 455–500.

[18] W. Liu, L. Chen, Y. Chen, and W. Zhang, Accelerating federated learning via momentum gradient descent, *IEEE Trans. Parall. Distr.* 31 (2020) 1754–1766.

[19] T. Maehara, K. Hayashi, and K. Kawarabayashi, Expected tensor decomposition with stochastic gradient descent, *In Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, (2016) 1919–1925.

[20] J. Mairal, F. R. Bach, J. Ponce, and G. Sapiro, Online learning for matrix factorization and sparse coding, *J. Mach. Learn. Res.* 11 (2010) 19–60.

[21] D. Mitchell, N. Ye, and H. D. Sterck, Nesterov acceleration of alternating least squares for canonical tensor decomposition: Momentum step size selection and restart mechanisms, *Numer. Linear Algebr.* 27 (2020).

[22] Y. E. Nesterov, Efficiency of coordinate descent methods on huge-scale optimization problems, *SIAM J. Optimi.* 22 (2012) 341–362.

[23] E. E. Papalexakis, C. Faloutsos, and N. D. Sidiropoulos, Parcube: Sparse parallelizable tensor decompositions, *In Machine Learning and Knowledge Discovery in Databases -European Conference*, 7523 (2012) 521–536.

[24] B. T. Polyak, Some methods of speeding up the convergence of iteration methods, *USSR Computational Mathematics and Mathematical Physics*, 4 (1964) 1–17.

[25] W. Qin, H. Wu, Q. Lai, and C. Wang, A parallelized, momentum-incorporated stochastic gradient descent scheme for latent factor analysis on high-dimensional and sparse matrices from recommender systems, *In 2019 IEEE International Conference on Systems, Man and Cybernetics*, (2019) 1744–1749.

[26] M. Razaviyayn, M. Hong, and Z. Luo, A unified convergence analysis of block successive minimization methods for nonsmooth optimization, *SIAM J. Optimi.* 23 (2013) 1126–1153.

[27] H. Robbins and S. Monro, A stochastic approximation method, *Ann. Math. Stat.* 22 (1951) 400–407.

[28] L. Sun, F. Wu, T. Zhan, W. Liu, J. Wang, and B. Jeon, Weighted nonlocal lowrank tensor decomposition method for sparse unmixing of hyperspectral images, *IEEE J-Stars.* 13 (2020) 1174–1188.

[29] Y. Tsitsikas and E. E. Papalexakis, NSVD: normalized singular value deviation reveals number of latent factors in tensor decomposition, *Big Data.* 8 (2020) 412–430.

[30] N. Vervliet and L. D. Lathauwer, A randomized block sampling approach to canonical polyadic decomposition of large-scale tensors. *IEEE J-STSP.* 10 (2016) 284–295.

[31] M. D. Vos, A. Vergult, L. D. Lathauwer, W. D. Clercq, S. V. Huffel, P. Dupont, A. Palmini, and W. V. Paesschen, Canonical decomposition of ictal scalp EEG reliably detects the seizure onset zone. *NeuroImage.* 37 (2017) 844–854.

[32] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, Image quality assessment: From error visibility to structural similarity. *IEEE Trans. Image Proces.* 13 (2004) 600-612.

[33] B. Weng, H. Xiong, Y. Liang, and W. Zhang, Analysis of Q-learning with adaptation and momentum restart for gradient descent, *In Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, (2020) 3051–3057.

[34] Y. Xu, Z. Wu, J. Chanussot, and Z. Wei, Hyperspectral computational imaging via collaborative tucker3 tensor decomposition, *IEEE Trans. Circ. Syst. Vid.* 31 (2021) 98–111.

[35] Y. Xu and W. Yin, A block coordinate descent method for regularized multiconvex optimization with applications to nonnegative tensor factorization and completion, *SIAM J. on Imaging Sci.* 6 (2013) 1758–1789.

Manuscript received 1 April 2021
revised 13 May 2021
accepted for publication 20 May 2021

Qingsong Wang
LMIB of the Ministry of Education
School of Mathematical Sciences
Beihang University
Beijing, 100191, People's Republic of China
E-mail address: nothing2wang@hotmail.com

Chunfeng Cui
LMIB of the Ministry of Education
School of Mathematical Sciences
Beihang University
Beijing, 100191, People's Republic of China
E-mail address: chunfengcui@buaa.edu.cn

Deren Han
LMIB of the Ministry of Education
School of Mathematical Sciences
Beihang University
Beijing, 100191, People's Republic of China
E-mail address: handr@buaa.edu.cn