Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

A Tensor Greedy Double-Block Extended Kaczmarz Method for Inconsistent Tensor Linear Systems under the t-product

Jérémie Mabiala

jeremy@aimsammi.org

African Institute for Mathematical Sciences Senegal

Lionel Tondji

African Institute for Mathematical Sciences Senegal

Research Article

Keywords: t-product, tensor linear systems, Kaczmarz method, inconsistent systems

Posted Date: September 7th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-10807226/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

A Tensor Greedy Double-Block Extended Kaczmarz Method for Inconsistent Tensor Linear Systems under the t-product

Jérémie Mabiala[1*] and Lionel Tondji[1,2]

[1*]African Institute of Mathematical Sciences and African Master's of Machine Intelligence, Mbour, Senegal.
[2]Institute for Analysis and Algebra, TU Braunschweig, Braunschweig, Germany.

*Corresponding author(s). E-mail(s): jeremy@aimsammi.org;
Contributing authors: tngoupeyou@aimsammi.org;

Abstract

The randomized extended Kaczmarz method is an effective iterative framework for solving large-scale inconsistent linear systems. In this paper, we extend this framework to third-order inconsistent tensor linear systems under the t-product and propose the Tensor Greedy Double Block Extended Kaczmarz (TGDBEK) method. At each iteration, TGDBEK dynamically constructs active blocks of row and column slices via a residual-based greedy selection strategy, prioritizing the slices associated with the largest residual norms. Unlike existing tensor block Kaczmarz variants that rely on static, predefined partitions — the tensor randomized extended block Kaczmarz (TREBK) method, its greedy counterpart (TREGBK), and the tensor randomized extended average block Kaczmarz (TREABK) method — TGDBEK adapts the active block sizes dynamically at each step using a single intuitive threshold parameter $\boldsymbol{\eta}$. We establish the theoretical linear convergence of TGDBEK to the unique minimum-norm least-squares solution $\boldsymbol{\mathcal{A}}^{\dagger} * \boldsymbol{\mathcal{B}}$. Extensive numerical benchmarks on synthetic dense and sparse tensor systems, as well as multidimensional multichannel color and 3D volumetric MRI image deblurring problems, demonstrate that TGDBEK substantially outperforms state-of-the-art tensor Kaczmarz solvers in both iteration count and CPU running time.

Keywords: t-product, tensor linear systems, Kaczmarz method, inconsistent systems.

Mathematics Subject Classification (2010): 65F10 , 65F20 , 65F22 , 15A69

1 Introduction

In this work, we focus on solving the tensor inconsistent linear system of the form

$$\mathcal{A} * \mathcal{X} = \mathcal{B}, \ \mathcal{B} = \overline{\mathcal{B}} + \varepsilon \tag{1}$$

where $\mathcal{A} \in \mathbb{R}^{N_1 \times N_2 \times N_3}$, $\mathcal{B} \in \mathbb{R}^{N_1 \times K \times N_3}$, $\mathcal{X} \in \mathbb{R}^{N_2 \times K \times N_3}$, are third-order tensors, $\varepsilon \in \mathbb{R}^{N_1 \times K \times N_3}$ the error tensor, and $*$ the t-product proposed by Kilmer and collaborators in [1, 2]. The t-product provides a linear algebraic paradigm for third-order tensors, which includes tensor analogues of matrix concepts like inverse of a tensor, transpose, orthogonal projections, and pseudo-inverse, etc. [1, 2]. Since its introduction in [1], the t-product has been applied to several theoretical and practical applications. For instance, the authors in [6] proposed a tensor neural network (t-NN) framework, a novel paradigm for designing deep neural networks. We kindly refer to [3–6, 31] and to references therein for applications of the t-product. The problem of the form (1) arises in many practical applications, such as in color image recovery, MRI image deblurring with multiple channels, where the tensor $\mathcal{X}$ is the unknown clean tensor image to be found, $\mathcal{A}$ is a tensor blur operator, and $\mathcal{B}$ is the observed blurred and noisy data. When the noise is relatively higher, it becomes difficult to solve this problem through direct methods.

In the matrix setting, where $\mathcal{A}$ is a matrix, $\mathcal{X}$ and $\mathcal{B}$ are vectors, iterative methods like the randomized Kaczmarz (RK)[37], a randomized variant of the classical Kaczmarz method [13], can be used to solve the linear system (1). The RK method selects rows at each iteration with probability proportional to their Euclidean norm square, unlike the traditional Kaczmarz method, where rows are selected cyclically and can result in a non-linear convergence. In their seminal work, Strhomer et al. [37] proved that this row selection strategy makes the RK converge linearly in expectation for consistent and overdetermined linear systems. For inconsistent systems, however, RK only converges to within a fixed radius of the least-squares solution [34]. To solve inconsistent linear systems, Zouzias and Freris [38] extended the randomized Kaczmarz by introducing an additional iterative dual variable and proposed the randomized extended Kaczmarz (REK) which converges linearly in expectation to the minimum-norm least-squares solution $\boldsymbol{A}^{\dagger}\boldsymbol{b}$, where $\boldsymbol{A}^{\dagger}$ is the Moore-Penrose pseudoinverse of the system matrix $\boldsymbol{A}$. Explicitly, the REK method runs the RK iteration twice at each step: once on the consistent system $\boldsymbol{A}^{\top}\boldsymbol{z} = \boldsymbol{0}$ to update $\boldsymbol{z}$, and once on $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} - \boldsymbol{z}$ to update $\boldsymbol{x}$, writing $\boldsymbol{z}_0 = \boldsymbol{b}$,

$$\boldsymbol{z}^{(k+1)} = \boldsymbol{z}^{(k)} - \frac{\left(\boldsymbol{A}_{:j_k}\right)^{\top}\boldsymbol{z}^{(k)}}{\|\boldsymbol{A}_{:j_k}\|_2^2}\,\boldsymbol{A}_{:j_k}, \tag{2}$$

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} + \frac{\boldsymbol{b}_{i_k} - \boldsymbol{z}_{i_k}^{(k+1)} - \boldsymbol{A}_{i_k:}\boldsymbol{x}^{(k)}}{\|\boldsymbol{A}_{i_k:}\|_2^2}\left(\boldsymbol{A}_{i_k:}\right)^{\top}, \tag{3}$$

where $\boldsymbol{A}_{i_k:}$ and $\boldsymbol{A}_{:j_k}$ denote the i_k-th row and j_k-th column of $\boldsymbol{A}$, $\boldsymbol{b}_{i_k}$ and $\boldsymbol{z}_{i_k}$ the corresponding entries of $\boldsymbol{b}$ and of $\boldsymbol{z}$, and the pair (i_k, j_k) is drawn independently

with probability proportional to $\|\boldsymbol{A}_{i_k:}\|_2^2/\|\boldsymbol{A}\|_F^2$ and $\|\boldsymbol{A}_{:j_k}\|_2^2/\|\boldsymbol{A}\|_F^2$, respectively. As $k \to \infty$, $\boldsymbol{z}^{(k)} \to \boldsymbol{b}_{\mathcal{R}(\boldsymbol{A})^\perp}$ and $\boldsymbol{x}^{(k)} \to \boldsymbol{A}^\dagger \boldsymbol{b}$, each row and column of $\boldsymbol{A}$ being used once at each iteration to update $\boldsymbol{x}$ and $\boldsymbol{z}$, respectively.

The randomized (extended) Kaczmarz method can be slow. Two practical ideas emerged to accelerate it. The first samples blocks of rows instead of a single row, and result in the block variant methods, which includes the averaged-block variant methods[11, 12, 33, 35, 36, 40]. Averaging the block updates, as in [11, 12], keeps the iteration pseudoinverse-free and amenable to parallel implementation. In line with [12], several other accelerations have been developed for the Bregman version of the method, named Bregman-Kaczmarz: [25] transfers accelerated randomized coordinate descent from the dual to the primal space, and obtained an accelerated randomized Bregman–Kaczmarz method for linearly constrained problems; [23] adds a restart strategy and a block variant, whose linear rate in the primal follows from a Polyak–Łojasiewicz property of the dual objective; and [26] proves linear convergence in expectation for a residual-weighted sampling variant of the sparse randomized Kaczmarz method, a selection rule very close to the greedy criterion used below. A unified account of these accelerations and of their behaviour on inconsistent systems is given in the thesis [24]. The second selects rows or blocks according to a greedy criterion where only those with largest residuals are chosen, resulting in greedy randomized Kaczmarz variants studied in [7–9] and in references therein.

Alternatives approaches have been proposed to solve inconsistent linear systems, for instance through Bregman projections [21]. Schöpfer et al. [20] merged the extended and the sparse randomized Kaczmarz iterations and proved linear convergence to a *sparse* least-squares solution, in a generalized form that also tolerates corruption concentrated in a few entries of the right-hand side. Tondji et al. [21] assumed instead an independent noisy sample is drawn at each step and showed that an adaptive stepsize, estimated from the data alone, reaches the *exact* solution rather than a neighborhood of it; averaging the blocks and weighting them by their noise level accelerates this scheme further [22]. Their mechanism for dealing with inconsistency differs from the REK method we build upon herein in this respect: they replace the Euclidean projection by a Bregman one — and, sometimes in the block versions, the exact projection is replaced by an averaged-sum of projections onto each hyperplane composing the selected block — and assume about the noise is redrawn each step, whereas the REK method and the method proposed below project out the part of the right-hand side lying outside the range of the operator. It is also stated for matrices only, and t-product counterparts are still missing.

The block and greedy ideas we discussed previously have been recently brought to the t-product setting. Initially, Ma and Molitor [14] proposed in their seminal work the tensor randomized Kaczmarz (TRK) method, which extends the RK method to solve tensor linear systems under the t-product. It was shown that the TRK method is equivalent to the block randomized Kaczmarz method in the Fourier domain and

hence superior compared to the RK method applied to vectorized tensor systems. Subsequently, Chen and Qin [15] proposed the randomized regularized Kaczmarz method for tensor recovery. For applications of the randomized Kaczmarz method in tensor recovery and completion problems, please refer to [15–19, 27].The authors of [28] introduced the tensor randomized block Kaczmarz (TRBK) method. The authors of [28] introduced the tensor randomized block Kaczmarz (TRBK) method. To overcome the constraint of equation consistency, Huang and Zhong [30] extended REK directly to the t-product setting, resulting to the tensor randomized extended Kaczmarz (TREK) method. In fact, similar to the matrix randomized extended Kaczmarz, the tensor version TREK sets $\mathcal{Z}^{(0)} = \mathcal{B}$, and at each iteration k, draws the lateral and horizontal slice indexes j and i independently with probability $\|\mathcal{A}_{:,j,:}\|_F^2/\|\mathcal{A}\|_F^2$ and $\|\mathcal{A}_{i,:,:}\|_F^2/\|\mathcal{A}\|_F^2$ respectively, then updates the iterates using:

$$\mathcal{Z}^{(k+1)} = \mathcal{Z}^{(k)} - \mathcal{A}_{:,j,:} * \left((\mathcal{A}_{:,j,:})^\top * \mathcal{A}_{:,j,:}\right)^\dagger * (\mathcal{A}_{:,j,:})^\top * \mathcal{Z}^{(k)}, \tag{4}$$

$$\mathcal{X}^{(k+1)} = \mathcal{X}^{(k)} - (\mathcal{A}_{i,:,:})^\top * \left(\mathcal{A}_{i,:,:} * (\mathcal{A}_{i,:,:})^\top\right)^\dagger * \left(\mathcal{A}_{i,:,:} * \mathcal{X}^{(k)} - \mathcal{B}_{i,:,:} + \mathcal{Z}_{i,:,:}^{(k)}\right). \tag{5}$$

Furthermore, in the same paper, Huang and Zhong [30] also introduced two block variants of TREK, namely the tensor randomized extended block Kaczmarz (TREBK) and its greedy counterpart (TREGBK), which pre-partition the lateral and horizontal slices into fixed blocks rather than sampling a single slice at each step. Recently, An and collaborators [29] proposed tensor randomized extended average block Kaczmarz (TREABK) a pseudoinverse-free randomized extended average block Kaczmarz which inherently uses the averaging idea appeared before in [11, 33]. Castillo and collaborators [41] further extend Kaczmarz- and Gauss–Seidel-type iterations to tensor regression under the t-product, with an application to image deblurring, and provide a comprehensive litterature review on Kaczmarz methods in the t-product setting.

Many of these tensor block methods [28, 29] we discussed previously rely primarily on static, predefined partitions of rows or columns of the tensor system, which of course can depend on the structure of the tensor system: sparse, dense, etc. In this paper, inspired by [10], we apply the greedy randomized block extended Kaczmarz method's idea from matrices to tensor equations. Consequently, we propose a tensor greedy double block extended Kaczmarz (TGDBEK) method. The TGDBEK method constructs iteratively lateral(columns) and horizontal(rows) slices at each iteration, and does not require a fixed partition of the tensor system. In fact, the TGDBEK method focuses projection computation on where the current residual is the largest. We prove the linear convergence of the TGDBEK and demonstrate its practical effectiveness against TREK [14], TREBK [30], TREGBK [30], including applications to image deblurring and sparse tensor systems from SuiteSparse matrices [43].

The remainder of the paper is organized as follows. In Section 2, we introduce the notations and definitions. In Section 3, we describe the TGDBEK algorithm and prove its linear convergence. Numerical examples are provided in Section 4 and finally we draw conclusions in Section 5.

