Numerical Algorithms (2024) 96:1755–1778
https://doi.org/10.1007/s11075-023-01684-w

ORIGINAL PAPER

Tensor randomized extended Kaczmarz methods for large inconsistent tensor linear equations with t-product

Guang-Xin Huang[1] · **Shuang-You Zhong**[2]

Received: 16 August 2023 / Accepted: 11 October 2023 / Published online: 8 November 2023

Abstract
This paper presents three tensor randomized extended Kaczmarz methods for solving a tensor inconsistent linear system of equations under the t-product between tensors. The randomized extended Kaczmarz, the randomized extended block Kaczmarz, and the randomized extended greedy block Kaczmarz methods in tensor form are proposed to solve inconsistent tensor linear system of equations, respectively. The convergence of each method is proved. Several numerical examples are given to show the efficiency and effectiveness of our methods.

Keywords Tensor equations · T-product · Inconsistent · Convergence

Mathematics Subject Classification (2010) 65F10 · 65F20

1 Introduction

In this paper, we are mainly concerned with solving the large inconsistent tensor linear system of equations of the form

$$\mathscr{A} * \mathscr{X} = \mathscr{B}, \tag{1}$$

where $*$ denotes the t-product similar to that in [1, 2], the blurred tensor $\mathscr{A} \in \mathbb{C}^{m\times l\times n}$, the data tensor $\mathscr{B} \in \mathbb{C}^{m\times p\times n}$ is contaminated by measurement error or noise that is

✉ Guang-Xin Huang
huangx@cdut.edu.cn

Shuang-You Zhong
uu547274611@163.com

1 College of Computer Science and Cyber Security, Sichuan Geomathematics Key Laboratory, Chengdu University of Technology, Chengdu 610059, People's Republic of China

2 College of Mathematics and Physics, Sichuan Geomathematics Key Laboratory, Chendu University of Technology, Chengdu 610059, People's Republic of China

represented by a tensor $\epsilon \in \mathbb{C}^{m\times p\times n}$, i.e.,

$$\mathscr{B} = \mathscr{B}_{true} + \epsilon,$$

where $\mathscr{B}_{true} \in \mathbb{R}^{m\times l\times n}$ is an unknown and unavailable error-free tensor related to $\mathscr{B}$, and $\mathscr{B}_{true}$ is determined by $\mathscr{A} * \mathscr{X}_{true} = \mathscr{B}_{true}$, where $\mathscr{X}_{true}$ represents the explicit solution of Eq. (1) that is to be found. Since the noise ϵ will be heavily amplified, it is difficult to solve Eq. (1) directly. As we all know, a color image, a frame of video, and time data can all be represented by three-dimensional arrays. Many methods of dealing with tensors matricize or vectorize tensors, which destroys their original structure. The emergence of t-product avoids the occurrence of such a situation, which greatly improves the efficiency of some algorithms on tensors. The t-product has applications in many fields, for example, it is widely used in image deblurring problems [1–3], face recognition [4], neural networks [5], and tomographic image reconstruction [6].

In the setting of the matrix system $Ax = b$, the randomized Kaczmarz (RK) method is a good choice to solve a system of matrix linear equations. And for each iteration of the RK method, a row i_k is selected with the probability $\|A_{i_k:}\|_2^2/\|A\|_F^2$. To speed up the convergence, some extensions have been made to the RK method, such as the block [7–10] and greedy [11–14] variants. And for the matrix inconsistent linear systems, Needell [15] proved that the solution vector obtained by the RK method is within a fixed distance from the least squares solution and also that this distance is proportional to the distance of b from $b_{R(A)}$, where $b_{R(A)}$ denotes the column space of A. To solve this situation, some Kaczmarz type iterative methods have been proposed, for example, randomized extended Kaczmarz (REK) [16], partially randomized extended Kaczmarz (PREK) [17], and randomized extended average block Kaczmarz (READBK) [18]. For the REK method, each iteration consists of two parts:

$$z^k = z^{k-1} - \frac{(A_{:j})^T z^{k-1}}{\|A_{:j}\|_2^2} A_{:j},$$
$$x^k = x^{k-1} + \frac{b_i - A_{i:}x_k - z_i^k}{\|A_{i:}\|_2^2} (A_{i:})^T,$$

where i, j is randomly selected according to the probability $\|A_{i:}\|_2^2/\|A\|_F^2$ and $\|A_{:j}\|_2^2/\|A\|_F^2$, respectively. The first part of the REK method is to solve the equation $A^T z = 0$ to get an approximation of $b_{R(A)}$, which is $b - z^k$, the second part of the REK method is to solve the equation $Ax = b - z^k$ to obtain an approximate solution of original equations. And the resulting approximate solution converges to the unique least squares solution of the original equations. Notice that the first part uses orthogonal projections to solve the equation $A^T z = 0$, and the second part uses the RK method to solve the equation $Ax = b - z^k$. The READBK method is obtained by blocking the REK method, which greatly improves the convergence speed. It can be

expressed in the following form:

$$z^k = z^{k-1} - \frac{\alpha}{\|A_{:J_j}\|_2^2} A_{:J_j} (A_{:J_j})^T z^{k-1},$$
$$x^k = x^{k-1} + \frac{\alpha}{\|A_{I_i:}\|_2^2} (A_{I_i:})^T (b_{I_i:} - A_{I_i:} x_k - z^k_{I_i:}),$$

where α is the stepsize, block I_i,J_j is randomly selected according to the probability $\|A_{I_i:}\|_2^2/\|A\|_F^2$ and $\|A_{:J_j}\|_2^2/\|A\|_F^2$ respectively. Recently, Ma and Molitor in [15] proposed the tensor randomized Kaczmarz (TRK) method, which extends the RK method in matrix form to solve the tensor linear system of equations. It is superior in terms of computation for tensor systems when compared to unfolding the tensor and applying the matrix version of RK. In this paper, inspired by this idea, we present the REK and REABK methods in tensor form to solve the inconsistent tensor linear system of Eq. (1), and the proposed methods are called tensor randomized extended Kaczmarz (TREK) method and tensor randomized extended block Kaczmarz (TREBK) method, respectively. In addition, we apply the idea of almost-maximum residual block in [19, 20] to the first part of the TREBK method, and thus obtain a tensor randomized extended greedy block Kaczmarz (TREGBK). Finally, we also give the convergence theory of these methods.

Table 1 Symbols commonly used in this paper

Natation	Description	Used in section
$\mathscr{A}$	Tensor	1-4
A	Matrix	1,4
b	Vector	1
b_i	The ith entry of vector b	1
$A_{i:}$	The ith row of A	1
$A_{:j}$	The jth column of A	1
$\mathscr{A}_{k::}$	The kth horizontal slice of $\mathscr{A}$	2-3
$\mathscr{A}_{:k}$ or $\mathbf{A}_k$	The kth frontal slice of $\mathscr{A}$	2-4
$\mathscr{A}^*$	Transpose of tensor $\mathscr{A}$	2-4
$\mathscr{A}^\dagger$	Moore-Penrose inverse of tensor $\mathscr{A}$	2-4
$\|A\|_F^2$	$\sum A_{ij}^2$	2-4
$\|\mathscr{A}\|_F^2$	$\sum \mathscr{A}_{ijk}^2$	3
$\sigma_{min}(A)$	The smallest nonzero singular value of A	3
$\sigma_{max}(A)$	The maximal nonzero singular value of A	3
$[m]$	$\{1, 2, \cdots, m\}$	2
$*$	t-product	2-4
$\cdot$	Matrix-matrix product	3
$\otimes$	Kronecker product	2

The rest of this paper is organized as follows. Section 2 introduces the some definitions ans notations that will be used in the context. In Sect. 3, we present the TREK, TREBK, and TREGBK methods and give the convergence. Several examples are provided to discuss the properties of the TREK,TREBK and TREGBK methods in Sect. 4 and some conclusions are drawn in Sect. 5.

2 Preliminary

In this section, we mainly introduce the background of linear algebra related to the tensor t-product. Table 1 lists these notations. Figure 1 shows the kth horizontal slice and the kth frontal slice of a third-order tensor $\mathscr{A}$, respectively. And in this article, the operator between tensors is the t-product, we omit the t-product symbol, the reader can judge according to the context. Eq. (2) shows the conversion process between a tensor $\mathscr{A} \in \mathbb{C}^{m\times l\times n}$ and a matrix $A \in \mathbb{C}^{mn\times l}$,

$$\mathscr{A} = \text{fold}(\text{unfold}(\mathscr{A})) = \text{fold}\left(\begin{bmatrix} \mathscr{A}_{::1} \\ \vdots \\ \mathscr{A}_{::n} \end{bmatrix}\right) = \text{fold}\left(\begin{bmatrix} \mathbf{A}_1 \\ \vdots \\ \mathbf{A}_n \end{bmatrix}\right), \tag{2}$$

where $A = (\mathbf{A}_1^T, \mathbf{A}_2^T, \cdots, \mathbf{A}_n^T)$.