2 Notations and preliminaries

2.1 Basic notation

Throughout this paper, the calligraphic letters such as $\mathcal{A}, \mathcal{B}$ denote tensors. Bold capital letters represent matrices, for example, $\boldsymbol{A}, \boldsymbol{B}$. For an integer $m \geq 1$, we denote $[m] := \{1, \ldots, m\}$. If $\mathcal{A}$ is a third-order tensor, $\mathcal{A}_{ijk}$ denotes the (i, j, k)-th element. $\mathcal{A}_{i,:,:}$, $\mathcal{A}_{:,j,:}$, and $\mathcal{A}_{:,:,k}$ denote the horizontal, lateral, and frontal slices of $\mathcal{A}$, respectively. The tensor $\mathcal{A}$ and its three slices are illustrated in Fig. 1. For $J \subseteq [N_1]$ we denote $\mathcal{A}_{J,:,:} \in \mathbb{R}^{|J| \times N_2 \times N_3}$ a sub-tensor of $\mathcal{A}$ whose horizontal slice indexes belong to J. Similarly, for $U \subseteq [N_2]$ we denote $\mathcal{A}_{:,U,:} \in \mathbb{R}^{N_1 \times |U| \times N_3}$ the sub-tensor $\mathcal{A}$ whose lateral slice indexes belong to U. The pseudo-inverse of $\mathcal{A}$ is denoted $\mathcal{A}^\dagger$ and the minimum and maximum nonzero singular values of a matrix are denoted $\sigma^+_{\min}(\cdot)$ and $\sigma_{\max}(\cdot)$, respectively. Similarly, we denote $\mathrm{R}(\cdot)$ and $\ker(\cdot)$, respectively, the range and null spaces of a tensor $\mathcal{A}$ or a matrix $\boldsymbol{A}$.

We now define the intrinsic geometric parameters that the main convergence theorem will rely on.

Definition of Intrinsic Geometric Parameters.

Let $\Omega_r = \{J \subseteq [N_1] : J \neq \emptyset\}$ be the collection of all non-empty row index subsets of $\mathcal{A}$. $J \in \Omega_r$ is an admissible row block. We define the **intrinsic geometric parameters** of the tensor $\mathcal{A} \in \mathbb{R}^{N_1 \times N_2 \times N_3}$ as follows:

1. **Tensor Block Condition Number:**

$$\kappa_{\text{block}}(\mathcal{A}) := \min_{J \in \Omega_r} \frac{\left(\sigma^+_{\min}(\text{bcirc}(\mathcal{A}_{J,:,:}))\right)^2}{\sigma^2_{\max}(\text{bcirc}(\mathcal{A}_{J,:,:}))} \in (0, 1]$$

(measures the worst-case conditioning across all admissible row blocks)

2. **Smallest Active Block Singular Value:**

$$\sigma^+_{\min,\text{block}}(\mathcal{A}) := \min_{J \in \Omega_r} \sigma^+_{\min}(\text{bcirc}(\mathcal{A}_{J,:,:})) > 0$$

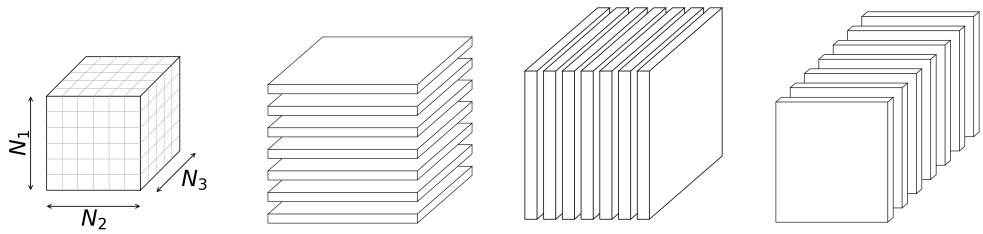
3. **Subspace Error Fraction:**

For any iteration $k \geq 0$, let θ_k denote the fraction of error captured by the active row range $R(\mathcal{A}^\top_{J_k,:,:})$:

$$\theta_k := \frac{\left\| \left(\mathcal{X}^{(k)} - \mathcal{X}_*\right)_{R\left((\mathcal{A}_{J_k,:,:})^\top\right)} \right\|_F^2}{\left\|\mathcal{X}^{(k)} - \mathcal{X}_*\right\|_F^2} \in (0, 1), \qquad \theta := \min_{k \geq 0} \theta_k > 0$$

2.2 Tensor preliminaries

We follow Kilmer and Martin [1, 2] to provide the background for the t-product linear algebra.



(a) Tensor $\mathcal{A}$ (b) Horizontal slices $\mathcal{A}_{i,:,:}$ (c) Lateral slices $\mathcal{A}_{:,j,:}$ (d) Frontal slices $\mathcal{A}_{:,:,k}$

Fig. 1 A third-order tensor $\mathcal{A} \in \mathbb{R}^{N_1 \times N_2 \times N_3}$ and its three slices : horizontal (row), lateral (column), and frontal.

Given a third-order tensor $\mathcal{A} \in \mathbb{R}^{N_1 \times N_2 \times N_3}$, we can unfold $\mathcal{A}$ into a matrix, and inversely fold the result back. The operation *unfold* and its inverse *fold* are given as

$$\text{unfold}(\mathcal{A}) := \begin{bmatrix} \mathcal{A}_{:,:,1} \\ \mathcal{A}_{:,:,2} \\ \vdots \\ \mathcal{A}_{:,:,N_3} \end{bmatrix} \in \mathbb{R}^{N_1 N_3 \times N_2}, \qquad \text{fold}\left(\begin{bmatrix} \mathcal{A}_{:,:,1} \\ \mathcal{A}_{:,:,2} \\ \vdots \\ \mathcal{A}_{:,:,N_3} \end{bmatrix} \right) := \mathcal{A}.$$

The block circulant matrix of $\mathcal{A}$ is given as

$$\text{bcirc}(\mathcal{A}) := \begin{bmatrix} \mathcal{A}_{:,:,1} & \mathcal{A}_{:,:,N_3} & \cdots & \mathcal{A}_{:,:,2} \\ \mathcal{A}_{:,:,2} & \mathcal{A}_{:,:,1} & \cdots & \mathcal{A}_{:,:,3} \\ \vdots & \vdots & \ddots & \vdots \\ \mathcal{A}_{:,:,N_3} & \mathcal{A}_{:,:,N_3-1} & \cdots & \mathcal{A}_{:,:,1} \end{bmatrix} \in \mathbb{R}^{N_1 N_3 \times N_2 N_3}.$$

Definition 2.1 (t-product [1, 2]) For $\mathcal{A} \in \mathbb{R}^{N_1 \times N_2 \times N_3}$ and $\mathcal{B} \in \mathbb{R}^{N_2 \times K \times N_3}$, the t-product $\mathcal{A} * \mathcal{B}$ is the $N_1 \times K \times N_3$ tensor defined by

$$\mathcal{A} * \mathcal{B} := \text{fold}(\text{bcirc}(\mathcal{A})\text{unfold}(\mathcal{B})).$$

Definition 2.2 ([1]) The identity tensor $\mathcal{I} \in \mathbb{R}^{N \times N \times N_3}$ is the tensor whose first frontal slice is the $N \times N$ identity matrix, and whose other frontal slices are all zeros. The identity tensor satisfies $\mathcal{A} * \mathcal{I} = \mathcal{I} * \mathcal{A} = \mathcal{A}$ for all tensors $\mathcal{A}$ of compatible dimensions.

Definition 2.3 ([1, 2]) The transpose of $\mathcal{A} \in \mathbb{R}^{N_1 \times N_2 \times N_3}$, denoted $\mathcal{A}^\top$, is the $N_2 \times N_1 \times N_3$ tensor obtained by transposing each frontal slice and then reversing the order of transposed frontal slices 2 through N_3.

Definition 2.4 ([1]) A tensor $\mathcal{A} \in \mathbb{R}^{N_1 \times N_2 \times N_3}$ defines a linear map $\mathcal{X} \mapsto \mathcal{A} * \mathcal{X}$ from $\mathbb{R}^{N_2 \times K \times N_3}$ to $\mathbb{R}^{N_1 \times K \times N_3}$. Its range and null spaces are defined as

$$\mathrm{R}(\mathcal{A}) := \left\{ \mathcal{A} * \mathcal{Y} \ : \ \mathcal{Y} \in \mathbb{R}^{N_2 \times K \times N_3} \right\}, \ \mathrm{null}(\mathcal{A}) := \left\{ \mathcal{X} \in \mathbb{R}^{N_2 \times K \times N_3} \ : \ \mathcal{A} * \mathcal{X} = \mathcal{O} \right\}.$$

The following properties will be useful in the sequel.

Fact 1. [14] For $\mathcal{A} \in \mathbb{R}^{N_1 \times N_2 \times N_3}$ and $\mathcal{B} \in \mathbb{R}^{N_2 \times K \times N_3}$, the block circulant operator satisfies:

$$\mathrm{bcirc}(\mathcal{A}^\top) = \mathrm{bcirc}(\mathcal{A})^\top, \ \mathrm{bcirc}(\mathcal{A} * \mathcal{B}) = \mathrm{bcirc}(\mathcal{A})\,\mathrm{bcirc}(\mathcal{B}).$$

As an immediate consequence: $(\mathcal{A} * \mathcal{B})^\top = \mathcal{B}^\top * \mathcal{A}^\top$.

Definition 2.5 ([17, 32]) For $\mathcal{A} \in \mathbb{R}^{N_1 \times N_2 \times N_3}$, its Moore–Penrose pseudoinverse, denoted $\mathcal{A}^\dagger$, is the unique $N_2 \times N_1 \times N_3$ tensor that satisfies $\mathrm{bcirc}(\mathcal{A}^\dagger) = \mathrm{bcirc}(\mathcal{A})^\dagger$.

Fact 2. [32, Theorem 3.1] For $\mathcal{A} \in \mathbb{R}^{N_1 \times N_2 \times N_3}$,

$$\mathcal{A} * \mathcal{A}^\dagger * \mathcal{A} = \mathcal{A}, \ \mathcal{A}^\dagger * \mathcal{A} * \mathcal{A}^\dagger = \mathcal{A}^\dagger, \ \left(\mathcal{A} * \mathcal{A}^\dagger\right)^\top = \mathcal{A} * \mathcal{A}^\dagger, \ \left(\mathcal{A}^\dagger * \mathcal{A}\right)^\top = \mathcal{A}^\dagger * \mathcal{A}.$$

These are the tensor analogues of the matrix Penrose conditions; they follow from the matrix Penrose conditions applied to $\mathrm{bcirc}(\mathcal{A})$, together with Definition 2.5, Fact 1, and the injectivity of $\mathrm{bcirc}(\cdot)$.

Fact 3. For $\mathcal{A} \in \mathbb{R}^{N_1 \times N_2 \times N_3}$, the following holds:

$$(\mathcal{A}^\dagger)^\top = (\mathcal{A}^\top)^\dagger, \ \ \mathcal{A}^\top * \mathcal{A} * \mathcal{A}^\dagger = \mathcal{A}^\top.$$

Proof The first assertion holds from Fact 1 and the Definition 2.5. For the second, it suffices to use the Fact 2 and the first assertion to obtain

$$\mathcal{A}^\top * \mathcal{A} * \mathcal{A}^\dagger = \mathcal{A}^\top * (\mathcal{A} * \mathcal{A}^\dagger)^\top = \mathcal{A}^\top * (\mathcal{A}^\dagger)^\top * \mathcal{A}^\top = \mathcal{A}^\top$$

□

Definition 2.6 ([1]) A tensor $\mathcal{P}$ is a projector if $\mathcal{P} * \mathcal{P} = \mathcal{P}$, and it is orthogonal if $\mathcal{P}^\top = \mathcal{P}$.

Lemma 2.1 ([1]) *For $\mathcal{A} \in \mathbb{R}^{N_1 \times N_2 \times N_3}$, the tensors $\mathcal{P} = \mathcal{A} * \mathcal{A}^\dagger$ and $\mathcal{Q} = \mathcal{A}^\dagger * \mathcal{A}$ are orthogonal projection tensors.*

The proof of this Lemma follows directly from Facts 1 & 2, and the Definition 2.6.

Since Lemma 2.1 holds for any tensor $\mathcal{A} \in \mathbb{R}^{N_1 \times N_2 \times N_3}$, it applies in particular to the sub-tensors used in the TGDBEK algorithm in Section 3. More precisely, for any index set $U_k \subseteq [N_2]$, the tensor $\left(\mathcal{A}^\top_{:,U_k,:}\right)^\dagger * \mathcal{A}^\top_{:,U_k,:}$ is an orthogonal projector. Similarly, for any index set $J_k \subseteq [N_1]$, the tensor $\mathcal{A}_{J_k,:,:} * (\mathcal{A}_{J_k,:,:})^\dagger$ is also an orthogonal

projector.