Definition 1 For a tensor $\mathscr{A} \in \mathbb{C}^{m\times l\times n}$, the block-circulant matrix $\text{bcirc}(\mathscr{A}) \in \mathbb{C}^{mn\times ln}$ is denoted by

$$\text{bcirc}(\mathscr{A}) = \begin{bmatrix} \mathbf{A}_1 & \mathbf{A}_n & \dots & \mathbf{A}_2 \\ \mathbf{A}_2 & \mathbf{A}_1 & \dots & \mathbf{A}_3 \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{A}_n & \mathbf{A}_{n-1} & \dots & \mathbf{A}_1 \end{bmatrix}.$$

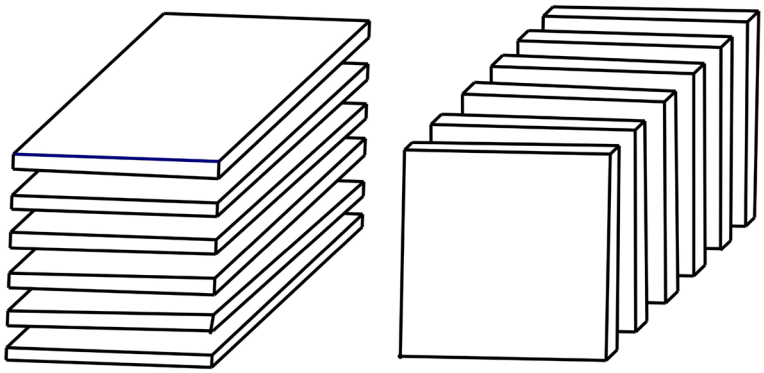

Fig. 1 **a** horizontal slices and **b** frontal slices of a third-order tensor $\mathscr{A}$

Definition 2 Let tensor $\mathscr{A} \in \mathbb{C}^{m\times l\times n}$ and $\mathscr{B} \in \mathbb{C}^{l\times p\times n}$, then the t-product between $\mathscr{A}$ and $\mathscr{B}$ is

$$\mathscr{A} * \mathscr{B} = \text{fold}(\text{bcirc}(\mathscr{A}) \cdot \text{unfold}(\mathscr{B})) \in \mathbb{C}^{m\times p\times n}.$$

The block diagonalization of block circulant matrix can be achieved by the normalized discrete Fourier transform (DFT) [2] matrix combined with the Kronecker product. Suppose $\mathscr{A} \in \mathbb{C}^{m\times l\times n}$ and F_n is the normalized discrete Fourier transform matrix, then

$$\text{bdiag}(\widehat{\mathscr{A}}) = (F_n \otimes I_m)\text{bcirc}(\mathscr{A})(F_n^* \otimes I_l) = \begin{bmatrix} \widehat{\mathbf{A}}_1 & & & \\ & \widehat{\mathbf{A}}_2 & & \\ & & \ddots & \\ & & & \widehat{\mathbf{A}}_n \end{bmatrix}, \tag{3}$$

where $\widehat{\mathscr{A}}$ is obtained by applying FFT along each tube fiber of $\mathscr{A}$, F_n^* is the transform of F_n, and I denotes the identity matrix. So the t-product $\mathscr{A} * \mathscr{B}$ can be computed efficiently by using the transformation Eq. (3), for example:

$$\mathscr{A} * \mathscr{B} = fold((F_n^* \otimes I_l)\text{bdiag}(\widehat{\mathscr{A}})(F_n \otimes I_m) \cdot unfold(\mathscr{B})).$$

In MATLAB, the t-product $\mathscr{A} * \mathscr{B}$ can be computed by first getting the tensor $\widehat{\mathscr{A}} = \text{fft}(\mathscr{A}, [], 3)$ and $\widehat{\mathscr{B}} = \text{fft}(\mathscr{B}, [], 3)$ in the Fourier domain, multiplying each pair of the frontal slices of $\widehat{\mathscr{A}}$ and $\widehat{\mathscr{B}}$,

$$\widehat{\mathscr{D}}(:,:,i) = \widehat{\mathscr{A}}(:,:,i) \cdot \widehat{\mathscr{B}}(:,:,i), \quad i = 1, 2\cdots, n.$$

and then taking the tensor $\widehat{\mathscr{D}}$ out of Fourier domain to obtain $\mathscr{A} * \mathscr{B} = \text{ifft}(\widehat{\mathscr{D}}, [], 3)$.

3 The tensor randomized extended Kaczmarz method

For inconsistent tensor linear system of Eq. (1), there is currently no tensor method of type Kaczmarz to solve it. So we extend the Kaczmarz type methods used on matrices to tensors to solve the inconsistent tensor problem. For the REK method in matrix form, we extend its two parts directly to the tensor, and call the resulting in tensor method as a tensor randomized extended Kaczmarz (TREK) method. Making use of the properties of the t-product, the TREK method for inconsistent tensor linear equations can be written as

$$\begin{aligned} \mathscr{Z}^k &= \mathscr{Z}^{k-1} - \mathscr{A}_{:j:}(\mathscr{A}_{:j:}^*\mathscr{A}_{:j:})^\dagger \mathscr{A}_{:j:}^* \mathscr{Z}^{k-1}, \\ \mathscr{X}^k &= \mathscr{X}^{k-1} - \mathscr{A}_{i::}^*(\mathscr{A}_{i::}\mathscr{A}_{i::}^*)^\dagger(\mathscr{A}_{i::}\mathscr{X}^{k-1} - \mathscr{B}_{i::} + \mathscr{Z}_{i::}^k), \end{aligned}$$

there $\mathscr{A}_{i::}$,$\mathscr{A}_{:j:}$ are the ith frontal and jth lateral slices of the tensor $\mathscr{A}$ respectively, the index i, j are selected according to the probability $\|\mathscr{A}_{i::}\|_2^2/\|\mathscr{A}\|_F^2$ and $\|\mathscr{A}_{:j:}\|_2^2/\|\mathscr{A}\|_F^2$ respectively. Algorithm 1 summarizes the TREK method.

Algorithm 1 A tensor randomized extended Kaczmarz algorithm (TREK).

Require: $\mathscr{A} \in \mathbb{C}^{m\times l\times n}$, $\mathscr{B} \in \mathbb{C}^{m\times p\times n}$, the maximum number of iterations T.
Ensure: the approximation solution $\mathscr{X}^k$ of $\mathscr{A}\mathscr{X} = \mathscr{B}$.
1: Initialize $\mathscr{X}^0 = 0$ and $\mathscr{Z}^0 = \mathscr{B}$.
2: **for** $k = 1, 2, \cdots, T$ **do**
3: Pick $j \in [l]$ with probability $\|\mathscr{A}_{:j:}\|_F^2/\|\mathscr{A}\|_F^2$,
4: $\mathscr{Z}^k = \mathscr{Z}^{k-1} - \mathscr{A}_{:j:}(\mathscr{A}_{:j:}^*\mathscr{A}_{:j:})^\dagger \mathscr{A}_{:j:}^*\mathscr{Z}^{k-1}$,
5: Pick $i \in [m]$ with probability $\|\mathscr{A}_{i::}\|_F^2/\|\mathscr{A}\|_F^2$,
6: $\mathscr{X}^k = \mathscr{X}^{k-1} - \mathscr{A}_{i::}^*(\mathscr{A}_{i::}\mathscr{A}_{i::}^*)^\dagger(\mathscr{A}_{i::}\mathscr{X}^{k-1} - \mathscr{B}_{i::} + \mathscr{Z}_{i::}^k)$.
7: **end for**

We can derive the convergence of the TREK method as follows similarly to that of the TRK method. The proof is in the appendix.

Theorem 1 *Let $\mathscr{X}_*$ be the true solution tensor of the inconsisitent tensor linear system of Eq. (1) and $\mathscr{X}^k$ be the kth approximation of $\mathscr{X}_*$ given by applying TREK to Eq. (1) with an initial iterate $\mathscr{X}^0$. The expected error at the kth iteration satisfies*

$$\mathbb{E}\big[\|\mathscr{X}^k - \mathscr{X}_*\|_F^2\big] \le (1-\beta_1)^k\|\mathscr{X}^0 - \mathscr{X}_*\|_F^2 + \frac{\sigma_{max}^2(\mathbb{E}(bcirc(\mathscr{R}_i)))(1-\beta_2)^k}{\beta_1}\|\mathscr{Z}^0\|_F^2,$$

where $\sigma_{\min}(\cdot)$ and $\sigma_{\max}(\cdot)$ denote the smallest and maximal nonzero singular value. $\mathscr{Q}_j = \mathscr{A}_{:j:}(\mathscr{A}_{:j:}^*\mathscr{A}_{:j:})^{-1}\mathscr{A}_{:j:}^*$, $\mathscr{R}_i = \mathscr{A}_{i::}^*(\mathscr{A}_{i::}\mathscr{A}_{i::}^*)^{-1}$, $\mathscr{P}_i = \mathscr{A}_{i::}^*(\mathscr{A}_{i::}\mathscr{A}_{i::}^*)^{-1}\mathscr{A}_{i::}$, $\beta_1 = \sigma_{min}^2(\mathbb{E}(bcirc(\mathscr{P}_i)))$, $\beta_2 = \sigma_{min}^2(\mathbb{E}(bcirc(\mathscr{Q}_j)))$.

The TREK method is able to solve some inconsistent tensor linear problems such as Eq. (1). However, since the TREK method only selects a horizontal slice to iterate over during each iteration, its calculation speed is not ideal. To improve the calculation speed of the TREK method, we propose the block variant of the TREK method and name it tensor randomized extended block Kaczmarz (TREBK). Based on given partitions of [m] and [n], the iteration of the TREBK method can be carried out by the following equations:

$$\begin{aligned}\mathscr{Z}^k &= \mathscr{Z}^{k-1} - \mathscr{A}_{:J_j:}\mathscr{A}_{:J_j:}^\dagger\mathscr{Z}^{k-1},\\ \mathscr{X}^k &= \mathscr{X}^{k-1} - \mathscr{A}_{I_i::}^\dagger(\mathscr{A}_{I_i::}\mathscr{X}^{k-1} - \mathscr{B}_{I_i::} + \mathscr{Z}_{I_i::}^k),\end{aligned}$$

where $\mathscr{A}_{I_i::}$,$\mathscr{A}_{:J_j:}$ are the subtensor of the tensor $\mathscr{A}$, index I_i,J_j are selected according to the probability $\|\mathscr{A}_{I_i::}\|_2^2/\|\mathscr{A}\|_F^2$ and $\|\mathscr{A}_{:J_j:}\|_2^2/\|\mathscr{A}\|_F^2$ respectively. The TREBK method is detailed in Algorithm 2.

Similar to the TREK method, we can similarly derive the convergence theory of the TREBK method.

Theorem 2 *Let $\mathscr{X}_*$ be the true solution tensor of the inconsisitent tensor linear system Eq. (1) and $\mathscr{X}^k$ be the kth approximation of $\mathscr{X}_*$ given by applying TREBK to Eq. (1)*

Algorithm 2 Tensor randomized extended block Kaczmarz algorithm.