Definition 2.7 ([1]) For $\mathcal{A} \in \mathbb{R}^{N_1 \times N_2 \times N_3}$, its spectral norm and the Frobenius norm are defined as follows

$$\|\mathcal{A}\|_2 := \|\text{bcirc}(\mathcal{A})\|_2, \qquad \|\mathcal{A}\|_F := \sqrt{\langle \mathcal{A}, \mathcal{A} \rangle} = \sqrt{\sum_{i,j,k} \mathcal{A}_{ijk}^2} \tag{6}$$

From this definition, we see that $\sigma_{\max}(\mathcal{A}) := \sigma_{\max}(\text{bcirc}(\mathcal{A}))$ and similarly, $\sigma_{\min}^+(\mathcal{A}) := \sigma_{\min}^+(\text{bcirc}(\mathcal{A}))$. The t-inner product is the sum of all A_{ijk} squared. The relation between $\|\text{bcirc}(\mathcal{A})\|_F$ and $\|\mathcal{A}\|_F$ is given as

$$\|\text{bcirc}(\mathcal{A})\|_F = \sqrt{N_3}\|\mathcal{A}\|_F \tag{7}$$

Lemma 2.2 ([15, 17]) *For $\mathcal{A} \in \mathbb{R}^{N_1 \times N_2 \times N_3}$ and any $\mathcal{X} \in \mathbb{R}^{N_2 \times K \times N_3}$, we have*

$$\|\mathcal{A} * \mathcal{X}\|_F^2 \leq \sigma_{\max}^2(\text{bcirc}(\mathcal{A}))\, \|\mathcal{X}\|_F^2. \tag{8}$$

Moreover, for $\mathcal{X} \in \text{R}(\mathcal{A}^\top) = \text{null}(\mathcal{A})^\perp$

$$\|\mathcal{A} * \mathcal{X}\|_F^2 \geq {\sigma_{\min}^+}^2(\text{bcirc}(\mathcal{A}))\, \|\mathcal{X}\|_F^2. \tag{9}$$

Lemma 2.3 (Tensor Pythagorean theorem) *Let $\mathcal{P} \in \mathbb{R}^{N \times N \times N_3}$ be a t-orthogonal projector. Then for any tensor $\mathcal{X} \in \mathbb{R}^{N \times K \times N_3}$ we have,*

$$\|\mathcal{X}\|_F^2 = \|\mathcal{P} * \mathcal{X}\|_F^2 + \|(\mathcal{I} - \mathcal{P}) * \mathcal{X}\|_F^2. \tag{10}$$

Proof Since $\mathcal{P}$ is a t-orthogonal projector, $(\mathcal{I} - \mathcal{P})$ is also a t-orthogonal projector with $\mathcal{P} * (\mathcal{I} - \mathcal{P}) = \mathcal{O}$, so their ranges are orthogonal. Therefore $\langle \mathcal{P} * \mathcal{X},\ (\mathcal{I} - \mathcal{P}) * \mathcal{X} \rangle = 0$, and the identity follows from $\mathcal{X} = \mathcal{P} * \mathcal{X} + (\mathcal{I} - \mathcal{P}) * \mathcal{X}$. □

Lemma 2.4 *Let $\mathcal{A} \in \mathbb{R}^{N_1 \times N_2 \times N_3}$ and $\mathcal{X}_* = \mathcal{A}^\dagger * \mathcal{B}$. Then:*

1. $\text{R}(\mathcal{A}^\dagger) = \text{R}(\mathcal{A}^\top)$ *and* $\text{R}((\mathcal{A}^\dagger)^\top) = \text{R}(\mathcal{A})$
2. $\mathcal{X}_* \in \text{R}(\mathcal{A}^\top)$ *is the unique minimum-norm least-squares solution to* $\mathcal{A} * \mathcal{X} = \mathcal{B}$
3. $\text{R}(\mathcal{A}^\top) \cap \text{null}(\mathcal{A}) = \{\mathcal{O}\}$

3 The TGDBEK algorithm

In this section, we construct the tensor greedy double extended Kaczmarz, as illustrated in Algorithm 1. We assume the system (1) is inconsistent, thus $\mathcal{B} \notin \text{R}(\mathcal{A})$. From the orthogonal decomposition $\text{R}(\mathcal{A}) \perp \text{null}(\mathcal{A}^\top)$ there must exist $\mathcal{Z} \in \text{null}(\mathcal{A}^\top)$ such that $\mathcal{B} = (\mathcal{B} - \mathcal{Z}) + \mathcal{Z}$ and $\mathcal{B} - \mathcal{Z} \in \text{R}(\mathcal{A})$, in order the system $\mathcal{A} * \mathcal{X} = \mathcal{B} - \mathcal{Z}$ be consistent. Solving (1) is thus equivalent to finding $\mathcal{Z}$ and $\mathcal{X}$ that satisfy $\mathcal{A}^\top * \mathcal{Z} =$

$\mathcal{O}$ and $\mathcal{A} * \mathcal{X} = \mathcal{B} - \mathcal{Z}$, which obviously is a tensor analogue of the extended Kaczmarz decomposition of Zouzias and Freris [38]. The resulting $\mathcal{X}$ is the minimum-norm least-squares solution $\mathcal{X}_* = \mathcal{A}^\dagger * \mathcal{B}$ of eq. (1).

The TREK Algorithm (eqs. (4)–(5)) solves the two problems above by applying the tensor analogue of RK twice at each iteration: once on $\mathcal{A}^\top * \mathcal{Z} = \mathcal{O}$, the iterate $\mathcal{Z}^{(k)}$ is updated along a single, randomly drawn lateral slice index, and once on $\mathcal{A} * \mathcal{X} = \mathcal{B} - \mathcal{Z}$, and the iterate $\mathcal{X}^{(k)}$ is updated along a single, randomly drawn horizontal slice index. Each slice of $\mathcal{A}$ is thus used once at each iteration to update $\mathcal{Z}$ and $\mathcal{X}$, exactly as in REK style (2)–(3). Following Sun and Qin [10], we can construct blocks of lateral(column) and horizontal(row) slice indexes using a greedy strategy, instead of simply sampling individual slices. The greedy deterministically selects, at the k-th iteration, a block of lateral slice set U_k and a block of horizontal slice set J_k of $\mathcal{A}$, then update the $\mathcal{X}$ and Z iterates slightly as in (4)–(5). In fact, the current iterates are orthogonally projected onto the solution spaces of $\mathcal{A}_{:,U_k,:}^\top * \mathcal{Z} = \mathcal{O}$ and $\mathcal{A}_{J_k,:,:} * \mathcal{X}^{(k)} = \mathcal{B}_{J_k,:,:} - \mathcal{Z}_{J_k,:,:}^{(k+1)}$. The sets U_k and J_k are non-empty by design and are given in the Algorithm 1 below. By selecting the threshold η, we control the sizes of U_k and J_k. Precisely, the objective here is to identify the largest entries of the residual tensors $-\mathcal{A}^\top * \mathcal{Z}^{(k)}$ and $-\mathcal{A} * \mathcal{X}^{(k)} + \mathcal{Z}^{(k+1)} - \mathcal{B}$ at each iteration, ensure that these entries are prioritized for elimination and so to accelerate the convergence of $\mathcal{Z}^{(k)}$ and $\mathcal{X}^{(k)}$. Whereas TREBK or TREGBK instead update a static, predefined block of slices and randomly selects them at each step, TGDBEK builds these blocks at each iteration from the current residual and are not necessarily randomly sampled.

Algorithm 1: Tensor Greedy Double Block Extended Kaczmarz (TGDBEK)

Input: $\mathcal{A} \in \mathbb{R}^{N_1 \times N_2 \times N_3}$, $\mathcal{B} \in \mathbb{R}^{N_1 \times K \times N_3}$, iterations ℓ, parameter $\eta \in (0, 1]$.

Initialize: $\mathcal{X}^{(0)} = \mathcal{O}$ and $\mathcal{Z}^{(0)} = \mathcal{B}$.

for $k = 0, 1, \ldots, \ell - 1$ **do**

$$\varepsilon_k^z = \eta \max_{1 \le j \le N_2} \left\{ \frac{\left\| \mathcal{A}_{:,j,:}{}^\top * \mathcal{Z}^{(k)} \right\|_{\mathrm{F}}^2}{\left\| \mathcal{A}_{:,j,:} \right\|_{\mathrm{F}}^2} \right\}$$

$$U_k = \left\{ j \in [N_2] : \left\| \mathcal{A}_{:,j,:}{}^\top * \mathcal{Z}^{(k)} \right\|_{\mathrm{F}}^2 \ge \varepsilon_k^z \left\| \mathcal{A}_{:,j,:} \right\|_{\mathrm{F}}^2 \right\}$$

$$\text{Update } \mathcal{Z}^{(k+1)} = \mathcal{Z}^{(k)} - \mathcal{A}_{:,U_k,:} * (\mathcal{A}_{:,U_k,:})^\dagger * \mathcal{Z}^{(k)}$$

$$\varepsilon_k^x = \eta \max_{1 \le i \le N_1} \left\{ \frac{\left\| \mathcal{B}_{i,:,:} - \mathcal{Z}_{i,:,:}^{(k+1)} - \mathcal{A}_{i,:,:} * \mathcal{X}^{(k)} \right\|_{\mathrm{F}}^2}{\left\| \mathcal{A}_{i,:,:} \right\|_{\mathrm{F}}^2} \right\}$$

$$J_k = \left\{ i \in [N_1] : \left\| \mathcal{B}_{i,:,:} - \mathcal{Z}_{i,:,:}^{(k+1)} - \mathcal{A}_{i,:,:} * \mathcal{X}^{(k)} \right\|_{\mathrm{F}}^2 \ge \varepsilon_k^x \left\| \mathcal{A}_{i,:,:} \right\|_F^2 \right\}$$

$$\text{Update } \mathcal{X}^{(k+1)} = \mathcal{X}^{(k)} + (\mathcal{A}_{J_k,:,:})^\dagger * \left(\mathcal{B}_{J_k,:,:} - \mathcal{Z}_{J_k,:,:}^{(k+1)} - \mathcal{A}_{J_k,:,:} * \mathcal{X}^{(k)} \right)$$

end

The Theorem 3.2 states the lower bound for the sequence $\{\mathcal{X}^{(k)}\}$ generated by the Algorithm 1 and proves that it converges to the minimum-norm solution of eq. (1). Its

proof uses the Lemma 3.1 which establishes the convergence of the sequence $\{\mathcal{Z}^{(k)}\}$ generated by the Algorithm 1.

Lemma 3.1 (Linear Convergence of the $\mathcal{Z}$-Iterates) *Let $\mathcal{B}^{\perp} := \mathcal{B} - \mathcal{A} * \mathcal{A}^{\dagger} * \mathcal{B}$ denote the component of $\mathcal{B}$ orthogonal to $\mathrm{R}(\mathcal{A})$. The sequence $\{\mathcal{Z}^{(k)}\}_{k\geq 0}$ generated by Algorithm 1 converges linearly to $\mathcal{B}^{\perp}$ satisfying, for all $k \geq 0$:*

$$\|\mathcal{Z}^{(k+1)} - \mathcal{B}^{\perp}\|_F^2 \leq \beta \, \|\mathcal{Z}^{(k)} - \mathcal{B}^{\perp}\|_F^2, \tag{11}$$

with contraction factor: $\beta = 1 - \eta \frac{{\sigma_{\min}^{+}}^2(\mathrm{bcirc}(\mathcal{A}))}{\|\mathrm{bcirc}(\mathcal{A})\|_F^2} \in (0,1)$.

Proof of Lemma 3.1 From Fact 3 we obtain

$$\mathcal{A}^{\top} * \mathcal{B}^{\perp} = \mathcal{A}^{\top} * \mathcal{B} - \mathcal{A}^{\top} * \mathcal{A} * \mathcal{A}^{\dagger} * \mathcal{B} = \mathcal{A}^{\top} * \mathcal{B} - \mathcal{A}^{\top} * \mathcal{B} = 0.$$

In particular, for any $U_k \subset [N_2]$ we get $\mathcal{A}_{:,U_k,:}^{\top} * \mathcal{B}^{\perp} = \mathcal{O}$. Now, using the fact $\mathcal{A} * \mathcal{A}^{\dagger} = \left(\mathcal{A} * \mathcal{A}^{\dagger}\right)^{\top} = (\mathcal{A}^{\dagger})^{\top} * \mathcal{A}^{\top} = (\mathcal{A}^{\top})^{\dagger} * \mathcal{A}^{\top}$ (which follows from Penrose condition (iii) and Fact 3) and the $\mathcal{Z}-$ update in Algorithm 1, we can get

$$\begin{aligned}
\mathcal{Z}^{(k+1)} - \mathcal{B}^{\perp} &= \left(\mathcal{Z}^{(k)} - \mathcal{B}^{\perp}\right) - \left(\mathcal{A}_{:,U_k,:}^{\top}\right)^{\dagger} * \mathcal{A}_{:,U_k,:}^{\top} * \left(\mathcal{Z}^{(k)} - \mathcal{B}^{\perp}\right) \\
&= \left(\mathcal{I} - \left(\mathcal{A}_{:,U_k,:}^{\top}\right)^{\dagger} * \mathcal{A}_{:,U_k,:}^{\top}\right) * \left(\mathcal{Z}^{(k)} - \mathcal{B}^{\perp}\right).
\end{aligned}$$

Since $\left(\mathcal{A}_{:,U_k,:}^{\top}\right)^{\dagger} \mathcal{A}_{:,U_k,:}^{\top}$ is a t-orthogonal projector, the Pythagorean theorem (Lemma 2.3) gives

$$\|\mathcal{Z}^{(k+1)} - \mathcal{B}^{\perp}\|_F^2 = \|\mathcal{Z}^{(k)} - \mathcal{B}^{\perp}\|_F^2 - \| \left(\mathcal{A}_{:,U_k,:}^{\top}\right)^{\dagger} * \mathcal{A}_{:,U_k,:}^{\top} * \left(\mathcal{Z}^{(k)} - \mathcal{B}^{\perp}\right) \|_F^2. \tag{12}$$

We need to upper bound the second term of the addition above. Since $\mathcal{A}_{:,U_k,:}^{\top} * \mathcal{Z}^{(k)} \in \mathrm{R}(\mathcal{A}_{:,U_k,:}^{\top}) = \mathrm{R}\Big(((\mathcal{A}_{:,U_k,:}^{\top})^{\dagger})^{\top}\Big)$, the lower singular value bound in Lemma 2.2 applied to $(\mathcal{A}_{:,U_k,:}^{\top})^{\dagger}$ gives

$$\begin{aligned}
\| \left(\mathcal{A}_{:,U_k,:}^{\top}\right)^{\dagger} * \mathcal{A}_{:,U_k,:}^{\top} * \left(\mathcal{Z}^{(k)} - \mathcal{B}^{\perp}\right) \|_F^2 &= \| \left(\mathcal{A}_{:,U_k,:}^{\top}\right)^{\dagger} * \mathcal{A}_{:,U_k,:}^{\top} * \mathcal{Z}^{(k)} \|_F^2 \\
&\geq {\sigma_{\min}^{+}}^2((\mathcal{A}_{:,U_k,:}^{\top})^{\dagger}) \|\mathcal{A}_{:,U_k,:}^{\top} * \mathcal{Z}^{(k)}\|_F^2 \quad \text{(by Lemma 2.2)} \\
&= {\sigma_{\min}^{+}}^2(\mathrm{bcirc}(\mathcal{A}_{:,U_k,:}^{\top})^{\dagger}) \sum_{j \in U_k} \|\mathcal{A}_{:,j,:}^{\top} * \mathcal{Z}^{(k)}\|_F^2 \\
&\geq {\sigma_{\min}^{+}}^2(\mathrm{bcirc}(\mathcal{A}_{:,U_k,:}^{\top})^{\dagger}) \epsilon_k^z \|\mathcal{A}_{:,U_k,:}\|_F^2 \\
&= \frac{\|\mathcal{A}_{:,U_k,:}\|_F^2}{\sigma_{\max}^2(\mathrm{bcirc}(\mathcal{A}_{:,U_k,:}))} \epsilon_k^z \\
&\geq \frac{1}{N_3} \epsilon_k^z,
\end{aligned} \tag{13}$$

where in the last inequality we use eq. (7), and the facts $\sigma_{\max}^2(\boldsymbol{M}^{\top}) = \sigma_{\max}^2(\boldsymbol{M})$ and $\|\boldsymbol{M}\|_F^2 \geq \|\boldsymbol{M}\|_2$, for any matrix $\boldsymbol{M}$. Now, observe that

$$\mathcal{A}_{:,U_{k-1},:}^{\top} * \mathcal{Z}^{(k)} = \mathcal{A}_{:,U_{k-1},:}^{\top} * \left(\mathcal{Z}^{(k-1)} - \mathcal{A}_{:,U_{k-1},:} * \mathcal{A}_{:,U_{k-1},:}^{\dagger} * \mathcal{Z}^{(k-1)}\right)$$

$$= \mathcal{A}^{\top}_{:,U_{k-1},:} * \mathcal{Z}^{(k-1)} - \mathcal{A}^{\top}_{:,U_{k-1},:} * \mathcal{A}_{:,U_{k-1},:} * \mathcal{A}^{\dagger}_{:,U_{k-1},:} * \mathcal{Z}^{(k-1)} = \mathcal{O},$$

so that for all $k = 0, 1, \ldots,$

$$\begin{aligned}
\|\mathcal{A}^{\top} * \mathcal{Z}^{(k)}\|_F^2 &= \sum_{j \in U_{k-1}} \|\mathcal{A}^{\top}_{:,j,:} * \mathcal{Z}^{(k)}\|_F^2 + \sum_{j \in U_{k-1}^c} \|\mathcal{A}^{\top}_{:,j,:} * \mathcal{Z}^{(k)}\|_F^2 \\
&= \sum_{j \in U_{k-1}^c} \|\mathcal{A}^{\top}_{:,j,:} * \mathcal{Z}^{(k)}\|_F^2 = \sum_{j \in U_{k-1}^c} \frac{\|\mathcal{A}^{\top}_{:,j,:} * \mathcal{Z}^{(k)}\|_F^2}{\|\mathcal{A}_{:,j,:}\|_F^2} \|\mathcal{A}_{:,j,:}\|_F^2 \\
&\leq \max_{j \in [N_2]} \left\{ \frac{\|\mathcal{A}^{\top}_{:,j,:} * \mathcal{Z}^{(k)}\|_F^2}{\|\mathcal{A}_{:,j,:}\|_F^2} \right\} \left(\|\mathcal{A}\|_F^2 - \|\mathcal{A}_{:,U_k,:}\|_F^2 \right) \\
&\leq \max_{j \in [N_2]} \left\{ \frac{\|\mathcal{A}^{\top}_{:,j,:} * \mathcal{Z}^{(k)}\|_F^2}{\|\mathcal{A}_{:,j,:}\|_F^2} \right\} \|\mathcal{A}\|_F^2,
\end{aligned}$$

since $\|\mathcal{A}\|_F^2 - \|\mathcal{A}_{:,U_k,:}\|_F^2 \leq \|\mathcal{A}\|_F^2$. Hence

$$\begin{aligned}
\epsilon_k^z = \eta \max_{j \in [N_2]} \left\{ \frac{\|\mathcal{A}^{\top}_{:,j,:} * \mathcal{Z}^{(k)}\|_F^2}{\|\mathcal{A}_{:,j,:}\|_F^2} \right\} &\geq \eta \frac{\|\mathcal{A}^{\top} * \mathcal{Z}^{(k)}\|_F^2}{\|\mathcal{A}\|_F^2} \\
&\geq \eta \frac{{\sigma_{\min}^{+}}^2(\mathrm{bcirc}(\mathcal{A}^{\top}))}{\|\mathcal{A}\|_F^2} \|\mathcal{Z}^{(k)} - \mathcal{B}^{\perp}\|_F^2 \\
&= \eta N_3 \frac{{\sigma_{\min}^{+}}^2(\mathrm{bcirc}(\mathcal{A}^{\top}))}{\|\mathrm{bcirc}(\mathcal{A})\|_F^2} \|\mathcal{Z}^{(k)} - \mathcal{B}^{\perp}\|_F^2. \qquad (14)
\end{aligned}$$

The last inequality holds from Lemma (2.2) applied to $\mathcal{A}^{\top}$ and again eq. (14) uses the relation between $\|\mathcal{A}\|$ and $\|\mathrm{bcirc}(\mathcal{A})\|_F$ given in eq. (7). Substituting (14) into (13), we obtain the lower bound

$$\| \left(\mathcal{A}^{\top}_{:,U_k,:} \right)^{\dagger} * \mathcal{A}^{\top}_{:,U_k,:} * \left(\mathcal{Z}^{(k)} - \mathcal{B}^{\perp} \right) \|_F^2 \geq \eta \frac{{\sigma_{\min}^{+}}^2(\mathrm{bcirc}(\mathcal{A}^{\top}))}{\|\mathrm{bcirc}(\mathcal{A}^{\top})\|_F^2} \|\mathcal{Z}^{(k)} - \mathcal{B}^{\perp}\|_F^2. \qquad (15)$$

Finally, the combination of (15) and (12) gives

$$\|\mathcal{Z}^{(k+1)} - \mathcal{B}^{\perp}\|_F^2 \leq \underbrace{\left(1 - \eta \frac{{\sigma_{\min}^{+}}^2(\mathrm{bcirc}(\mathcal{A}^{\top}))}{\|\mathrm{bcirc}(\mathcal{A})\|_F^2} \right)}_{=: \beta} \|\mathcal{Z}^{(k)} - \mathcal{B}^{\perp}\|_F^2.$$

□

Theorem 3.2 (Linear Convergence of TGDBEK) *Assume the system (1) is inconsistent and let $\mathcal{B}^{\perp} := \mathcal{B} - \mathcal{A} * \mathcal{A}^{\dagger} * \mathcal{B} \in \mathrm{null}(\mathcal{A}^{\top})$ denote the component of $\mathcal{B}$ orthogonal to the range space $R(\mathcal{A})$.*

Let $J_k \subseteq [N_1]$ be the active row blocks selected by Algorithm 1 at iteration k. Then, the sequence $\{\mathcal{X}^{(k)}\}_{k \geq 0}$ generated by Algorithm 1 converges linearly in the Frobenius norm to the unique minimum-norm least-squares solution $\mathcal{X}_* = \mathcal{A}^{\dagger} * \mathcal{B}$.

Specifically, for all $k \geq 0$:

$$\|\mathcal{X}^{(k+1)} - \mathcal{X}_*\|_F^2 \leq \alpha^{k+1} \|\mathcal{X}^{(0)} - \mathcal{X}_*\|_F^2 + \gamma \left(\frac{\alpha^{k+1} - \beta^{k+1}}{\alpha - \beta} \right) \beta \|\mathcal{B} - \mathcal{B}^{\perp}\|_F^2 \quad (\textit{if } \alpha \neq \beta), \qquad (16)$$

and if $\alpha = \beta$*:*

$$\|\mathcal{X}^{(k+1)} - \mathcal{X}_*\|_F^2 \leq \alpha^{k+1}\|\mathcal{X}^{(0)} - \mathcal{X}_*\|_F^2 + \gamma(k+1)\alpha^{k+1}\|\mathcal{B} - \mathcal{B}^{\perp}\|_F^2, \tag{17}$$

where

$$\alpha = 1 - \theta \cdot \kappa_{\text{block}}(\mathcal{A}) \in (0,1), \quad \gamma = \frac{1}{\left(\sigma^+_{\min,\text{block}}(\mathcal{A})\right)^2}, \quad \beta = 1 - \eta\,\frac{{\sigma^+_{\min}}^2(\text{bcirc}(\mathcal{A}))}{\|\text{bcirc}(\mathcal{A})\|_F^2} \in (0,1),$$

with $\eta \in (0,1]$ *the parameter of Algorithm 1.*

The proof of this theorem relies on the previous contraction lemma 3.1.

Proof of the main Theorem 3.2. Since $\mathcal{B}_{J_k,:,:} = \mathcal{A}_{J_k,:,:} * \mathcal{X}^* + \mathcal{B}^{\perp}_{J_k,:,:}$, the $\mathcal{X}$-update becomes

$$\mathcal{X}^{(k+1)} = \mathcal{X}^{(k)} - \mathcal{A}^{\dagger}_{J_k,:,:} * \left(\mathcal{A}_{J_k,:,:} * \left(\mathcal{X}^{(k)} - \mathcal{X}^*\right) + \left(\mathcal{Z}^{(k+1)}_{J_k,:,:} - \mathcal{B}^{\perp}_{J_k,:,:}\right)\right).$$

Subtracting $\mathcal{X}_*$ from both sides yields

$$\mathcal{X}^{(k+1)} - \mathcal{X}_* = \left(\mathcal{I} - \mathcal{A}^{\dagger}_{J_k,:,:} * \mathcal{A}_{J_k,:,:}\right) * \left(\mathcal{X}^{(k)} - \mathcal{X}^*\right) + \mathcal{A}^{\dagger}_{J_k,:,:} * \left(\mathcal{Z}^{(k+1)}_{J_k,:,:} - \mathcal{B}^{\perp}_{J_k,:,:}\right). \tag{18}$$

$\mathcal{A}^{\dagger}_{J_k,:,:} * \mathcal{A}_{J_k,:,:}$ is a t-orthogonal projector by Lemma 2.1 , so the range $\text{R}\left(\mathcal{I} - \mathcal{A}^{\dagger}_{J_k,:,:} * \mathcal{A}_{J_k,:,:}\right)$ is orthogonal to $\text{R}(\mathcal{A}^{\dagger}_{J_k,:,:})$. The Pythagorean theorem (Lemma 2.3) gives

$$\|\mathcal{X}^{(k+1)} - \mathcal{X}_*\|_F^2 = \| \left(\mathcal{I} - \mathcal{A}^{\dagger}_{J_k,:,:} * \mathcal{A}_{J_k,:,:}\right) * \left(\mathcal{X}^{(k)} - \mathcal{X}^*\right) \|_F^2 + \|\mathcal{A}^{\dagger}_{J_k,:,:} * \left(\mathcal{Z}^{(k+1)}_{J_k,:,:} - \mathcal{B}^{\perp}_{J_k,:,:}\right) \|_F^2. \tag{19}$$

It suffices to upper bound the two terms of the right-hand side in equation above. For the first term, note that we can decompose $\mathcal{X}^{(k+1)} - \mathcal{X}_*$ as follow

$$\mathcal{X}^{(k)} - \mathcal{X}_* = \left(\mathcal{X}^{(k)} - \mathcal{X}_*\right)_{\text{R}((\mathcal{A}_{J_k,:,:})^\top)} + \left(\mathcal{X}^{(k)} - \mathcal{X}_*\right)_{\text{R}((\mathcal{A}_{J_k,:,:})^\top)^\perp}. \tag{20}$$

Let recall $\theta_k \in (0,1]$ defined by

$$\theta_k = \frac{\left\|(\mathcal{X}^{(k)} - \mathcal{X}_*)_{\text{R}((\mathcal{A}_{J_k,:,:})^\top)}\right\|_F^2}{\|\mathcal{X}^{(k)} - \mathcal{X}_*\|_F^2}, \tag{21}$$