Require: $\mathscr{A} \in \mathbb{C}^{m\times l\times n}$, $\mathscr{B} \in \mathbb{C}^{m\times p\times n}$, the maximum number of iterations T.
Ensure: the approximation solution $\mathscr{X}^k$ of $\mathscr{A}\mathscr{X} = \mathscr{B}$.
1: Let$\{I_1, I_2, \cdots, I_s\}$ and $\{J_1, J_2, \cdots, J_t\}$ be partitions of $[m]$ and $[l]$, respectively.
2: Initialize $\mathscr{X}^0 = 0$ and $\mathscr{Z}^0 = \mathscr{B}$.
3: **for** $k = 1, 2, \cdots, T$ **do**
4: Pick $j \in [t]$ with probability $\|\mathscr{A}_{:J_j:}\|_F^2/\|\mathscr{A}\|_F^2$,
5: $\mathscr{Z}^k = \mathscr{Z}^{k-1} - \mathscr{A}_{:J_j:}(\mathscr{A}_{:J_j:})^\dagger \mathscr{Z}^{k-1}$,
6: Pick $i \in [s]$ with probability $\|\mathscr{A}_{I_i::}\|_F^2/\|\mathscr{A}\|_F^2$,
7: $\mathscr{X}^k = \mathscr{X}^{k-1} - (\mathscr{A}_{I_i::})^\dagger(\mathscr{A}_{I_i::}\mathscr{X}^{k-1} - \mathscr{B}_{I_i::} + \mathscr{Z}^k_{I_i::})$.
8: **end for**

with an initial iterate $\mathscr{X}^0$*. The expected error at the* k*th iteration satisfies*

$$\mathbb{E}\left[\|\mathscr{X}^k - \mathscr{X}_*\|_F^2\right] \le (1-\beta_3)^k\|\mathscr{X}^0 - \mathscr{X}_*\|_F^2 + \frac{\sigma_{min}^2(\mathbb{E}(bcirc(\mathscr{A}_{I_i::})))(1-\beta_4)^k}{\beta_3}\|\mathscr{Z}^0\|_F^2,$$

where $\sigma_{\min}(\cdot)$ *and* $\sigma_{\max}(\cdot)$ *denote the smallest and maximal nonzero singular value.* $\mathscr{Q}_{J_j} = \mathscr{A}_{:J_j:}(\mathscr{A}_{:J_j:})^\dagger$, $\mathscr{P}_{I_i} = (\mathscr{A}_{I_i::})^\dagger\mathscr{A}_{I_i::}$, $\beta_3 = \sigma_{min}^2(\mathbb{E}(bcirc(\mathscr{P}_{I_i})))$, $\beta_4 = \sigma_{min}^2(\mathbb{E}(bcirc(\mathscr{Q}_{J_j})))$.

Compared to the TREK method, the TREBK method achieves the specified accuracy in less time. However, the row or column indexes in the block used for iteration in the TREBK method are selected randomly and there is a lot of room for improvement. Among the many extensions of the Kaczmarz method in matrix form, Kaczmarz of the greedy type can select better rows for iteration, which greatly improves the calculation speed. In these extension methods, the almost-maximal residual block Kaczmarz (AMRBK) [19, 20] method is a better working method, we abbreviate this method as BEM. During each iteration, the BEM method uses the idea of maximum remains, collecting indexes of larger entries in the residual vector during each iteration and iterating over the resulting set of indexes. Inspired by this idea, we try to apply the concept of almost-maximal residual block to the first part of the TREBK method and named this method tensor randomized extended greedy block Kaczmarz (TREGBK).

In each iteration of the TREGBK method, we first collect the indices of the larger entries of the residual vector as the column block index set required for the first part iteration. Mathematically, the block index set of the TREGBK method can be characterized as

$$\tau_j = \left\{ j \mid \|(\mathscr{A}\mathscr{Z})_{:j:}\|_F^2 \ge \delta \cdot \max_{1\le t\le l}\|(\mathscr{A}\mathscr{Z})_{:t:}\|_F^2 \right\},$$

where $j \in \{1, 2, \cdots, l\}$ and $\delta \in (0, 1]$. Then select the row block index required for the second part iteration according to the probability $\|\mathscr{A}_{I_i::}\|_F^2/\|\mathscr{A}\|_F^2$. Therefore, the

iterative process of the TREGBK method is

$$\mathscr{Z}^k = \mathscr{Z}^{k-1} - \mathscr{A}_{:\tau_j:}(\mathscr{A}_{:\tau_j:})^\dagger \mathscr{Z}^{k-1},$$
$$\mathscr{X}^k = \mathscr{X}^{k-1} - (\mathscr{A}_{I_i::})^\dagger(\mathscr{A}_{I_i::}\mathscr{X}^{k-1} - \mathscr{B}_{I_i::} + \mathscr{Z}^k_{I_i::}),$$

where $\mathscr{A}_{I_i::}$,$\mathscr{A}_{:\tau_j:}$ are the subtensor of the tensor $\mathscr{A}$. Algorithm 3 lists the TREGBK method.

Algorithm 3 Tensor randomized extended greedy block Kaczmarz algorithm.

Require: $\mathscr{A} \in \mathbb{C}^{m\times l\times n}$, $\mathscr{B} \in \mathbb{C}^{m\times p\times n}$, the maximum number of iterations T, $\delta \in (0, 1]$.
Ensure: the approximation solution $\mathscr{X}^k$ of $\mathscr{A}\mathscr{X} = \mathscr{B}$.
1: Let$\{I_1, I_2, \cdots, I_s\}$ be partitions of $[m]$.
2: Initialize $\mathscr{X}^0 = 0$ and $\mathscr{Z}^0 = \mathscr{B}$.
3: **for** $k = 1, 2, \cdots, T$ **do**
4: Compute the target block τ_k of positive integers

$$\tau_j = \left\{ j \mid \|(\mathscr{A}\mathscr{Z})_{:j:}\|_F^2 \geq \delta \cdot \max_{1\leq t\leq l} \|(\mathscr{A}\mathscr{Z})_{:t:}\|_F^2 \right\}, \tag{4}$$

5: $\mathscr{Z}^k = \mathscr{Z}^{k-1} - \mathscr{A}_{:\tau_j:}(\mathscr{A}_{:\tau_j:})^\dagger \mathscr{Z}^{k-1}$,
6: Pick $i \in [s]$ with probability $\|\mathscr{A}_{I_i::}\|_F^2/\|\mathscr{A}\|_F^2$,
7: $\mathscr{X}^k = \mathscr{X}^{k-1} - (\mathscr{A}_{I_i::})^\dagger(\mathscr{A}_{I_i::}\mathscr{X}^{k-1} - \mathscr{B}_{I_i::} + \mathscr{Z}^k_{I_i::})$.
8: **end for**

The following results give the convergence of the TREGBK method.

Theorem 3 *Let $\mathscr{X}_*$ be the true tensor of the inconsisitent tensor linear system of Eq. (1) and $\mathscr{X}^k$ be the kth approximation of $\mathscr{X}_*$ given by applying TREGBK to Eq. (1), with $\delta \in (0, 1]$. The error at the kth iteration satisfies*

$$\|\mathscr{X}^k - \mathscr{X}_*\|_F^2 \leq \prod_{p=1}^{k} U_{\tau_{i_p}} \|\mathscr{X}^0 - \mathscr{X}_*\|_F^2 + \sum_{h=0}^{k-1}\Big(\prod_{p=k-h+1}^{k} U_{\tau_{i_p}}\Big)\frac{1}{\sigma^2_{min}(bcirc(\mathscr{A}_{\tau_{i_{k-h}}::}))}\prod_{q=1}^{k-h} S_{\tau_{j_q}}\|\mathscr{Z}^0\|_F^2,$$

where $\sigma_{\min}(\cdot)$ and $\sigma_{\max}(\cdot)$ denotes the smallest and maximal nonzero singular value. $N(\tau_k)$ is the cardinality of τ_k ($defining N(\tau_{-1}) = 0$). $\mathscr{P}_{\tau_{i_k}} = \mathscr{A}^\dagger_{\tau_{i_k}::}\mathscr{A}_{\tau_{i_k}::}$, $S_{\tau_{j_q}} = 1 - \frac{\delta\cdot N(\tau_{j_q})}{l}\cdot\frac{\sigma^2_{min}(bcirc(\mathscr{A}_{:\tau_{j_q}:}))}{\sigma^2_{min}(bcirc(\mathscr{A}))}$, $U_{\tau_{i_p}} = 1 - \sigma^2_{min}(bcirc(\mathscr{P}_{\tau_{i_p}}))$.

Since the proof of this theorem also follows a similar result to the existing result, we move it to the appendix.

Table 2 IT, CPU of TREK, TREBK, and TREGBK for Eq. (1) with $m = 500$, $n = 10$ and different l

Noise level	name	l	20	30	40	50	60	70	80
	TREK	IT	229.68	362.3	504.88	654.44	830.48	1019.4	1224.6
		CPU(s)	28.349	52.995	69.079	87.146	115.23	295.32	259.20
$a = 10^{-3}$	TREBK	IT	63.2	68.86	74.18	80.84	85.96	99.56	99.04
		CPU(s)	8.4605	10.113	10.786	11.657	13.006	190.56	14.34
	TREGBK	IT	2.04	7.42	12.82	16.98	23.08	28.22	33.75
		CPU(s)	0.1535	1.0166	1.852	2.5768	3.751	32.291	5.403
	$S - U_{\text{TREK}}$		184.68	52.13	37.30	33.82	63.73	9.15	47.97
	$S - U_{\text{TREBK}}$		55.10	9.95	5.82	4.52	3.47	5.91	2.65
	TREK	IT	226.92	362.74	508.26	667.98	837.74	1030.3	1249
		CPU(s)	34.840	72.445	62.583	73.134	110.02	280.09	200.80
$a = 10^{-2}$	TREBK	IT	55.22	69.64	75.58	84.26	86.62	95.92	101.10
		CPU(s)	9.2913	15.186	9.485	9.952	12.565	12.318	17.656
	TREGBK	IT	2.02	7.56	12.44	17.66	23.82	28.66	34.387
		CPU(s)	0.193	1.551	1.603	2.177	3.715	3.748	7.589
	$S - U_{\text{TREK}}$		180.14	46.71	39.03	33.60	29.62	74.73	26.46
	$S - U_{\text{TREBK}}$		48.04	9.79	5.92	4.57	3.38	3.29	2.33

4 Numerical examples

In this section, several experiments were proposed to show the effectiveness and effciency of these methods in this paper. For the first one, the coefficient tensor $\mathscr{A}$ is a dense tensor, which is randomly derived by the MATLAB function **randn**. For the second one, the coefficient tensor $\mathscr{A}$ is a sparse tensor. The third experiment performs the restoration of medical image, and the fourth is the color image "flower." Finally, the fifth is the restoration of the "Traffic video." For each experiment, the noise ϵ in the data tensor $\mathscr{B}$ is obtained by

$$\epsilon = a \cdot \frac{\epsilon_0}{\|\epsilon_0\|_F} \|\mathscr{B}_{true}\|_F, \tag{5}$$

where a is the specified noise level, $\mathscr{B}_{true}$ is the unavailable error-free data tensor that is associated with the known data tensor $\mathscr{B}$, and the tensor ϵ_0 is randomly obtained by function **randn**. The initial tensor $\mathscr{X}^0$ and $\mathscr{Z}^0$ are set as $\mathscr{X}^0 = 0$, $\mathscr{Z}^0 = \mathscr{B}$ respectively. The relative solution error (RSE) is given by

$$RSE = \frac{\|\mathscr{X}^k - \mathscr{X}_*\|_F^2}{\|\mathscr{X}_*\|_F^2},$$

and the computing time speed-up of TREK, TREBK against TREGBK is defined as

$$\textbf{S-U}_{method} = \frac{\text{The CPU time of method}}{\text{The CPU time of TREGBK}}.$$

And to reduce the experimental errors, the iteration steps (IT) and CPU time for the TREK, TREBK, and TREGBK methods are the averaged results of IT and CPU over 50 independent simulations.