The constant θ_k cannot be zero. In fact, $\theta_k = 0$ means its numerator is also zero, which implies $\left(\mathcal{X}^{(k)} - \mathcal{X}_*\right)_{\text{R}((\mathcal{A}_{J_k,:,:})^\top)} = 0$ or $\mathcal{X}^{(k+1)} - \mathcal{X}_* \in \ker(\mathcal{A}_{J_k,:,:})) = \text{R}((\mathcal{A}_{J_k,:,:})^\top)^\perp$, i.e. $\mathcal{A}_{J_k,:,:}\left(\mathcal{X}^{(k+1)} - \mathcal{X}_*\right) = 0$. Hence, from the decomposition in eq (18), it results $\mathcal{X}^{(k+1)} - \mathcal{X}_* = \mathcal{A}^{\dagger}_{J_k,:,:} * \left(\mathcal{Z}^{(k+1)}_{J_k,:,:} - \mathcal{B}^{\perp}_{J_k,:,:}\right) = \mathcal{A}^{\dagger}_{J_k,:,:} * \mathcal{Z}^{(k+1)}_{J_k,:,:}$. In particular, $\mathcal{X}^{(k+1)} - \mathcal{X}_* \in \text{R}((\mathcal{A}_{J_k,:,:})^\dagger) = \text{R}((\mathcal{A}_{J_k,:,:})^\top)$. We have both $\mathcal{X}^{(k+1)} - \mathcal{X}_* \in \text{R}((\mathcal{A}_{J_k,:,:})^\top)$ and in its orthogonal, i.e. $\mathcal{X}^{(k+1)} - \mathcal{X}_* = 0$, for k.
The first term in the right-hand side of (19) satisfies

$$\begin{aligned} \| \left(\mathcal{I} - \mathcal{A}^{\dagger}_{J_k,:,:} * \mathcal{A}_{J_k,:,:}\right) * \left(\mathcal{X}^{(k)} - \mathcal{X}^*\right) \|_F^2 &= \|\mathcal{X}^{(k)} - \mathcal{X}_*\|_F^2 \qquad (22)\\ &\quad - \|\mathcal{A}^{\dagger}_{J_k,:,:} * \mathcal{A}_{J_k,:,:} * \left(\mathcal{X}^{(k)} - \mathcal{X}^*\right) \|_F^2 \\ &\leq \|\mathcal{X}^{(k)} - \mathcal{X}_*\|_F^2 - {\sigma^+_{\min}}^2(\mathcal{A}^{\dagger}_{J_k,:,:}) \qquad (23) \end{aligned}$$

$$
\begin{aligned}
&\times \|\mathcal{A}_{J_k,:,:} * \left(\mathcal{X}^{(k)} - \mathcal{X}^*\right)\|_F^2 \\
&\leq \|\mathcal{X}^{(k)} - \mathcal{X}_*\|_F^2 - {\sigma_{\min}^+}^2(\mathcal{A}^\dagger_{J_k,:,:}){\sigma_{\min}^+}^2(\mathcal{A}_{J_k,:,:}) \\
&\times \| \left(\mathcal{X}^{(k)} - \mathcal{X}_*\right)_{\mathrm{R}((\mathcal{A}_{J_k,:,:})^\top)} \|_F \qquad (24) \\
&= \|\mathcal{X}^{(k)} - \mathcal{X}_*\|_F^2 - \frac{{\sigma_{\min}^+}^2(\mathrm{bcirc}(\mathcal{A}_{J_k,:,:}))}{\sigma_{\max}^2(\mathrm{bcirc}(\mathcal{A}_{J_k,:,:}))} \\
&\times \theta_k \|\mathcal{X}^{(k)} - \mathcal{X}_*\|_F^2 \qquad (25) \\
&= \left(1 - \theta_k \frac{{\sigma_{\min}^+}^2(\mathrm{bcirc}(\mathcal{A}_{J_k,:,:}))}{\sigma_{\max}^2(\mathrm{bcirc}(\mathcal{A}_{J_k,:,:}))}\right) \|\mathcal{X}^{(k)} - \mathcal{X}_*\|_F^2 \\
&\leq (1 - \theta \cdot \kappa_{\mathrm{block}}(\mathcal{A}))\, \|\mathcal{X}^{(k)} - \mathcal{X}_*\|_F^2, \qquad (26)
\end{aligned}
$$

where the eq. (22) holds from Lemma 2.3, eq. (23) uses Lemma 2.2 part two and since by design, the greedy row blocks J_k generated at each iteration by Algorithm 1 are non-empty, we have $J_k \in \Omega_r$. To see eq. (24), with the help of Lemma 2.4 we get $\mathrm{R}((\mathcal{A}_{J_k,:,:})^\top)^\perp = \ker(\mathcal{A}_{J_k,:,:})$, then applying $\mathcal{A}_{J_k,:,:}$ in eq. (20) gives

$$
\mathcal{A}_{J_k,:,:}\left(\mathcal{X}^{(k)} - \mathcal{X}^*\right) = \mathcal{A}_{J_k,:,:}\left(\mathcal{X}^{(k)} - \mathcal{X}_*\right)_{\mathrm{R}((\mathcal{A}_{J_k,:,:})^\top)}.
$$

Since, $\left(\mathcal{X}^{(k)} - \mathcal{X}_*\right)_{\mathrm{R}((\mathcal{A}_{J_k,:,:})^\top)} \in \mathrm{R}((\mathcal{A}_{J_k,:,:})^\top)$, apply Lemma 2.2 to get eq. (24), i.e.

$$
\begin{aligned}
\|\mathcal{A}_{J_k,:,:}\left(\mathcal{X}^{(k)} - \mathcal{X}^*\right)\|_F^2 &= \|\mathcal{A}_{J_k,:,:}\left(\mathcal{X}^{(k)} - \mathcal{X}_*\right)_{\mathrm{R}((\mathcal{A}_{J_k,:,:})^\top)}\| \\
&\geq {\sigma_{\min}^+}^2(\mathcal{A}_{J_k,:,:})\| \left(\mathcal{X}^{(k)} - \mathcal{X}_*\right)_{\mathrm{R}((\mathcal{A}_{J_k,:,:})^\top)} \|_F^2.
\end{aligned}
$$

Furthermore, eq. (25) uses ${\sigma_{\min}^+}^2(\boldsymbol{M}^\dagger) = \frac{1}{\sigma_{\max}^2(\boldsymbol{M})}$ which holds for any matrix $\boldsymbol{M}$. Using Lemma 2.2, the second term in the right-hand side of (19) satisfies,

$$
\begin{aligned}
\|\mathcal{A}^\dagger_{J_k,:,:} * \left(\mathcal{Z}^{(k+1)}_{J_k,:,:} - \mathcal{B}^\perp_{J_k,:,:}\right)\|_F^2 &\leq \sigma_{\max}^2(\mathrm{bcirc}((\mathcal{A}_{J_k,:,:})^\dagger))\|\mathcal{Z}^{(k+1)}_{J_k,:,:} - \mathcal{B}^\perp_{J_k,:,:}\|_F^2 \\
&= \frac{1}{{\sigma_{\min}^+}^2(\mathrm{bcirc}(\mathcal{A}_{J_k,:,:}))}\|\mathcal{Z}^{(k+1)}_{J_k,:,:} - \mathcal{B}^\perp_{J_k,:,:}\|_F^2 \\
&\leq \frac{1}{{\sigma_{\min}^+}^2(\mathrm{bcirc}(\mathcal{A}_{J_k,:,:}))}\|\mathcal{Z}^{(k+1)} - \mathcal{B}^\perp\|_F^2. \qquad (27)
\end{aligned}
$$

It results from (19), (26), and (27) that

$$
\begin{aligned}
\|\mathcal{X}^{(k+1)} - \mathcal{X}_*\|_F^2 &\leq \left(1 - \theta\frac{{\sigma_{\min}^+}^2(\mathrm{bcirc}(\mathcal{A}_{J,:,:}))}{\sigma_{\max}^2(\mathrm{bcirc}(\mathcal{A}_{J,:,:}))}\right)\|\mathcal{X}^{(k)} - \mathcal{X}_*\|_F^2 \\
&+ \frac{1}{{\sigma_{\min}^+}^2(\mathrm{bcirc}(\mathcal{A}_{J_k,:,:}))}\|\mathcal{Z}^{(k+1)} - \mathcal{B}^\perp\|_F^2 \\
&\leq \left(1 - \theta\frac{{\sigma_{\min}^+}^2(\mathrm{bcirc}(\mathcal{A}_{J,:,:}))}{\sigma_{\max}^2(\mathrm{bcirc}(\mathcal{A}_{J,:,:}))}\right)\|\mathcal{X}^{(k)} - \mathcal{X}_*\|_F^2 \\
&+ \frac{\beta}{{\sigma_{\min}^+}^2(\mathrm{bcirc}(\mathcal{A}_{J_k,:,:}))}\|\mathcal{Z}^{(k)} - \mathcal{B}^\perp\|_F^2
\end{aligned}
$$

$$\leq \left(1 - \theta \frac{{\sigma^+_{\min}}^2(\mathrm{bcirc}(\mathcal{A}_{J,:,:}))}{\sigma^2_{\max}(\mathrm{bcirc}(\mathcal{A}_{J,:,:}))}\right) \|\mathcal{X}^{(k)} - \mathcal{X}_*\|_F^2$$
$$+ \frac{\beta}{(\sigma^+_{\min,\mathrm{block}}(\mathcal{A}))^2} \|\mathcal{Z}^{(k)} - \mathcal{B}^\perp\|_F^2.$$

Now, let $\alpha = 1 - \theta \cdot \kappa_{\mathrm{block}}(\mathcal{A})$ and $\gamma = \frac{1}{(\sigma^+_{\min,\mathrm{block}}(\mathcal{A}))^2}$. Then we have

$$\begin{aligned}
\|\mathcal{X}^{(k+1)} - \mathcal{X}_*\|_F^2 &\leq \alpha \|\mathcal{X}^{(k)} - \mathcal{X}_*\|_F^2 + \gamma\beta \|\mathcal{Z}^{(k)} - \mathcal{B}^\perp\|_F^2 \\
&\leq \alpha \left(\alpha \|\mathcal{X}^{(k)} - \mathcal{X}_*\|_F^2 + \gamma\beta \|\mathcal{Z}^{(k)} - \mathcal{B}^\perp\|_F^2\right) + \gamma\beta^2 \|\mathcal{Z}^{(k-1)} - \mathcal{B}^\perp\|_F^2 \\
&\leq \cdots \\
&\leq \alpha^{k+1} \|\mathcal{X}^{(0)} - \mathcal{X}_*\|_F^2 + \gamma \sum_{l=0}^{k} \alpha^l \beta^{k-l+1} \|\mathcal{Z}^{(k-l)} - \mathcal{B}^\perp\|_F^2.
\end{aligned}$$

The map $m \mapsto \|\mathcal{Z}^{(m)} - \mathcal{B}^\perp\|_F^2$ is a contraction by Lemma 3.1. Therefore $\|\mathcal{Z}^{(m)} - \mathcal{B}^\perp\|_F^2 \leq \|\mathcal{Z}^{(0)} - \mathcal{B}^\perp\|_F^2$ for all $m \geq 0$. With the initialization $\mathcal{Z}^{(0)} = \mathcal{B}$ we have $\|\mathcal{Z}^{(0)} - \mathcal{B}^\perp\|_F^2 = \|\mathcal{B} - \mathcal{B}^\perp\|_F^2$, so every term $\|\mathcal{Z}^{(k-l)} - \mathcal{B}^\perp\|_F^2$ in the sum is bounded above by $\|\mathcal{B} - \mathcal{B}^\perp\|_F^2$. Hence

$$\begin{aligned}
\|\mathcal{X}^{(k+1)} - \mathcal{X}_*\|_F^2 &\leq \alpha^{k+1} \|\mathcal{X}^{(0)} - \mathcal{X}_*\|_F^2 + \gamma \|\mathcal{B} - \mathcal{B}^\perp\|_F^2 \sum_{l=0}^{k} \alpha^l \beta^{k-l+1} \\
&\leq \alpha^{k+1} \|\mathcal{X}^{(0)} - \mathcal{X}_*\|_F^2 + \gamma \left(\frac{\alpha^{k+1} - \beta^{k+1}}{\alpha - \beta}\right) \beta \, \|\mathcal{B} - \mathcal{B}^\perp\|_F^2
\end{aligned}$$

□

Remark 3.1 We cannot exclude $\alpha = \beta$ *a priori*, in which case the quotient $\frac{\alpha^{k+1} - \beta^{k+1}}{\alpha - \beta} = \sum_{l=0}^{k} \alpha^l \beta^{k-l+1}$ and the noise component of (16) becomes clearly

$$\gamma(k+1)\alpha^{k+1} \|\mathcal{B} - \mathcal{B}^\perp\|_F^2,$$

which obviously tends to zero as k increases (by the ratio test) since generally $\alpha < 1$. When $N_3 = 1$, $\mathcal{A} \in \mathbb{R}^{N_1 \times N_2 \times 1}$ is exactly a matrix $A \in \mathbb{R}^{N_1 \times N_2}$, the t-product $*$ reduces to the ordinary matrix multiplication, and $\mathrm{bcirc}(\mathcal{A}) = A$, so also $\|\mathrm{bcirc}(\mathcal{A})\|_F = \|\mathcal{A}\|_F$. Algorithm 1 then reduces to the GDBEK algorithm of [10], and the theorem above recovers [10, Theorem 2.1]: the only difference occurs in Lemma 3.1, where we use $\|\mathcal{A}\|_F^2 - \|\mathcal{A}_{:,U_k,:}\|_F^2 \leq \|\mathcal{A}\|_F^2$.

4 Numerical experiments

In this section, we conduct numerical experiments to demonstrate the effectiveness of TGDBEK method and compare it with TREK [14], TREBK [30], and TREGBK [30] methods. All the experiments were implemented in Python using PyTorch and run on an Apple MacBook Pro with an M3 chip. The complete code is publicly available at https://github.com/jnlandu/tensor-greedy-double-extended-kaczmarz.

For a fair comparison, we use the number of iteration steps ("IT") and the running time ("CPU"). Here, IT and CPU represent the arithmetic mean of the number of

iterations and running time required to each algorithm to reach the tolerance 10^{-5} over N independent trials. Sometimes we take $N = 5$, $N = 10$ or $N = 20$. The algorithm terminates when the relative squared error (RSE) satisfies

$$\text{RSE} := \frac{\|\mathcal{X}^{(k)} - \mathcal{X}_*\|_F^2}{\|\mathcal{X}_*\|_F^2} < 10^{-5},$$

where $\mathcal{X}_* = \mathcal{A}^\dagger * \mathcal{B}$ is generated using PyTorch, or when the total number of iterations is exceeded. We set the initial iterates $\mathcal{X}^{(0)} = \mathcal{O}$ and $\mathcal{Z}^{(0)} = \mathcal{B}$. Finally, the right-hand side $\mathcal{B}$ is obtained as $\mathcal{B} = \mathcal{A} * \mathcal{X}_* + \epsilon$, where the the noise ϵ, it is given as follows $\epsilon = a\zeta \frac{\|\mathcal{B}\|_F}{\|\zeta\|_F}$, with a a positive constant and $\zeta \sim \mathcal{N}(0, 1)$ a gaussian noise.