Example 4.1 (dense) In this example, we use the TREK, TREBK, and TREGBK methods to solve the tensor linear system of Eq. (1), where the coefficient tensor $\mathscr{A} \in \mathbb{C}^{m\times l\times n}$ is a dense tensor. We use the MATLAB function **randn** to randomly generate the coefficient tensor $\mathscr{A} \in \mathbb{C}^{500\times l\times 10}$ and the exact solution tensor $\mathscr{X} \in \mathbb{C}^{l\times 10\times 10}$, then the tensor $\mathscr{B}$ is obtained by $\mathscr{B} = \mathscr{A}\mathscr{X}_* + \epsilon$, where the noise tensor ϵ is defined as described above. All computations are terminated once it holds that $RSE \leq 10^{-4}$. The IT and CPU required for TREK, TREBK, and TREGBK to achieve the corresponding precision at different noise levels are listed in Table 2, and the speedups $\textbf{S-U}_{\textbf{TREK}}$ and $\textbf{S-U}_{\textbf{TREBK}}$ are also reported in this table. To illustrate the iterative process of the TREK, TREBK, and TREGBK methods more clearly, we give the iteration process diagram of these methods when $m = 500$, $l = 50$, $n = 10$, $l = 10$.

Form Table 2, as for IT and CPU, we know that the TREGBK method is superior to the TREK and TREBK methods for both noise level. As can be seen from Fig. 2, all three methods achieve an accuracy smaller than 10^{-6} when the noise level is 0.001, and the TREGBK method is able to achieve this accuracy in a shorter time with fewer iteration steps compared to the TREK and TREBK methods. When the noise level is 0.01, the accuracy achieved by the three methods is 10^{-5}, and the TREGBK method

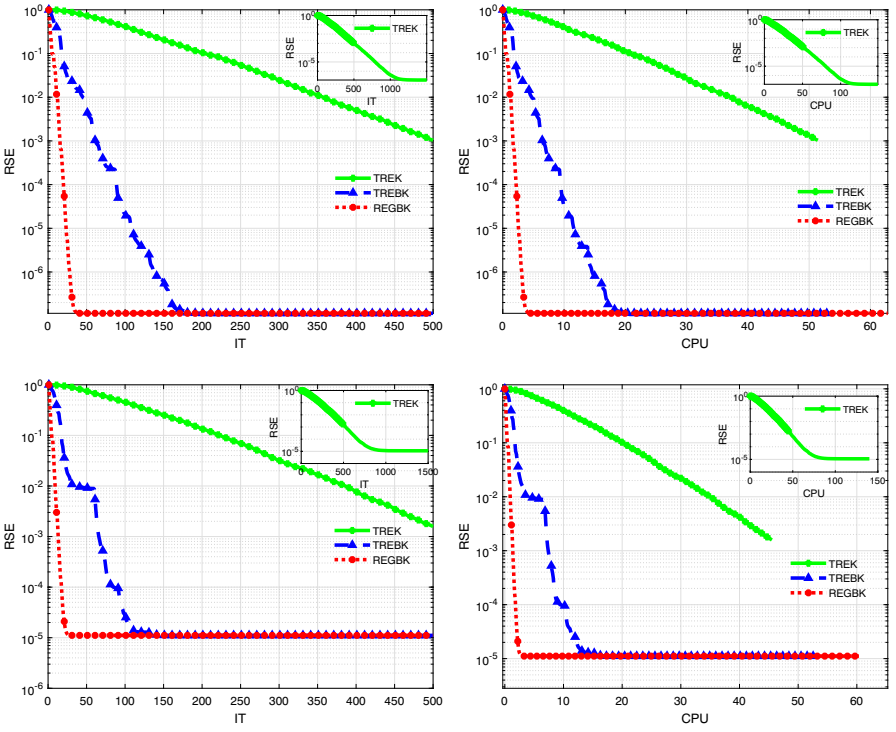

Fig. 2 Example 4.1: IT and CPU versus RSE for the TREK, TREBK, and TREGBK methods with respect to the dense tensor with the size of 500 ×50 ×10. Upper: the noise level is 0.001. Lower: the noise level is 0.01

still achieves the specified accuracy in less time and iteration steps than the TREK and TREBK methods.

Example 4.2 (sparse) In this example, we use the TREK, TREBK, and TREGBK methods to solve the tensor linear system of Eq. (1), where the coefficient tensor $\mathscr{A}$ is a sparse tensor. The sparse tensor $\mathscr{A}$ is constructed from the sparse matrix by the MATLAB function **reshape**, which is derived form the Florida sparse matrix collection [21]. The exact solution tensor $\mathscr{X}_*$ is randomly generated by the MATLAB function **randn**, the data tensor $\mathscr{B}$ is obtained by $\mathscr{B} = \mathscr{A}\mathscr{X}_* + \epsilon$, where ϵ is determined by Eq. (5), the termination criterion is exactly same as Example 4.1. But for the TREBK method, we are determined the number of blocks by the dimension of the tensor $\mathscr{A}$. For different sparse tensor systems, the IT and CPU for the TREK, TREBK, and TREGBK methods when the desired accuracy is reached are given in Table 3, and the speedups **S-U**$_{\mathbf{TREK}}$ and **S-U**$_{\mathbf{TREBK}}$ are also reported in this table. Similarly, we give an iterative process diagram of these methods to solve an example in Fig. 3.

As we can see in Table 3, for different sparse tensor systems, the TREGBK method works better than the TREK and TREBK methods. At the noise level of 0.001, the minmum and maxmum values of the speedup ratio **S-U**$_{\mathrm{TREK}}$ is 2.875 and 29.94, respectively, and the minmum and maxmum values of the speedup ratio **S-U**$_{\mathrm{TREBK}}$ is

Table 3 IT, CPU of TREK, TREBK, and TREGBK for sparse tensor systems Eq. (1)

	name	-	nos5	gre_216a	WorldCities	cities	ash85
	density	-	2.36%	1.74%	23.87%	53.04%	3.6%
	$\boldsymbol{m}$	-	468	243	650	46	85
Noise level	$\boldsymbol{l}$	-	39	24	10	11	17
	$\boldsymbol{n}$	-	12	8	5	5	5
	TREK	IT	559	333.8	427.1	770.4	675.4
		CPU(s)	77.203	8.7534	36.18	2.178	2.166
$a = 10^{-3}$	TREBK	IT	18.9	22.36	326.4	539.5	90.7
		CPU(s)	2.9664	0.6605	28.23	1.082	0.405
	TREGBK	IT	17.1	12.20	37.06	187.0	43.2
		CPU(s)	2.5785	0.3659	3.354	0.762	0.192
	$S - U_{\text{TREK}}$	-	29.94	25.31	10.79	2.857	5.356
	$S - U_{\text{TREBK}}$	-	1.15	1.81	8.415	2.363	2.102
	TREK	IT	555.2	332.8	416.96	883.6	705.2
		CPU(s)	70.629	5.7476	32.396	2.221	2.767
$a = 10^{-2}$	TREBK	IT	19.78	24.240	323.44	639.7	96.52
		CPU(s)	2.906	0.4759	25.351	1.841	0.572
	TREGBK	IT	16.16	3.240	27.06	185.2	52.24
		CPU(s)	2.299	0.0534	2.138	0.658	0.310
	$S - U_{\text{TREK}}$	-	30.72	107.6	15.156	3.37	8.93
	$S - U_{\text{TREBK}}$	-	1.26	8.91	11.868	2.80	1.847

1.15 and 8.415, respectively. As can be seen from Fig. 3, regardless of the noise level, the TREGBK method takes less time and iteration steps to achieve relative accuracy than the TREK and TREBK methods.

Example 4.3 (Medical imaging) This example restores a "MRI" image from MATLAB. The 3D "MRI" dataset is stored in a third-order tensor by using the MATLAB function **squeeze**, that is, $\mathscr{X}_* \in \mathbb{C}^{128\times128\times27}$. The deblurring tensor $\mathscr{A}$ is generated by using a modified form of the function blur with $N = 128$, $s = 1$ and band = 6, i.e.,

$$z = [exp(-([0 : band - 1].^2)/(2s^2)), zeros(1, N - band)],$$

$$A = \frac{1}{s\sqrt{2\pi}} toeplitz([z(1)\, fliplr(z(1 : end))], z),$$

$$\mathscr{A}^{(i)} = A(i, 1)A, \qquad i = 1, 2, \cdots, 128.$$

The tensor $\mathscr{B}$ is obtained by $\mathscr{B} = \mathscr{A}\mathscr{X}_* + \epsilon$, where the noise ϵ is obtained by Eq. (5). Then, the TREK, TREBK, and TREGBK methods are used to solve the resulting tensor linear equations. For the TREK, TREBK, and TREGBK methods, the calculation stops when the number of iteration steps reaches 500. We are mainly