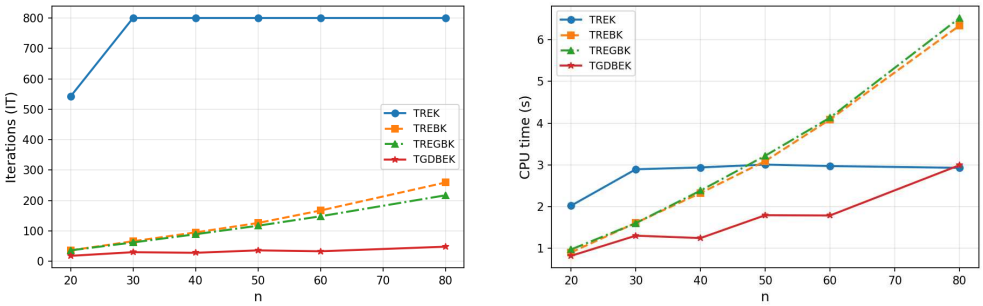
Example 4.1 (**Dense**) In this example, we apply TREK, TREBK, TREGBK and TGDBEK to solve a randomly generated dense tensor systems. The system tensor $\mathcal{A} \in \mathbb{R}^{500\times n\times 10}$ is constructed with i.i.d standard Gaussian entries, and similarly the exact solution $\mathcal{X}_* \in \mathbb{R}^{n\times 10\times 10}$. We take $n \in \{20, 30, 40, 50, 60, 80\}$. The right-hand side is computed as $\mathcal{B} = \mathcal{A} * \mathcal{X}_* + \epsilon$, where the noise ϵ is generated as described above with a noise level $a = 10^{-3}$. For TGDBEK we choose the threshold $\eta = 0.6$. The block-partitioned methods use sequential blocks of size $\tau = 10$, defined by $I_k = \{(k-1)\tau + 1, \ldots, \min\{k\tau, N\}\}$ for $k = 1, \ldots, \lfloor N/\tau \rfloor$, applied separately to $N_1 = 500$ and $N_2 = n$, resulting to 10 row blocks and $\lfloor n/10 \rfloor$ column blocks. For this example, the total number of iteration is 800 and result averaged over 5 trials. See Table 1 and Figs. 2–5.

Table 1 IT and CPU (averaged over 5 trials) for dense overdetermined tensor systems $\mathcal{A} \in \mathbb{R}^{500\times n\times 10}$, noise $a = 10^{-3}$, maximum 800 iterations. Bold: best per n.

Method	Metric	$n = 20$	$n = 30$	$n = 40$	$n = 50$	$n = 60$	$n = 80$
TREK	IT	547	800	800	800	800	800
	CPU(s)	1.114	1.458	1.569	1.569	1.473	1.559
TREBK	IT	36	69	97	122	160	247
	CPU(s)	0.705	1.415	2.478	2.849	3.760	5.835
TREGBK	IT	37	63	90	123	151	217
	CPU(s)	0.580	1.168	1.860	2.707	3.416	5.632
TGDBEK	IT	**23**	**28**	**33**	**35**	**37**	**44**
	CPU(s)	**0.497**	**0.652**	**1.007**	**1.401**	**1.460**	**2.385**

From Table 1, as for IT and CPU, it is clear that the TGDBEK method requires the fewest iterations and least running time across all values of n. For example, the TREK method exhausts the total iteration without even reaching the given tolerance for $n \geq 30$. At $n = 80$, TGDBEK converges in only 44 iterations, compared to 247 for TREBK and 217 for TREGBK, this is undoubtedly a considerable reduction in both IT and CPU.

As can be seen from Fig. 3, TGDBEK reaches RSE $< 10^{-5}$ in 23 iterations at $n = 20$ and 44 iterations at $n = 80$. Fig. 4 shows the convergence curves (RSE) against CPU. Again, we can see that TGDBEK achieves the given tolerance faster than all competing methods. At $n = 80$, TREK exhausted fast the total number of iterations without achieving the prescribed tolerance. Fig. 2 reports IT and CPU across all the values of n. It can be seen that TGDBEK has the least iteration count, and requires fewer CPU among all the methods. Fig. 5 plots the errors for both the x and z components against for $n = 20$ and $n = 80$. It reveals the convergence of the sequence $\{\mathcal{Z}^{(k)}\}$. Finally, Fig. 6 reports the cardinalities of greedy blocks U_k and J_k per iteration, and on average the TGDBEK method builds blocks of size 4 and 37.9 for $n = 20$ and 12.8 and 40 for $n = 80$, respectively for U_k and J_k.



(a) Iteration count (IT) vs. n.

(b) CPU time (s) vs. n.

Fig. 2 IT and CPU as functions of n for dense overdetermined systems ($\mathcal{A} \in \mathbb{R}^{500\times n\times 10}$, noise $a = 10^{-3}$, max 800 iterations, 5 trials). TREK exhausts the budget for $n \geq 30$.

Example 4.2 (**Sparse**) In this example, we use the TREK, TREBK, TREGBK, and TGDBEK methods to solve the tensor linear system (1), where the coefficient tensor $\mathcal{A}$ is a sparse tensor. The sparse tensor here $\mathcal{A}$ is constructed by reading five matrices from the SuiteSparse Matrix Collection [43] and reshaping them into third-order tensors: `nos5` ($468 \times 39 \times 12$), `ash85` ($85 \times 17 \times 5$), `Cities` ($55 \times 23 \times 2$), `WorldCities` ($315 \times 20 \times 5$), and `gre_216a` ($216 \times 24 \times 9$). The exact solution tensor $\mathcal{X}_*$ is randomly generated with standard Gaussian entries, the data tensor $\mathcal{B}$ is obtained by $\mathcal{B} = \mathcal{A} * \mathcal{X}_* + \epsilon$, where ϵ is generated as described above with noise level reported in Table 2, and the termination criterion is the same as Example 4.1. The greedy threshold is $\eta = 0.5$ for `nos5`, `ash85`, and `Cities`, and $\eta = 0.7$ for `WorldCities` and `gre_216a`. The number of blocks for TREBK and TREGBK is determined by the dimension of the tensor $\mathcal{A}$.

For the different sparse tensor systems, IT, CPU, and RSE for the four methods when the desired accuracy is reached are given in Table 2. An iterative convergence diagram is given in Fig. 7 for the `nos5` system. In the Fig. 7, the TREK method exhausted the total number of iterations, without achieving the required tolerance.

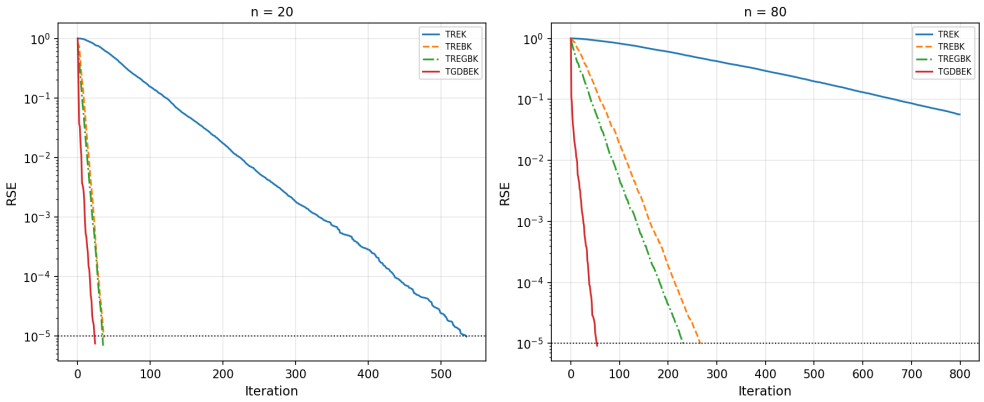


Fig. 3 RSE versus iteration for $n = 20$ (left) and $n = 80$ (right). TGDBEK converges in 23 and 43 iterations respectively; TREK exhausts the 800-iteration budget at $n = 80$. Dotted line: tolerance tol $= 10^{-5}$.

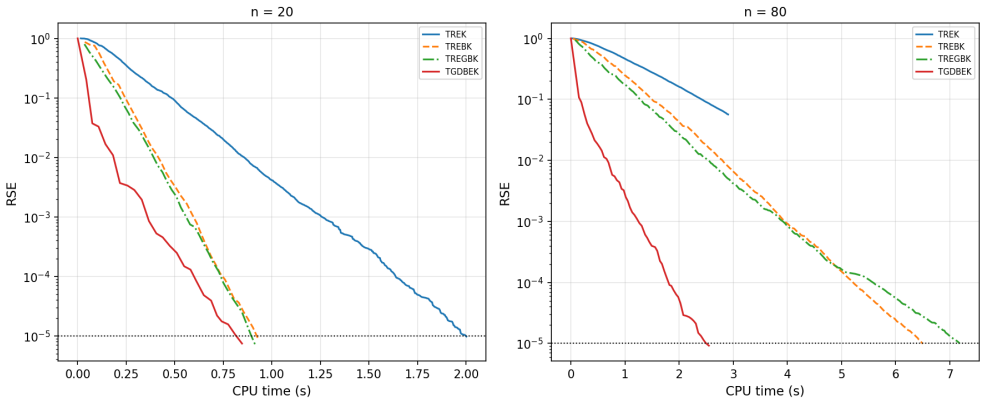
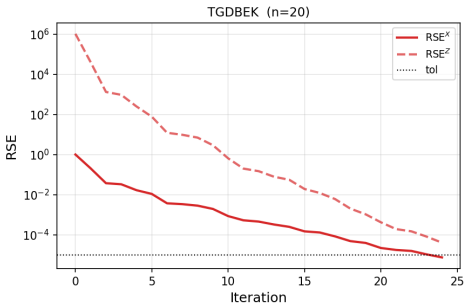
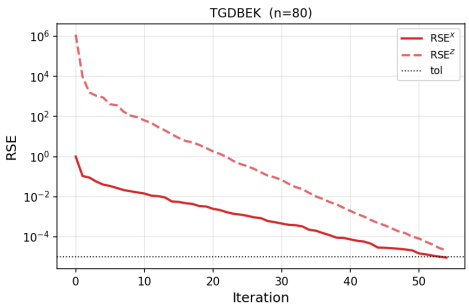


Fig. 4 RSE vs CPU for $n = 20$ (left) and $n = 80$ (right). TGDBEK reaches the tolerance in the least time across both problem sizes; TREK exhausts the budget at $n = 80$ without converging.

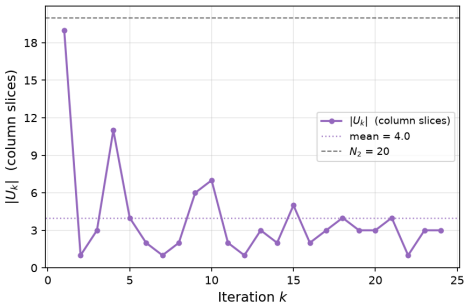


(a) RSE^Z vs. iteration, $n = 20$.

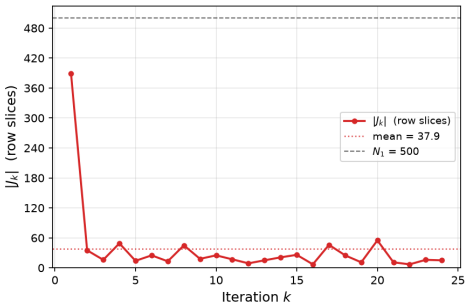


(b) RSE^Z vs. iteration, $n = 80$.

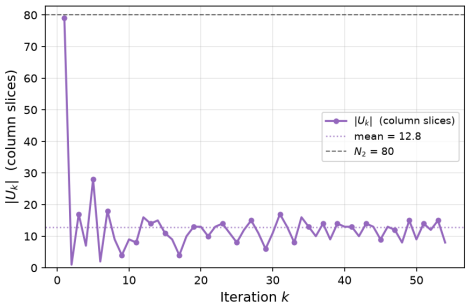
Fig. 5 Z-iterate relative squared error (RSE^Z) versus iteration for $n = 20$ (left) and $n = 80$ (right). TGDBEK drives RSE^Z to zero faster than all competitors; the Z-iterate convergence mirrors the X-iterate pattern. Dotted line: tolerance $\mathrm{tol} = 10^{-5}$.



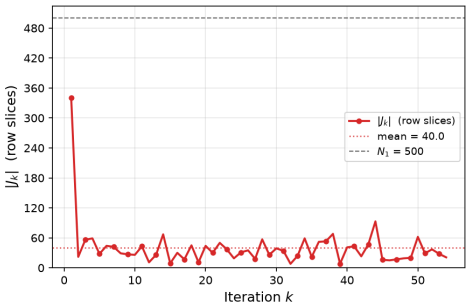
(a) $|U_k|$, $n = 20$.



(b) $|J_k|$, $n = 20$.



(c) $|U_k|$, $n = 80$.



(d) $|J_k|$, $n = 80$.

Fig. 6 Cardinalities of each of the greedy blocks U_k and J_k versus iteration for dense systems $\mathcal{A} \in \mathbb{R}^{500\times n\times 10}$ ($\eta = 0.6$, $a = 10^{-3}$). Column set $|U_k|$ (left) and row set $|J_k|$ (right), for $n = 20$ (top) and $n = 80$ (bottom). Dotted line: mean over the run; dashed line: maximal admissible size ($N_2 = n$ for U_k, $N_1 = 500$ for J_k).

Table 2 IT, CPU, and RSE for sparse tensor systems from SuiteSparse. Each block header gives tensor dimensions $N_1 \times N_2 \times N_3$ and density ρ of the underlying sparse matrix. NaN: method failed to converge in the given iteration budget. Noise: `nos5` uses $a = 10^{-2}$; `ash85`, `Cities`, `gre_216a` $a = 10^{-3}$; `WorldCities` $a = 10^{-1}$. Max iterations: 300 (`nos5`, `ash85`); 6000 (others). Bold: best per matrix.

Matrix	Method	IT	CPU (s)	RSE
`nos5` $468 \times 39 \times 12$, $\rho = 2.36\%$				
	TREK	300	1.249	9.288×10^{-2}
	TREBK	138	4.361	9.888×10^{-7}
	TREGBK	106	3.401	9.559×10^{-7}
	TGDBEK	**44**	**2.377**	$\mathbf{9.659 \times 10^{-7}}$
`ash85` $85 \times 17 \times 5$, $\rho = 7.24\%$				
	TREK	300	1.123	1.386×10^{-1}
	TREBK	171	2.315	9.463×10^{-7}
	TREGBK	300	4.117	1.028×10^{-6}
	TGDBEK	**128**	**2.395**	$\mathbf{9.716 \times 10^{-7}}$
`Cities` $55 \times 23 \times 2$, $\rho = 53.04\%$				
	TREK	6000	20.947	3.807×10^{-3}
	TREBK	1916	13.347	9.815×10^{-7}
	TREGBK	5740	45.566	9.881×10^{-7}
	TGDBEK	**1107**	**15.627**	$\mathbf{9.783 \times 10^{-7}}$
`WorldCities` $315 \times 20 \times 5$, $\rho = 23.87\%$				
	TREK	6000	21.676	NaN
	TREBK	1136	15.599	9.888×10^{-7}
	TREGBK	623	9.274	9.875×10^{-7}
	TGDBEK	**262**	**5.531**	$\mathbf{9.970 \times 10^{-7}}$
`gre_216a` $216 \times 24 \times 9$, $\rho = 1.88\%$				
	TREK	879	3.148	9.893×10^{-7}
	TREBK	100	2.316	8.839×10^{-7}
	TREGBK	72	1.846	9.749×10^{-7}
	TGDBEK	**61**	**2.176**	$\mathbf{9.192 \times 10^{-7}}$

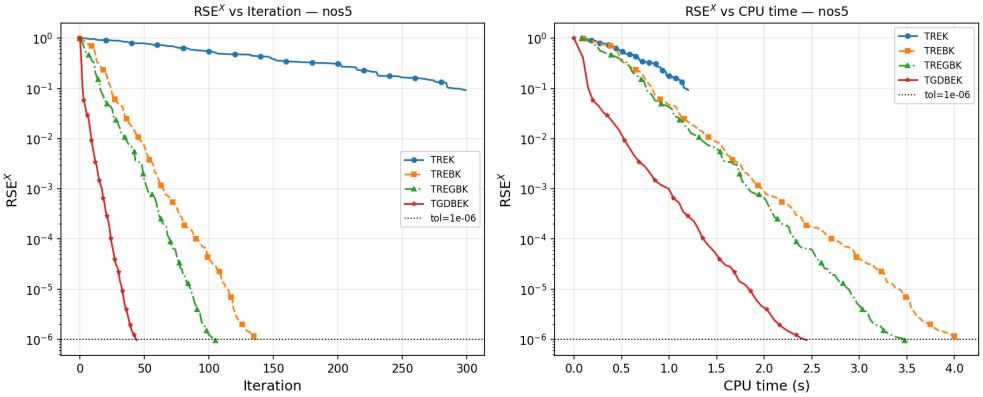


Fig. 7 RSE versus iteration (left) and RSE versus wall-clock CPU time (right) for the `nos5` sparse tensor system ($468 \times 39 \times 12$, $a = 10^{-2}$, $T = 300$). TGDBEK converges in 44 iterations and reaches the tolerance first in CPU time; TREK exhausts the budget. Dotted line: tolerance $\mathrm{tol} = 10^{-6}$.

Example 4.3 (**Color image deblurring**) In this example, we use the TREK, TREBK, TREGBK, and TGDBEK methods to solve the tensor linear system (1) arising from color image restoration. The test image is a public-domain flower photograph, resized and stored as a tensor $\mathcal{X}_* \in \mathbb{R}^{200\times200\times3}$. The deblurring operator $\mathcal{A} \in \mathbb{R}^{200\times200\times3}$ is constructed by setting the first frontal slice $\mathcal{A}_{:,:,1}$ to the normalized Gaussian Toeplitz blur matrix with $\sigma = 4$ and bandwidth band $= 32$, and all remaining frontal slices to zero. In fact, we let

$$z_j = \exp\left(-\frac{j^2}{2\sigma^2}\right), \quad j = 0, \ldots, \text{band} - 1, \qquad \mathcal{A}_{:,:,1} = \frac{1}{\sigma\pi} D^{-1}T(z), \quad D = \text{diag}(T(z)\,\mathbf{1}),$$

where $T(z)$ is the symmetric Toeplitz matrix generated by z, $\mathbf{1}$ is the all-ones vector, and $D^{-1}T(z)$ denotes row normalization, with parameters $\sigma = 4$ and band $= 32$. The data tensor is obtained by $\mathcal{B} = \mathcal{A} * \mathcal{X}_* + \epsilon$, where ϵ is generated as described above with noise level $a = 10^{-2}$, and the termination criterion is the same as Example 4.1. The maximum number of iterations is 1000.

The final experimental results are given in Table 3. The original image, the blurred and noisy image, and the images recovered by each method are shown in Fig. 8. The RSE and RSE^Z versus iteration are shown in Fig. 9.



(a) True image (b) Blurred + noisy (c) TREK

(d) TREBK (e) TREGBK (f) TGDBEK

Fig. 8 Color "flower" image restoration: (a) true image, (b) blurred and noisy, (c)–(f) recovered images by TREK, TREBK, TREGBK, and TGDBEK.

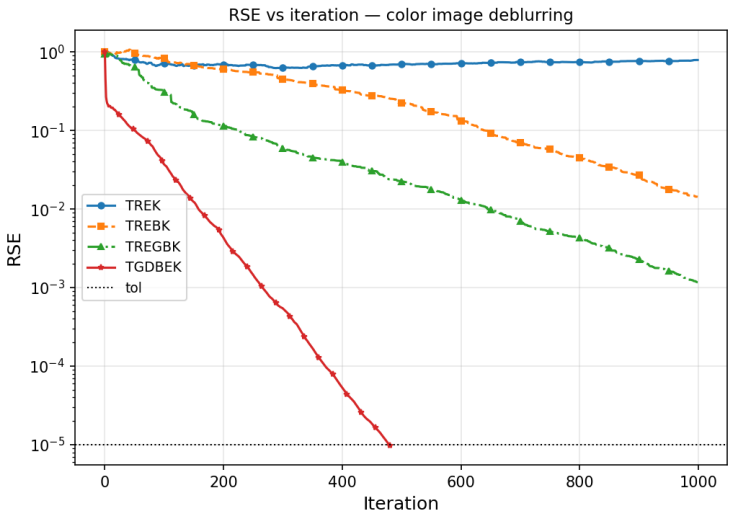


Fig. 9 RSE versus iteration for color image deblurring ($200\times200\times3$, $\sigma = 4$, $a = 10^{-2}$, max 1000 iterations). TGDBEK is the only method to reach tolerance 10^{-5}, converging at iteration 503.

Table 3 CPU, IT, RSEs, for color image deblurring ($200 \times 200 \times 3$, $\sigma = 4$, $a = 10^{-2}$, max 1000 iterations). The last column gives the RSE for the $\mathcal{Z}$ iterate.

Method	CPU (s)	IT	RSE	RSE
TREK	2.088	1000	7.92×10^{-1}	—
TREBK	8.822	1000	2.18×10^{-2}	—
TREGBK	8.216	1000	1.27×10^{-3}	—
TGDBEK	6.006	503	1.00×10^{-5}	8.745×10^{-10}

As can be seen from Table 3, TREK and TREBK exhaust the 1000-iteration budget without reaching the prescribed tolerance, stopping at $\mathrm{RSE}^X = 7.92\times10^{-1}$ and 2.18×10^{-2}, respectively. TREGBK also fails to converge, reaching only $\mathrm{RSE}^X = 1.27\times10^{-3}$. TGDBEK is the only method to reach $\mathrm{RSE}^X < 10^{-5}$, converging at iteration 503 in 6.01, a considerable reduction in wall-clock time compared to TREGBK (8.22 s). Moreover, from Fig. 8, the image recovered by TGDBEK is slightly sharper than those of all competing methods. Fig. 9 further confirms TGDBEK's superior convergence rate across all methods.

Example 4.4 (**Gray image deblurring**) In this example, we apply the TREK, TREBK, TREGBK, and TGDBEK methods to solve the tensor linear system (1) arising from gray MRI-like image deblurring. The test image is a Shepp–Logan phantom [44] resized to $128\times$

128 pixels. A volumetric dataset $\mathcal{X}_* \in \mathbb{R}^{128\times128\times27}$ is constructed by generating 27 frontal slices, each obtained by rotating the base phantom by an angle uniformly spaced in $[-8^\circ, 8^\circ]$ and applying a mild sinusoidal intensity scaling across slices. The deblurring operator $\mathcal{A} \in \mathbb{R}^{128\times128\times27}$ is constructed as follows. We let

$$z_j = \exp\left(-\frac{j^2}{2\sigma^2}\right), \quad j = 0, \ldots, \text{band} - 1, \qquad \mathcal{A}_{:,:,1} = \frac{1}{\sigma\pi} D^{-1}T(z), \quad D = \text{diag}(T(z)\,\mathbf{1}),$$

where $T(z)$ is the symmetric Toeplitz matrix generated by z, $\mathbf{1}$ is the all-ones vector, and $D^{-1}T(z)$ denotes row normalization, with parameters $\sigma = 4$ and band $= 64$. All remaining frontal slices are set to zero. The data tensor is obtained by $\mathcal{B} = \mathcal{A} * \mathcal{X}_* + \epsilon$, where ϵ is generated as described above with noise level $a = 10^{-3}$, and the termination criterion is the same as Example 4.1. The maximum number of iterations is 1000.

The final experimental results are given in Table 4. The blurred image and the images recovered by each method (central frontal slice) are shown in Fig. 10.

Table 4 IT, CPU, PSNR, and SSIM for gray MRI-like image deblurring ($128 \times 128 \times 27$, $\sigma = 4$, $a = 10^{-3}$, max 1000 iterations). PSNR and SSIM are computed on the central frontal slice.

Method	IT	CPU (s)	PSNR (dB)	SSIM
TREK	1000	1.523	18.14	0.602
TREBK	1000	3.041	57.74	0.998
TREGBK	1000	2.654	57.99	0.998
TGDBEK	**1000**	**2.013**	**57.99**	**0.998**

As can be seen from Table 4 and Fig. 10, all the algorithms exhaust the given total number of iterations, with TREK being the first to stop while it didn't achieve the prescribe tolerance. TREK recoves poorly the original image with PSNR = 18.14 dB, SSIM = 0.602. TREBK, TREGBK, and TGDBEK all recover the phantom faithfully, achieving PSNR $\approx$ 58 dB and SSIM $\approx$ 0.998. Notably, TGDBEK matches the reconstruction quality of TREGBK while achieving faster the tolerance in fewer seconds than TREGBK. That demonstrates how the greedy double-block strategy yields computational savings even when all methods exhaust the iteration budget.



(a) True image (b) Blurred + noisy (c) TREK



(d) TREBK (e) TREGBK (f) TGDBEK

Fig. 10 Gray MRI-like image deblurring ($128 \times 128 \times 27$, central frontal slice): (a) true image, (b) blurred and noisy, (c)–(f) recovered images by TREK, TREBK, TREGBK, and TGDBEK. TREK diverges; TREBK, TREGBK, and TGDBEK achieve near-identical high-fidelity recovery.

Example 4.5 (**Effect of the threshold** η) In this example, we examine how the greedy threshold η affects the convergence speed and computational cost of TGDBEK. Recall that $\eta \in (0, 1]$ controls the size of the active-sets: a row block i (resp. column block j) is included in the active set if and only if its squared residual contribution is at least η times the maximum over all blocks. A small η admits many blocks per step (akin to a full block update), whereas $\eta = 1$ retains only the single most informative block at each iteration (akin to TREK).

Here, the test system has coefficient tensor $\mathcal{A} \in \mathbb{R}^{500\times 20\times 10}$ with a randomly generated exact solution $\mathcal{X}_* \in \mathbb{R}^{20\times 10\times 10}$ and noise level $a = 10^{-2}$. The experiment is run for each value $\eta \in \{0.1, 0.2, \ldots, 1.0\}$, with a maximum of 2000 iterations and tolerance 10^{-6}, and results are averaged over 20 independent trials with standard Gaussian entries.

The average iteration count IT and CPU time as functions of η are reported in Table 5 and Fig. 11.

We can see that from Table 5 and Fig. 11, the behavior of TGDBEK is remarkably consistent across both noise levels. For $\eta \leq 0.4$, TGDBEK converges in a single iteration at both noise levels. At $\eta = 0.5$ the IT rises to 6, marking the onset of the transition regime. As η increases beyond 0.5, the active set shrinks and both IT and CPU time grow substantially. At $\eta = 1.0$, the algorithm degenerates to a TREK-like single-block update requiring 362 iterations — approximately 20× more than at $\eta = 0.6$. The recommended range

Table 5 Average IT and CPU of TGDBEK vs. greedy threshold η ($\mathcal{A} \in \mathbb{R}^{500\times20\times10}$, tol $= 10^{-6}$, 20 runs) at two noise levels.

	$a = 10^{-2}$		$a = 10^{-1}$	
η	IT	CPU (s)	IT	CPU (s)
0.1	1	0.031	1	0.031
0.2	1	0.027	1	0.027
0.3	1	0.027	1	0.027
0.4	1	0.029	1	0.029
0.5	6	0.264	6	0.273
0.6	18	0.674	18	0.650
0.7	37	1.273	35	1.184
0.8	57	1.796	58	1.716
0.9	88	2.441	88	2.343
1.0	362	7.971	362	7.898

is $\eta \in [0.1, 0.6]$, where convergence speed and per-iteration cost are jointly minimized, and this recommendation holds regardless of noise level.