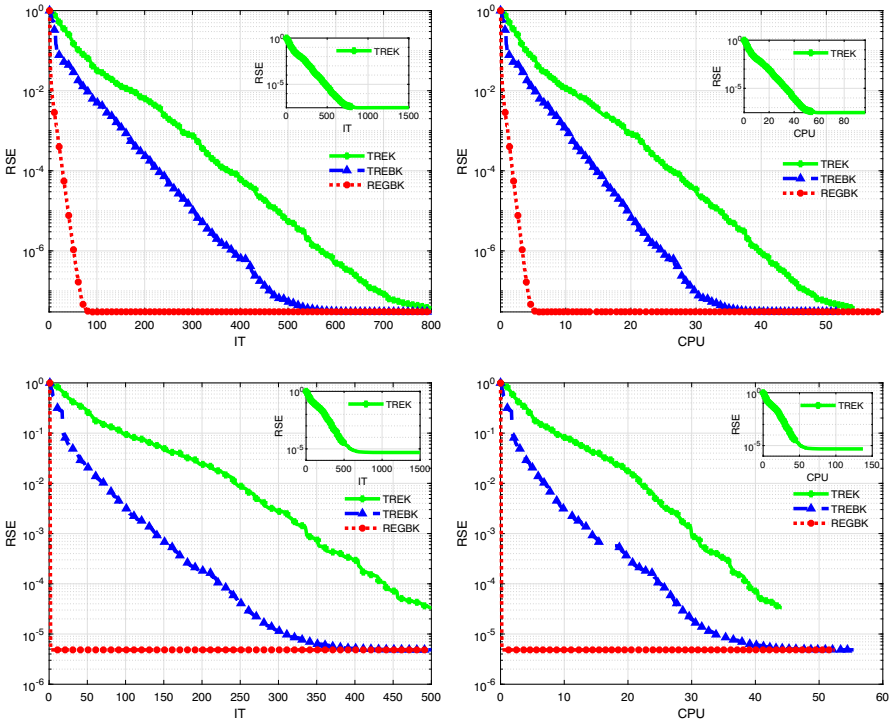

Fig. 3 Example 4.2: IT and CPU versus RSE for the TREK, TREBK, and TREGBK methods with respect to the sparse tensor with the size of 630 ×10 ×5 obtained from "WorldCities." Upper: the noise level is 0.001. Lower: the noise level is 0.01

concerned of the frames of 1, 8, 15, and 26 images of 3D dataset named "MRI," the final experimental results are given in Table 4. In addition, we give the restored images and iterative process diagram of the TREK, TREBK, and TEGBK methods when the noise level is 0.001. The original images, the blurred and noisy images, the images recovered by the TREK, TREBK, and TREGBK methods are shown in the Fig. 4. The RSE versus IT and CPU for the systems are shown in Fig. 5.

As can be seen from Table 4 and Fig. 5, the TREK, TREBK, and TREGBK methods can all recover MRI images with some accuracy. And it can be seen from Fig. 4 that

Table 4 Numerical results for the "MRI" image

Noise level	Method	RSE	CPU time	IT
$a = 10^{-3}$	TREK	1.71e−01	5.39e+01	500
	TREBK	4.97e−06	8.31e+01	500
	TREGBK	4.95e−06	9.64e+01	500
$a = 10^{-2}$	TREK	1.06e−01	5.69e+01	500
	TREBK	4.97e−04	8.31e+01	500
	TREGBK	4.93e−04	9.85e+01	500

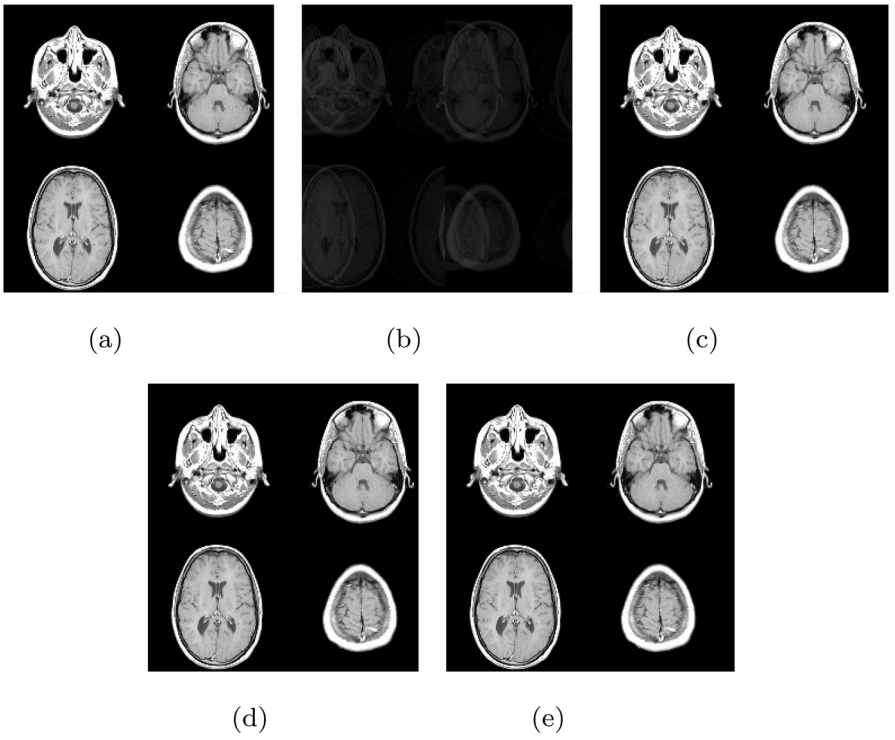

Fig. 4 **a** exact images; **b** blurred and noisy images; **c** restored images by the TREK method; **d** restored images by the TREBK method; **e** restored images by the TREGBK method

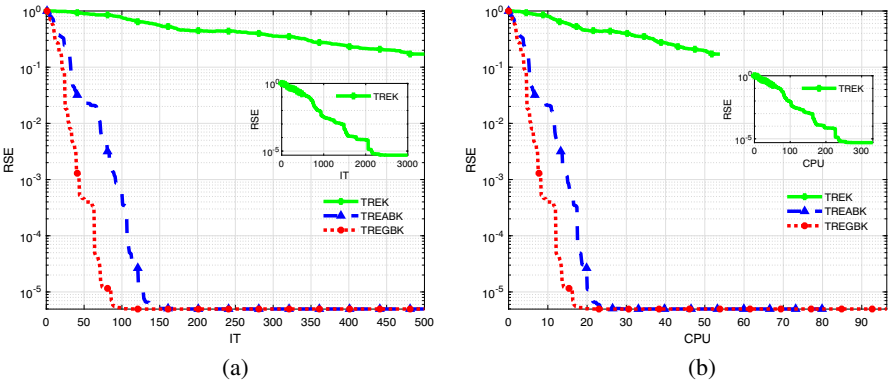

Fig. 5 Example 4.3: **a** plots of RSE versus IT for "MRI," by the TREK, TREBK, and TREGBK methods, respectively; **b** plots of RSE versus CPU for the "MRI" image, by the TREK, TREBK, and TREGBK methods, respectively

Table 5 Numerical results for the "flower" image

Noise level	Method	RSE	CPU time	IT
$a = 10^{-3}$	fTREK	3.36e−01	2.45e+02	500
	TREBK	8.54e−06	3.11e+02	500
	TREGBK	7.04e−06	3.86e+02	500
$a = 10^{-2}$	TREK	3.13e−01	2.47e+0	500
	TREBK	6.97e−04	3.13e+02	500
	TREGBK	6.98e−04	3.80e+02	500

when the noise level is 0.001, the relative error of the TREK, TREBK, and TREGBK methods stabilizes around a fixed value when the number of IT reaches a certain value. And Since the TREGBK method is an improvement on the TREK and TREBK methods, the TREGBK method can obtain stable relative error values in less time and fewer iteration steps.

Example 4.4 (Color image) This example shows the restoration of a blurred and noisy "flower" color image[1] by the TREK, TREBK, and TREGBK methods. The original image $\mathscr{X}_{ori} \in \mathbb{C}^{256\times3\times256}$ is stored as a tensor $\mathscr{X}_* \in \mathbb{C}^{200\times3\times200}$ by the MATLAB function **imresize** and **reshape**. The deblurring tensor $\mathscr{A}$ is get by

$$\mathscr{A}_{::i} = A(i,1)A, \qquad i = 1, 2, \cdots, 200,$$

where the matrix A is given by the MATLAB function

$$z = [exp(-([0 : band - 1].^2)/(2s^2)), zeros(1, N - band)],$$

$$A = \frac{1}{s\sqrt{2\pi}} toeplitz([z(1)\, fliplr(z(1 : end))], z),$$

with N=200, s=1 and band=6. The tensor $\mathscr{B}$ is obtained by $\mathscr{B} = \mathscr{A}\mathscr{X}_* + \epsilon$, where the noise ϵ is determined by Eq. (5). For the TREK, TREBK, and TREGBK methods, the maximum number of iterative steps is 500. The final experimental results are given in Table 5. In particular, when the noise level is 0.001, we give the color image recovered by the TREK, TREBK, and TREGBK methods and give iterative process diagrams of these methods. The original image, the blurred and noisy image, the images recovered by the TREK, TREBK, and TREGBK methods are shown in the Fig. 6. The RSE versus IT and CPU for this system are shown in the Fig. 7.

As can be seen from Table 5, the relative errors obtained by the TREK, TREBK, and TREGBK methods are not much different for a specific number of iteration steps. And compared with the TREK and TREBK method, the TREGBK method seems to require more time to iterate under the limitation of a certain number of iteration steps. However, as can be seen from Fig. 6, compared to the TREK and TREBK methods, the TREGBK method can obtain stable relative errors in less time and with fewer iteration steps.

[1] http://www.hlevkin.com/TestImages.

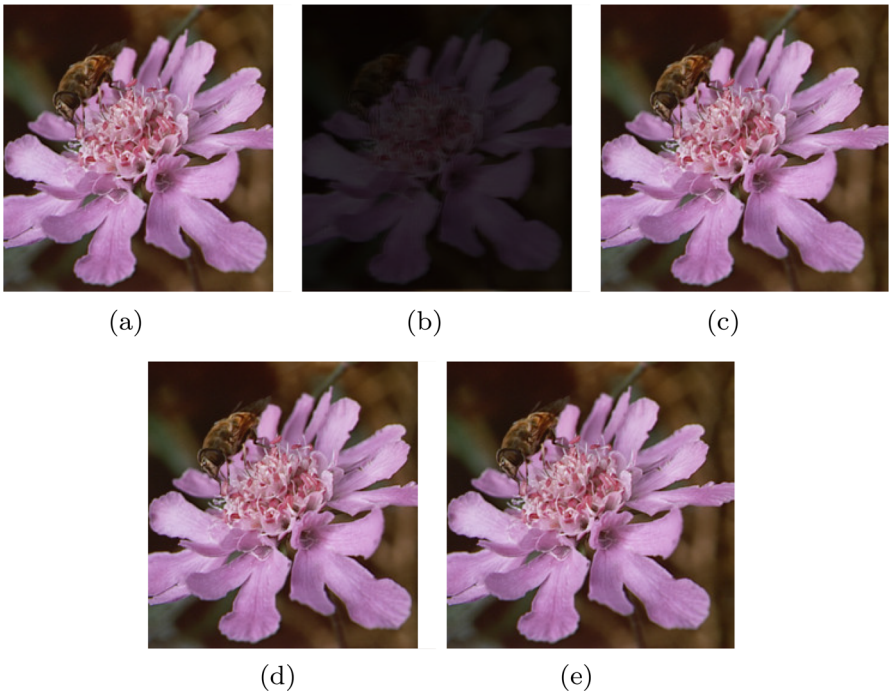

Fig. 6 **a** exact color image; **b** blurred and noisy image; **c** restored color image by the TREK method; **d** restored color image by the TREBK method; **e** restored color image by the TREGBK method

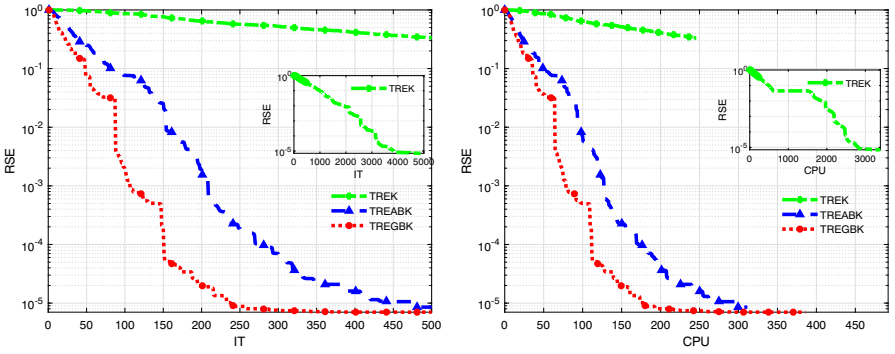

Fig. 7 Example 4.4: **a** plots of RSE versus IT for the "flower" image, by the TREK, TREBK, and TREGBK methods, respectively; **b** plots of RSE versus CPU for the "flower" image, by the TREK, TREBK, and TREGBK methods, respectively

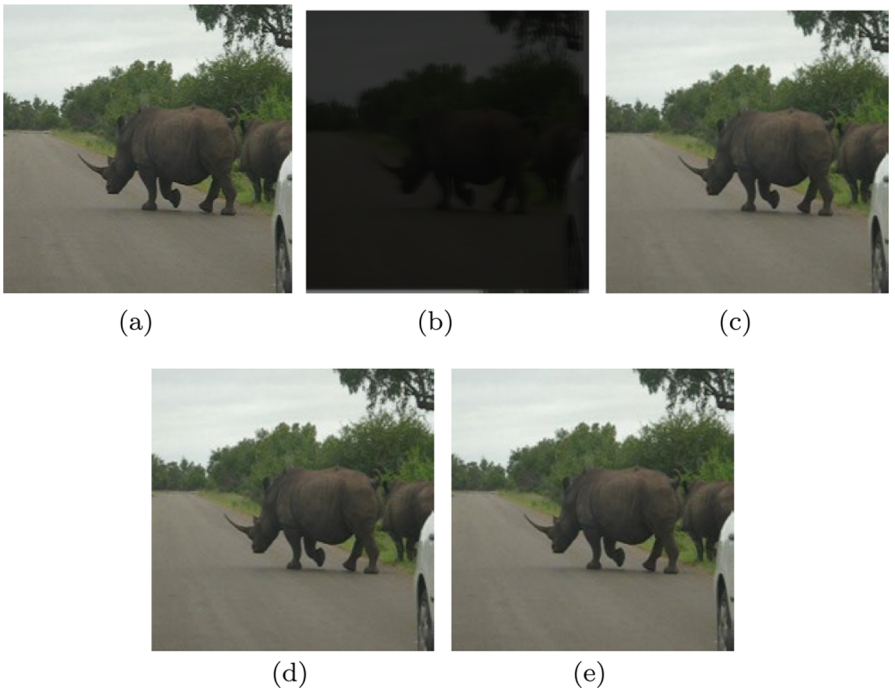

Fig. 8 **a** exact color image; **b** blurred and noisy image; **c** restored image by the TREK method; **d** restored image by the TREBK method; **e** restored image by the TREGBK method

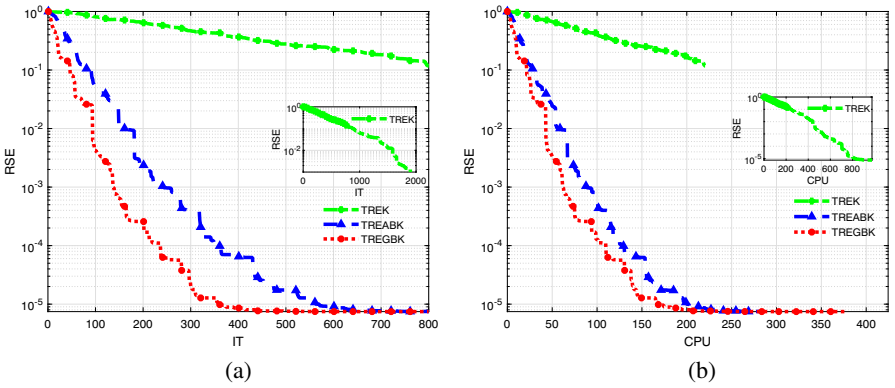

Fig. 9 Example 4.5: **a** plots of RSE versus IT for "Rhinos," by the TREK, TREBK, and TREGBK methods, respectively; **b** plots of RSE versus CPU for "Rhinos," by the TREK, TREBK, and TREGBK methods, respectively

Table 6 Numerical results for the "Rhinos" image

Noise level	Method	RSE	CPU time	IT
$a = 10^{-3}$	**TREK**	1.17e−01	2.20e+02	800
	TREBK	7.48e−06	2.82e+02	800
	TREGBK	7.45e−06	3.79e+02	800
$a = 10^{-2}$	**TREK**	1.01e−01	2.25e+02	800
	TREBK	7.47e−04	2.76e+02	800
	TREGBK	7.47e−04	3.78e+02	800

Example 4.5 (Video restoration) This example considers the restoration of the fifth frame of the Rhinos video from MATLAB. Using the **imresize** and **reshape** functions the blur- and noise-free frame can be stored as a tensor $\mathscr{X}_* \in \mathbb{C}^{240\times 3\times 240}$, it is blurred by the tensor $\mathscr{A} \in \mathbb{C}^{240\times 240\times 240}$, which is generated similarly as in Example 4.3 with $N = 240$, $s = 1$, band = 6. Then the blurred and noisy image is generated by $\mathscr{B} = \mathscr{A}\mathscr{X}_* + \epsilon$, where the noise ϵ is obtained by Eq. (5). For the TREK, TREBK, and TREGBK methods, we set their maximum number of iterations to 500. We give the restored images and iterative process diagram of the TREK, TREBK, and TEGBK methods when the noise level is 0.001 (Table 6). The original images, the blurred and noisy images, the images recovered by the TREK, TREBK, and TREGBK methods are shown in the Fig. 8. The RSE versus IT and CPU for the systems are shown in Fig. 9.

It can be seen from the experimental results that compared with the TREK and TREBK methods, the TREGBK method can achieve a certain recovery effect with a shorter time and fewer iteration steps.

5 Conclusion

This work extend the randomized extended kaczmarz tpye methods from matrix form to tensor form, and presents the TREK, TREBK, and TREGBK methods to solve large-scale inconsistent tensor linear system of equations. The convergence of each method is considered. Several numerical examples illustrate the proposed methods.

Acknowledgements The authors would like to thank the referees for their helpful and constructive comments. This research was supported in part by the Sichuan Science and Technology Program (grant 2022ZYD0008).

Author contribution G.H. proposed the idea of this paper and commented the manuscript, and S.Z. ran the examples and wrote the main manuscript text. These authors contributed equally to this work.

Funding None

Code availability Yes, it is

Declarations

Ethics approval Yes, it is

Consent to participate Not applicable

Consent for publication Yes, it is

Conflict of interest The authors declare no competing interests.

Appendix

The Proof of Theorem 1

Let $\mathscr{X}^k$ is obtained by iterating on k steps of the TREK method, then we have that

$$\begin{aligned}
\|\mathscr{X}^k - \mathscr{X}_*\|_F^2 &= \|\mathscr{X}^{k-1} - \mathscr{X}_* - \mathscr{A}_{i::}^*(\mathscr{A}_{i::}\mathscr{A}_{i::}^*)^\dagger(\mathscr{A}_{i::}\mathscr{X}^{k-1} - \mathscr{B}_{i::} + \mathscr{Z}_{i::}^k)\|_F^2 \\
&\le \|\mathscr{X}^{k-1} - \mathscr{X}_*\|_F^2 - \|\mathscr{A}_{i::}^*(\mathscr{A}_{i::}\mathscr{A}_{i::}^*)^\dagger(\mathscr{A}_{i::}\mathscr{X}^{k-1} - \mathscr{B}_{i::} + \mathscr{Z}_{i::}^k)\|_F^2 \\
&\le \|\mathscr{X}^{k-1} - \mathscr{X}_*\|_F^2 - \|\mathscr{A}_{i::}^*(\mathscr{A}_{i::}\mathscr{A}_{i::}^*)^\dagger\mathscr{A}_{i::}(\mathscr{X}^{k-1} - \mathscr{X}_*)\|_F^2 \\
&\quad + \|\mathscr{A}_{i::}^*(\mathscr{A}_{i::}\mathscr{A}_{i::}^*)^{-1}\mathscr{Z}_{i::}^k\|_F^2, \qquad (6) \\
&= \|\mathscr{X}^{k-1} - \mathscr{X}_*\|_F^2 - \|\mathscr{P}_i(\mathscr{X}^{k-1} - \mathscr{X}_*)\|_F^2 + \|\mathscr{R}_i\mathscr{Z}_{i::}^k\|_F^2 \\
&= \|\mathscr{X}^{k-1} - \mathscr{X}_*\|_F^2 - \|bcirc(\mathscr{P}_i)\cdot unfold(\mathscr{X}^{k-1} - \mathscr{X}_*)\|_F^2 \\
&\quad + \|bcirc(\mathscr{R}_i)\cdot unfold(\mathscr{Z}_{i::}^k)\|_F^2, \qquad (7)
\end{aligned}$$

where $\mathscr{P}_i = \mathscr{A}_{i::}^*(\mathscr{A}_{i::}\mathscr{A}_{i::}^*)^\dagger\mathscr{A}_{i::}$, $\mathscr{R}_i = \mathscr{A}_{i::}^*(\mathscr{A}_{i::}\mathscr{A}_{i::}^*)^\dagger$, and Eq. (6) is derived from the properties of norms:

$$\begin{aligned}
&\|\mathscr{A}_{i::}^*(\mathscr{A}_{i::}\mathscr{A}_{i::}^*)^\dagger(\mathscr{A}_{i::}\mathscr{X}^{k-1} - \mathscr{B}_{i::} + \mathscr{Z}_{i::}^k)\|_F^2 \\
&= \|\mathscr{A}_{i::}^*(\mathscr{A}_{i::}\mathscr{A}_{i::}^*)^\dagger\mathscr{A}_{i::}(\mathscr{X}^{k-1} - \mathscr{X}_*) + \mathscr{A}_{i::}^*(\mathscr{A}_{i::}\mathscr{A}_{i::}^*)^\dagger\mathscr{Z}_{i::}^k\|_F^2 \\
&\ge \|\mathscr{A}_{i::}^*(\mathscr{A}_{i::}\mathscr{A}_{i::}^*)^\dagger\mathscr{A}_{i::}(\mathscr{X}^{k-1} - \mathscr{X}_*)\|_F^2 - \|\mathscr{A}_{i::}^*(\mathscr{A}_{i::}\mathscr{A}_{i::}^*)^\dagger\mathscr{Z}_{i::}^k\|_F^2.
\end{aligned}$$