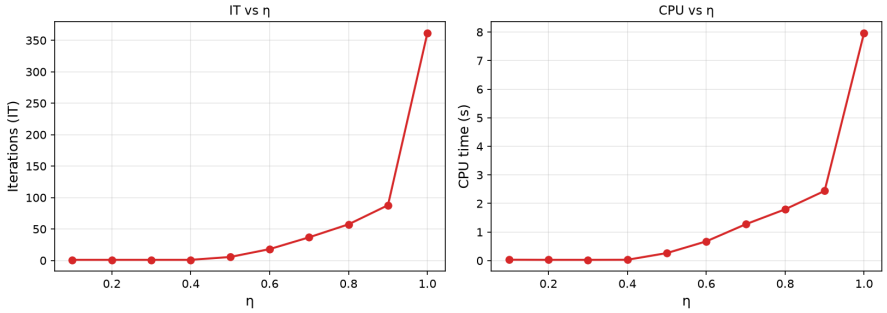
As can be seen from Table 5 and Fig. 11, the sharp transition between $\eta = 0.4$ (IT$= 1$) and $\eta = 0.5$ (IT$= 6$) reflects a qualitative change in block size. The practical recommendation is to keep $\eta \in [0.1, 0.5]$ when iteration count is the priority, or $\eta \approx$ 0.5–0.6 when per-step cost matters.

5 Conclusion

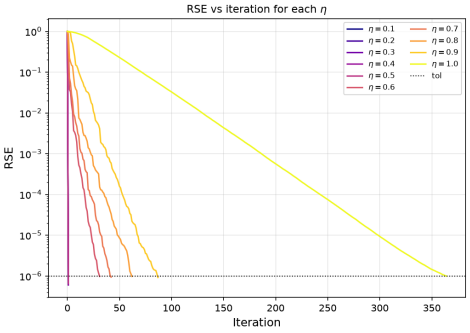
In this paper, we proposed the Tensor Greedy Double Block Extended Kaczmarz (TGDBEK) method for solving large, inconsistent tensor linear systems under the t-product algebraic framework. By dynamically selecting active sets of horizontal and lateral slices with maximal residual energy at each iteration, TGDBEK eliminates the need for predefined, rigid block partitions used in existing methods such as TREBK and TREGBK. We established a theoretical convergence analysis proving that TGDBEK converges linearly in the Frobenius norm to the unique minimum-norm least-squares solution $\mathcal{A}^{\dagger} * \mathcal{B}$.

Extensive numerical experiments across synthetic dense overdetermined tensor systems, real-world sparse tensors from the SuiteSparse collection, multichannel color image deblurring, and 3D volumetric MRI phantom reconstruction demonstrate the superiority of TGDBEK. Across all benchmarks, TGDBEK achieved up to a 5× reduction in iteration count and up to a 2.5× reduction in wall-clock CPU time compared to existing state-of-the-art tensor Kaczmarz solvers.

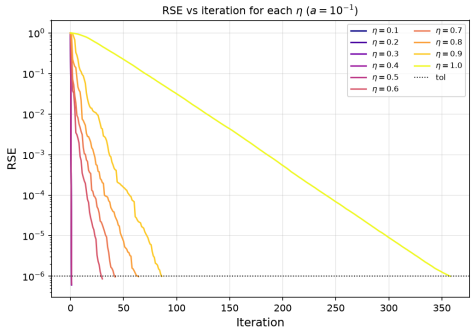
Several promising research directions arise from this work:



(a) Average IT (left) and CPU time (right) vs. η.



(b) RSE vs. iteration, $a = 10^{-2}$.



(c) RSE vs. iteration, $a = 10^{-1}$.

Fig. 11 Effect of the greedy threshold η on TGDBEK. Top: average IT and CPU vs. η ($a = 10^{-2}$, 20 runs). Bottom: RSE vs. iteration for each $\eta \in \{0.1, \ldots, 1.0\}$ at two noise levels ($\mathcal{A} \in \mathbb{R}^{500\times 20\times 10}$).

1. **Projection-Free (Pseudoinverse-Free) Variants by Averaging[12, 33]:** In each step of TGDBEK, computing the Moore–Penrose pseudoinverse $(\mathcal{A}_{J_k,:,:})^\dagger$ of active sub-tensors can become computationally demanding for very large block sizes. As a natural next step, we will develop *projection-free greedy extended block Kaczmarz methods* leveraging an *averaging strategy* (in the spirit of tensor randomized extended average block Kaczmarz). By replacing tensor pseudoinverses with weighted parallel slice updates and averaging, the per-iteration computational complexity can be drastically curtailed while preserving fast convergence.
2. **Adaptive and Dynamic Thresholding:** Investigating dynamic threshold sequences η_k that adaptively shrink or expand active block sizes based on the decay of the residual spectrum.
3. **High-Order Generalization:** Extending the TGDBEK and greedy averaging frameworks to arbitrary p-th order tensors ($p \geq 4$) under the generalized tuple t-product.

Acknowledgements. This research was conducted at the African Institute of Mathematical Sciences (AIMS). The authors would like to thank AIMS for providing research environment and technical supports.

Author contribution. J. Mabiala proposed the idea of this work, implemented, ran the examples, and wrote the manuscript, and L. Tondji supervised the work and contributed the connection with the Bregman–Kaczmarz literature and the accelerated variants. The statement of the first theorem was written by J. Mabiala, and was rewritten, together with the accompanying comments, by L. Tondji. Both authors read and approved the final manuscript.

Funding. None.

Code availability. Yes, it is at.

Declarations

Competing interests. The authors declare no competing interests.

Ethics approval and consent to participate. Not applicable.

Consent for publication. Yes, it is.

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., Braman, K., Hao, N., Hoover, R.: Third-order tensors as operators on matrices: A theoretical and computational framework with applications in imaging. SIAM J. Matrix Anal. Appl. **34**, 148–172 (2013)

[3] Zhang, J., Saibaba, A.K, Kilmer, M.E, Aeron S.: A randomized tensor singular value decomposition based on the t-product. Numer Linear Algebra Appl. 2018; 25:e2179. https://doi.org/10.1002/nla.2179

[4] Wang, X., Che, M., Wei, Y.: Tensor neural network models for tensor singular value decompositions. Comput. Optim. Appl. **75**, 753–777 (2020)

[5] Qi, L., Yu, G.: T-singular values and t-sketching for third order tensors. arXiv:2103.00976 (2021)

[6] Newman, E., Horesh, L., Avron, H., Kilmer, M.: Stable tensor neural networks for rapid deep learning. arXiv:1811.06569 (2018)

[7] 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)

[8] Bai, Z.Z., Wu, W.T.: On greedy randomized coordinate descent methods for solving large linear least-squares problems. Numer. Linear Algebra Appl. **26**(4), e2237 (2019)

[9] Bai, Z.Z., Wu, W.T.: On greedy randomized augmented Kaczmarz method for solving large sparse inconsistent linear systems. SIAM J. Sci. Comput. **43**(6), A3892–A3911 (2021)

[10] Sun, W.N., Qin, M.: On greedy double block extended Kaczmarz algorithm for solving large inconsistent linear systems. J. Math. Anal. Appl. **546** (2025)

[11] Du, K., Si, W.T., Sun, X.H.: Randomized extended average block Kaczmarz for solving least squares. SIAM J. Sci. Comput. **42**(6), A3541–A3559 (2020)

[12] Tondji, L., Lorenz, D.A.: Faster randomized block sparse Kaczmarz by averaging. Numer. Algorithms **93**(4), 1417–1451 (2023)

[13] Kaczmarz, S.: Angenäherte auflösung von systemen linearer gleichungen. Bull. Intern. Acad. Polonaise Sci. Lett. **35**, 355–357 (1937)

[14] Ma, A., Molitor, D.: Randomized Kaczmarz for tensor linear systems. BIT **62**(1), 171–194 (2022)

[15] Chen, X.M., Qin, J.: Regularized Kaczmarz algorithms for tensor recovery. SIAM J. Imaging Sci. **14**(4), 1439–1471 (2021)

[16] Wang, X., Che, M., Wei, Y.: Randomized Kaczmarz methods for tensor complementarity problems. Comput. Optim. Appl. **82**(3), 595–615 (2022)

[17] Du, K., Sun, X.H.: Randomized regularized extended Kaczmarz algorithms for tensor recovery. arXiv preprint arXiv:2112.08566 (2021)

[18] Wang, X., Che, M., Mo, C.X., Wei, Y.: Solving the system of nonsingular tensor equations via randomized Kaczmarz-like method. J. Comput. Appl. Math. **421** (2023)

[19] Liao, Y., Li, W., Yang, D.: The accelerated tensor Kaczmarz algorithm with adaptive parameters for solving tensor systems. Appl. Numer. Math. **202**, 100–119 (2024). https://doi.org/10.1016/j.apnum.2024.04.016

[20] Schöpfer, F., Lorenz, D.A., Tondji, L., Winkler, M.: Extended randomized Kaczmarz method for sparse least squares and impulsive noise problems. Linear Algebra Appl. **652**, 132–154 (2022)

[21] Tondji, L., Tondji, I., Lorenz, D.A.: Adaptive Bregman–Kaczmarz: an approach to solve linear inverse problems with independent noise exactly. Inverse Probl. **40**(9), 095006 (2024)

[22] Tondji, L., Mahamat, A.A., Tondji, I.: Accelerated exact recovery from noisy data via averaging and noise-aware adaptive Bregman–Kaczmarz. arXiv:2607.16003 (2026)

[23] Tondji, L., Necoara, I., Lorenz, D.A.: Acceleration and restart for the randomized Bregman–Kaczmarz method. Linear Algebra Appl. **699**, 508–538 (2024)

[24] Tondji, L.: Advances in Bregman–Kaczmarz methods: accelerations and inconsistency. PhD thesis, Technische Universität Braunschweig (2024)

[25] Tondji, L., Lorenz, D.A., Necoara, I.: An accelerated randomized Bregman–Kaczmarz method for strongly convex linearly constraint optimization. In: 2023 European Control Conference (ECC). IEEE (2023)

[26] Tondji, L.N., Winkler, M., Lorenz, D.A.: Linear convergence of a weighted randomized sparse Kaczmarz method. In: 2021 Online International Conference on Computational Harmonic Analysis (Online-ICCHA) (2021)

[27] Zhang, X., Guo, X., Pan, J.: A sampling greedy average regularized Kaczmarz method for tensor recovery. Numer. Linear Algebra Appl. **2560** (2024). https://doi.org/10.1002/nla.2560

[28] Bao, W.D., Zhang, F.Y., Li, W.G., Wang, Q., Gao, Y.: Randomized average Kaczmarz algorithm for tensor linear systems. Mathematics **10**(23), 4594 (2022)

[29] An, L., Liang, K., Han, J., Liu, Q.: Randomized extended average block Kaczmarz method for inconsistent tensor equations under t-product. Numer. Algorithms **100**, 1123–1144 (2025)

[30] Huang, G.X., Zhong, S.Y.: Tensor randomized extended Kaczmarz methods for large inconsistent tensor linear equations with t-product. Numer. Algorithms, 1–24 (2023)

[31] Miao, Y., Qi, L., Wei, Y.: Generalized tensor function via the tensor singular value decomposition based on the T-product. Linear Algebra Appl. **590**, 258–303 (2020)

[32] Jin, H.W., Bai, M.R., Benítez, J., Liu, X.J.: The generalized inverses of tensors and an application to linear models. Comput. Math. Appl. **74**(3), 385–397 (2017)

[33] Necoara, I.: Faster randomized block Kaczmarz algorithms. SIAM J. Matrix Anal. Appl. **40**(4), 1425–1452 (2019)

[34] Needell, D.: Randomized Kaczmarz solver for noisy linear systems. BIT **50**(2), 395–403 (2010)

[35] Needell, D., Tropp, J.A.: Paved with good intentions: analysis of a randomized block Kaczmarz method. Linear Algebra Appl. **441**, 199–221 (2014)

[36] Needell, D., Zhao, R., Zouzias, A.: Randomized block Kaczmarz method with projection for solving least squares. Linear Algebra Appl. **484**, 322–343 (2015)

[37] Strohmer, T., Vershynin, R.: A randomized Kaczmarz algorithm with exponential convergence. J. Fourier Anal. Appl. **15**(2), 262–278 (2009)

[38] Zouzias, A., Freris, N.M.: Randomized extended Kaczmarz for solving least squares. SIAM J. Matrix Anal. Appl. **34**(2), 773–793 (2013)

[39] Tarzanagh, D.A., Michailidis, G.: Fast randomized algorithms for t-product based tensor operations and decompositions with applications to imaging data. SIAM J. Imaging Sci. **11**(4), 2629–2664 (2018)

[40] Moorman, J.D., Tu, T.K., Molitor, D., Needell, D.: Randomized Kaczmarz with averaging. BIT Numer. Math. **61**, 337–359 (2021)

[41] Castillo, A., Haddock, J., Hartsock, I., et al.: Randomized iterative methods for tensor regression under the t-product. In: Advances in Data Science: Women in Data Science and Mathematics (WiSDM) 2023, pp. 3–38. Springer Nature Switzerland, Cham (2025)

[42] Marshall, N.F., Mickelin, O.: An optimal scheduled learning rate for a randomized Kaczmarz algorithm. SIAM J. Matrix Anal. Appl. **44**(1), 312–330 (2023)

[43] Kolodziej, S.P., et al.: The SuiteSparse Matrix Collection Website Interface. J. Open Source Softw. **4**(33), 1244 (2019)

[44] Shepp, L.A., Logan, B.F.: The Fourier reconstruction of a head section. IEEE Trans. Nucl. Sci. **21**(3), 21–43 (1974)