Due to the fact that

$$\sigma_{min}^2(A)\|x\|_2^2 < \|Ax\|_2^2 < \sigma_{max}^2(A)\|x\|_2^2,$$

Equation (7) can be deformed as

$$\begin{aligned}
&\|\mathscr{X}^{k-1} - \mathscr{X}_*\|_F^2 - \|bcirc(\mathscr{P}_i)\cdot unfold(\mathscr{X}^{k-1} - \mathscr{X}_*)\|_F^2 + \|bcirc(\mathscr{R}_i)\cdot unfold(\mathscr{Z}_{i::}^k)\|_F^2 \\
&\le \|\mathscr{X}^{k-1} - \mathscr{X}_*\|_F^2 - \sigma_{min}^2(bcirc(\mathscr{P}_i))\|\mathscr{X}^{k-1} - \mathscr{X}_*\|_F^2 + \sigma_{max}^2(bcirc(\mathscr{R}_i))\|\mathscr{Z}_{i::}^k\|_F^2 \\
&= (1 - \sigma_{min}^2(bcirc(\mathscr{P}_i)))\|\mathscr{X}^{k-1} - \mathscr{X}_*\|_F^2 + \sigma_{max}^2(bcirc(\mathscr{R}_i))\|\mathscr{Z}_{i::}^k\|_F^2.
\end{aligned}$$

And because

$$\begin{aligned}
\|\mathscr{Z}^k\|_F^2 &= \|\mathscr{Z}^{k-1} - \mathscr{A}_{:j:}(\mathscr{A}_{:j:}^*\mathscr{A}_{:j:})^\dagger \mathscr{A}_{:j:}^*\mathscr{Z}^{k-1}\|_F^2 \\
&\le \|\mathscr{Z}^{k-1}\|_F^2 - \|\mathscr{A}_{:j:}(\mathscr{A}_{:j:}^*\mathscr{A}_{:j:})^\dagger \mathscr{A}_{:j:}^*\mathscr{Z}^{k-1}\|_F^2 \\
&\le \|\mathscr{Z}^{k-1}\|_F^2 - \|\mathscr{Q}_j\mathscr{Z}^{k-1}\|_F^2 \\
&= \|\mathscr{Z}^{k-1}\|_F^2 - \|bcirc(\mathscr{Q}_j)\cdot unfold(\mathscr{Z}^{k-1})\|_F^2 \\
&\le \|\mathscr{Z}^{k-1}\|_F^2 - \sigma_{min}^2(bcirc(\mathscr{Q}_j))\|\mathscr{Z}^{k-1}\|_F^2 \\
&= (1-\sigma_{min}^2(bcirc(\mathscr{Q}_j)))\|\mathscr{Z}^{k-1})\|_F^2 \\
&\le (1-\sigma_{min}^2(bcirc(\mathscr{Q}_j)))^k\|\mathscr{Z}^0\|_F^2,
\end{aligned}$$

where $\mathscr{Q}_j = \mathscr{A}_{:j:}(\mathscr{A}_{:j:}^*\mathscr{A}_{:j:})^\dagger\mathscr{A}_{:j:}^*$, we have

$$\begin{aligned}
\|\mathscr{X}^k - \mathscr{X}_*\|_F^2 &\le (1-\sigma_{min}^2(bcirc(\mathscr{P}_i)))\|\mathscr{X}^{k-1} - \mathscr{X}_*\|_F^2 \\
&\quad +\sigma_{max}^2(bcirc(\mathscr{R}_i))\|\mathscr{Z}_{i::}^k\|_F^2 \\
&\le (1-\sigma_{min}^2(bcirc(\mathscr{P}_i)))\|\mathscr{X}^{k-1} - \mathscr{X}_*\|_F^2 \\
&\quad +\sigma_{max}^2(bcirc(\mathscr{R}_i))(1-\sigma_{min}^2(bcirc(\mathscr{Q}_j)))^k\|\mathscr{Z}^0\|_F^2. \qquad (8)
\end{aligned}$$

By taking the full expectation on both sides of Eq. (8), we have

$$\begin{aligned}
\mathbb{E}\big[\|\mathscr{X}^k - \mathscr{X}_*\|_F^2\big] &\le (1-\sigma_{min}^2(\mathbb{E}(bcirc(\mathscr{P}_i))))\mathbb{E}\|\mathscr{X}^{k-1} - \mathscr{X}_*\|_F^2 \\
&\quad + \sigma_{max}^2(\mathbb{E}(bcirc(\mathscr{R}_i)))(1-\sigma_{min}^2(\mathbb{E}(bcirc(\mathscr{Q}_j))))^k\|\mathscr{Z}^0\|_F^2. \qquad (9)
\end{aligned}$$

Let $\beta_1 = \sigma_{min}^2(\mathbb{E}(bcirc(\mathscr{P}_i)))$, $\beta_2 = \sigma_{min}^2(\mathbb{E}(bcirc(\mathscr{Q}_j)))$, and in order to ensure the convergence of the algorithm, β_1 and β_2 should be less than 1. Then Eq. (9) can ba written as

$$\begin{aligned}
\mathbb{E}\big[\|\mathscr{X}^k - \mathscr{X}_*\|_F^2\big] &\le (1-\beta_1)\|\mathscr{X}^{k-1} - \mathscr{X}_*\|_F^2 \\
&\quad +\sigma_{max}^2(\mathbb{E}(bcirc(\mathscr{R}_i)))(1-\beta_2)^k\|\mathscr{Z}^0\|_F^2 \\
&\le (1-\beta_1)^k\|\mathscr{X}^0 - \mathscr{X}_*\|_F^2 \qquad (10) \\
&\quad + \sum_{l=0}^{k-1}(1-\beta_1)^l\sigma_{max}^2(\mathbb{E}(bcirc(\mathscr{R}_i)))(1-\beta_2)^k\|\mathscr{Z}^0\|_F^2 \\
&\le (1-\beta_1)^k\|\mathscr{X}^0 - \mathscr{X}_*\|_F^2 \qquad (11) \\
&\quad + \sum_{l=0}^{\infty}(1-\beta_1)^l\sigma_{max}^2(\mathbb{E}(bcirc(\mathscr{R}_i)))(1-\beta_2)^k\|\mathscr{Z}^0\|_F^2. \qquad (12)
\end{aligned}$$

Because

$$\sum_{l=0}^{\infty}(1-\beta_1)^l = \frac{1}{1-(1-\beta_1)} = \frac{1}{\beta_1}, \tag{13}$$

the convergence rate of the TREK method is

$$\begin{aligned}\mathbb{E}\left[\|\mathscr{X}^k - \mathscr{X}_*\|_F^2\right] \le (1-\beta_1)^k \|\mathscr{X}^0 - \mathscr{X}_*\|_F^2 \\ + \frac{\sigma_{max}^2(\mathbb{E}(bcirc(\mathscr{R}_i)))(1-\beta_2)^k}{\beta_1}\|\mathscr{Z}^0\|_F^2.\end{aligned} \tag{14}$$

The Proof of Theorem 3

Proof In the kth iteration, let the column block selected by the maximum remaining is τ_{j_k}, the randomly selected row block is τ_{i_k}. On the one hand, we have

$$\begin{aligned}\|\mathscr{Z}^k\|_F^2 &= \|\mathscr{Z}^{k-1}\|_F^2 - \|\mathscr{A}_{:\tau_{j_k}:}\mathscr{A}_{:\tau_{j_k}:}^{-1}\mathscr{Z}^{k-1}\|_F^2 \\ &\le \|\mathscr{Z}^{k-1}\|_F^2 - \sigma_{min}^2(bcirc(\mathscr{A}_{:\tau_{j_k}:}))\|\mathscr{A}_{:\tau_{j_k}:}^{\dagger}\mathscr{Z}^{k-1}\|_F^2 \end{aligned} \tag{15}$$

$$\begin{aligned}&= \|\mathscr{Z}^{k-1}\|_F^2 - \sigma_{min}^2(bcirc(\mathscr{A}_{:\tau_{j_k}:}))\sum_{j\in\tau_{j_k}}\|\mathscr{A}_{:j:}^{\dagger}\mathscr{Z}^{k-1}\|_F^2 \\ &\le \|\mathscr{Z}^{k-1}\|_F^2 - \sigma_{min}^2(bcirc(\mathscr{A}_{:\tau_{j_k}:}))\cdot N(\tau_{j_k})\cdot\delta\cdot\max_{1\le j\le l}\|\mathscr{A}_{:j:}^{\dagger}\mathscr{Z}^{k-1}\|_F^2,\end{aligned} \tag{16}$$

where Eq. (15) is based on the following facts:

$$\sigma_{min}^2(A)\|x\|_2^2 < \|Ax\|_2^2 < \sigma_{max}^2(A)\|x\|_2^2.$$

And because

$$\begin{aligned}\|\mathscr{A}^{\dagger}\mathscr{Z}^{k-1}\|_F^2 &= \sum_{j\in l}\|\mathscr{A}_{:j:}^{\dagger}\mathscr{Z}^{k-1}\|_F^2 \\ &\le l\cdot\max_{1\le j\le l}\|\mathscr{A}_{:j:}^{\dagger}\mathscr{Z}^{k-1}\|_F^2,\end{aligned}$$

which implies that as

$$\delta\cdot\max_{1\le j\le l}\|\mathscr{A}_{:j:}^{\dagger}\mathscr{Z}^{k-1}\|_F^2 \ge \frac{\delta}{l}\|\mathscr{A}^{\dagger}\mathscr{Z}^{k-1}\|_F^2.$$

Then Eq. (16) can be written

$$\begin{aligned}\|\mathscr{Z}^k\|_F^2 &\le \|\mathscr{Z}^{k-1}\|_F^2 - \sigma_{min}^2(bcirc(\mathscr{A}_{:\tau_{j_k}:}))\frac{\delta\cdot N(\tau_{j_k})}{l}\|\mathscr{A}^\dagger\mathscr{Z}^{k-1}\|_F^2\\ &\le \|\mathscr{Z}^{k-1}\|_F^2 - \sigma_{min}^2(bcirc(\mathscr{A}_{:\tau_{j_k}:}))\frac{\delta\cdot N(\tau_{j_k})}{l}\sigma_{max}^2\\ &\qquad (bcirc(\mathscr{A}^\dagger))\|\mathscr{Z}^{k-1}\|_F^2\\ &\le \big(1-\frac{\delta\cdot N(\tau_{j_k})}{l}\cdot\frac{\sigma_{min}^2(bcirc(\mathscr{A}_{:\tau_{j_k}:}))}{\sigma_{min}^2(bcirc(\mathscr{A}))}\big)\|\mathscr{Z}^{k-1}\|_F^2. \end{aligned}\tag{17}$$

Let

$$S_{\tau_{j_q}} = 1-\frac{\delta\cdot N(\tau_{j_q})}{l}\cdot\frac{\sigma_{min}^2(bcirc(\mathscr{A}_{:\tau_{j_q}:}))}{\sigma_{min}^2(bcirc(\mathscr{A}))}.\tag{18}$$

where $q = 1, 2, \cdots, k$. Combining Eqs. (17) and (18), it can be obtained

$$\|\mathscr{Z}^k\|_F^2 \le \prod_{q=1}^{k} S_{\tau_{j_q}}\|\mathscr{Z}^0\|_F^2.\tag{19}$$

On the other hand, let $\Upsilon^k = \mathscr{X}^k - \mathscr{X}_*$, there is

$$\begin{aligned}\|\Upsilon^k\|_F^2 &= \|\mathscr{X}^{k-1} - \mathscr{X}_* - \mathscr{A}_{\tau_{i_k}::}^\dagger(\mathscr{A}_{\tau_{i_k}::}\mathscr{X}^{k-1} - \mathscr{B}_{\tau_{i_k}::} + \mathscr{Z}_{\tau_{i_k}::}^k)\\ &\le \|\Upsilon^{k-1}\|_F^2 - \|\mathscr{A}_{\tau_{i_k}::}^\dagger(\mathscr{A}_{\tau_{i_k}::}\mathscr{X}^{k-1} - \mathscr{B}_{\tau_{i_k}::} + \mathscr{Z}_{\tau_{i_k}::}^k)\|_F^2\\ &= \|\Upsilon^{k-1}\|_F^2 - \|\mathscr{A}_{\tau_{i_k}::}^\dagger\mathscr{A}_{\tau_{i_k}::}\Upsilon^{k-1} + \mathscr{A}_{\tau_{i_k}::}^\dagger\mathscr{Z}_{\tau_{i_k}::}^k\|_F^2. \end{aligned}\tag{20}$$

Now order $\mathscr{P}_{\tau_{i_k}} = \mathscr{A}_{\tau_{i_k}::}^\dagger\mathscr{A}_{\tau_{i_k}::}$, we use the norm properties to get

$$\begin{aligned}\|\Upsilon^k\|_F^2 &= \|\Upsilon^{k-1}\|_F^2 - \|\mathscr{P}_{\tau_{i_k}}\Upsilon^{k-1} + \mathscr{A}_{\tau_{i_k}::}^\dagger\mathscr{Z}_{\tau_{i_k}::}^k\|_F^2\\ &\le \|\Upsilon^{k-1}\|_F^2 - \|\mathscr{P}_{\tau_{i_k}}\Upsilon^{k-1}\|_F^2 + \|\mathscr{A}_{\tau_{i_k}::}^\dagger\mathscr{Z}_{\tau_{i_k}::}^k\|_F^2, \end{aligned}$$

then we have

$$\begin{aligned}\|\Upsilon^k\|_F^2 \le\ & \|\Upsilon^{k-1}\|_F^2 - \|\mathscr{P}_{\tau_{i_k}}\Upsilon^{k-1}\|_F^2 + \|\mathscr{A}_{\tau_{i_k}::}^\dagger\mathscr{Z}_{\tau_{i_k}::}^k\|_F^2\\ \le\ & \|\Upsilon^{k-1}\|_F^2 - \sigma_{min}^2(bcirc(\mathscr{P}_{\tau_{i_k}}))\|\Upsilon^{k-1}\|_F^2\\ & +\sigma_{max}^2(bcirc(\mathscr{A}_{\tau_{i_k}::}^\dagger))\|\mathscr{Z}_{\tau_{i_k}::}^k\|_F^2\\ \le\ & (1-\sigma_{min}^2(bcirc(\mathscr{P}_{\tau_{i_k}})))\|\Upsilon^{k-1}\|_F^2\\ & +\frac{1}{\sigma_{min}^2(bcirc(\mathscr{A}_{\tau_{i_k}::}))}\prod_{q=1}^{k} S_{\tau_{j_q}}\|\mathscr{Z}^0\|_F^2, \end{aligned}\tag{21}$$

where Eq. (21) is obtained by Eq. (18). Now let

$$U_{\tau_{i_p}} = 1 - \sigma_{min}^2(bcirc(\mathscr{P}_{\tau_{i_p}})),$$

where $p = 0, 1, 2, \cdots k$, $U_{\tau_{i_0}} = 0$. So the convergence of the TREGBK method is

$$\begin{aligned}\|\mathscr{X}^k - \mathscr{X}_*\|_F^2 \le & \prod_{p=1}^{k} U_{\tau_{i_p}} \|\mathscr{X}^0 - \mathscr{X}_*\|_F^2 \\ & + \sum_{h=0}^{k-1}\Big(\prod_{p=k-h+1}^{k} U_{\tau_{i_p}}\Big)\frac{1}{\sigma_{min}^2(bcirc(\mathscr{A}_{\tau_{i_{k-h}}::}))}\prod_{q=1}^{k-h} S_{\tau_{j_q}} \|\mathscr{Z}^0\|_F^2.\end{aligned}$$

References

1. Kilmer, M.E., Martin, C.D.: Factorization strategies for third-order tensors. Linear Algebra Appl. **435**(3), 641–658 (2011)
2. Kilmer, M.E., Braman, K., Hao, N., Hoover, R.C.: Third-order tensors as operators on matrices: a theoretical and computational framework with applications in imaging. SIAM J. Matrix Anal. A. **34**(1), 148–172 (2013)
3. Reichel, L., Ugwu, U.O.: The tensor Golub-Kahan-Tikhonov method applied to the solution of ill-posed problem with a t-product structure. Numer. Linear. Algebr. **29**(1), 1–34 (2022)
4. Hao, N., Kilmer, M.E., Braman, K., Hoover, R.C.: Facial recognition using tensor-tensor decompositions. SIAM J. Imaging Sci. **6**(1), 437–463 (2013)
5. Wang, X.Z., Che, M.L., Wei, Y.M.: Tensor neural network models for tensor singular value decompositions. Comput. Optim. Appl. **75**, 753–777 (2020)
6. Soltani, S., Kilmer, M.E., Hansen, P.C.: A tensor-based dictionary learning approach to tomographic image reconstruction. BIT **56**(4), 1425–1454 (2016)
7. Needell, D., Tropp, J.A.: Paved with good intentions: analysis of a randomized block Kaczmarz method. Linear Algebra Appl. **441**, 199–221 (2012)
8. Needell, D., Zhao, R., Zouzias, A.: Randomized block Kaczmarz method with projection for solving least squares. Linear Algebra Appl. **484**, 322–343 (2015)
9. Elfving, T.: Block-iterative methods for consistent and inconsistent linear equations. Numer. Math. **35**(1), 1–12 (1980)
10. Li, R., Liu, H.: On global randomized block Kaczmarz method for image reconstruction. Electron. Res. Arch. **30**(4), 1440–1453 (2022)
11. Bai, Z.Z., Wu, W.T.: On greedy randomized Kaczmarz method for solving large sparse linear systems. SIAM J. Sci. Comput. **40**(1), A592–A606 (2018)
12. Bai, Z.Z., Wu, W.T.: On relaxed greedy randomized Kaczmarz methods for solving large sparse linear systems. Appl Math Lett. **83**, 21–26 (2018)
13. Zhang, J.H., Guo, J.H.: On relaxed greedy randomized coordinate descent methods for solving large linear least-squares problems. Appl. Numer. Math. **157**, 372–384 (2020)
14. Bai, Z.Z., Wu, W.T.: On greedy randomized coordinate descent methods for solving large linear least-squares problems. Numer. Linear Algebra Appl. **26**, e2237 (2019)
15. Ma, A., Molitor, D.: Randomized Kaczmarz for tensor linear systems. BIT **62**(1), 171–194 (2022)
16. Zouzias, A., Freris, N.M.: Randomized extended Kaczmarz for solving least squares. SIAM J. Matrix Anal A. **34**(2), 773–793 (2013)
17. Bai, Z.Z., Wu, W.T.: On partially randomized extended Kaczmarz method for solving large sparse overdetermined inconsistent linear systems. Linear Algebra Appl. **578**, 225–250 (2019)
18. Du, K., Si, W., Sun, H.: Randomized extended average block Kaczmarz for solving least squares. SIAM j. Sci. Comput. **42**(6), 3541–3559 (2020)
19. Liu, Y., Gu, C.Q.: On greedy randomized block Kaczmarz method for consistent linear systems. Linear Algebra Appl. **616**, 178–200 (2021)

20. Liu, Y., Jiang, X.L., Gu, C.Q.: On maximum residual block and two-step Gauss-Seidel algorithms for linear least-squares problems. Calcolo. **58**, 1–32 (2021)
21. Davis, T.A., Hu, Y.: The university of Florida sparse matrix collection. ACM Trans. Math. Softw. **38**(1), 1–25 (2011)