An Efficient Two-Sided Sketching Method for Large-Scale Tensor Decomposition Based on Transformed Domains[*]

Zhiguang Cheng[†] Gaohang Yu[‡] Xiaohao Cai [§] Liqun Qi [¶]

September 26, 2024

Abstract

Large tensors are frequently encountered in various fields such as computer vision, scientific simulations, sensor networks, and data mining. However, these tensors are often too large for convenient processing, transfer, or storage. Fortunately, they typically exhibit a low-rank structure that can be leveraged through tensor decomposition. However, performing large-scale tensor decomposition can be time-consuming. Sketching is a useful technique to reduce the dimensionality of the data. In this paper, we propose a novel two-sided sketching method based on the $\star_L$-product decomposition and transformed domains like the discrete cosine transformation. A rigorous theoretical analysis is also conducted to assess the approximation error of the proposed method. Specifically, we improve our method with power iteration to achieve more precise approximate solutions. Extensive numerical experiments and comparisons on low-rank approximation of synthetic large tensors and real-world data like color images and grayscale videos illustrate the efficiency and effectiveness of the proposed approach in terms of both CPU time and approximation accuracy.

Key words. Large-scale tensor, tensor decomposition, sketching, t-product, discrete cosine transformation, power iteration.

MSC codes. 68Q25, 68R10, 68U05

[*]Submitted to the editors DATE.

Funding:This work was supported in part by National Natural Science Foundation of China (No. 12071104) and Natural Science Foundation of Zhejiang Province. (No. LD19A010002).

[†]Department of Mathematics, Hangzhou Dianzi University, Hangzhou, 310018, China; Department of Mathematics, Quzhou University, Quzhou, 324000, China; (32086@qzc.edu.cn).

[‡]Department of Mathematics, Hangzhou Dianzi University, Hangzhou, 310018, China; (maghyu@163.com).

[§]School of Electronics and Computer Science, University of Southampton, Southampton, UK; (x.cai@soton.ac.uk).

[¶]Department of Applied Mathematics, The Hong Kong Polytechnic University, Kowloon, Hong Kong; (liqun.qi@polyu.edu.hk).

1 Introduction

Multidimensional arrays, known as tensors, are often used to represent real-world high-dimensional data, such as videos [1, 2], hyperspectral images [3, 4, 5, 6], multilinear signals [7, 8], and communication networks [9, 10]. In most cases, these tensor data usually have a low-rank structure and can be approximated by tensor decomposition. Nevertheless, computing the tensor decomposition of these large-scale data is usually computationally demanding, and thus finding an accurate approximation of large-scale data with great efficiency plays a key role in tensor data analysis. Sketching is a useful technique for data compression, utilizing random projections or sampling to approximate the original data. Although the sketching technique may slightly reduce the accuracy of the approximation, it can significantly reduce the computational and storage complexity [11]. As a result, sketching is commonly used in low-rank tensor approximation, and many researchers have proposed various tensor sketching algorithms [12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24]. They have also been successfully applied to a variety of tasks, such as Kronecker product regression, polynomial approximation, the construction of deep convolutional neural networks [25, 26, 27, 28, 29, 30], etc.

Recently, extensive research has been carried out on the application of sketching algorithms for the low-rank matrix approximation [31, 32, 33, 34]. Woodruff et al. [32] examined the numerical linear algebra algorithms of linear sketching techniques and identified their limitations. Tropp et al. [33] developed a two-sided matrix sketching algorithm, which can maintain the structural properties of the input matrix and generate a low-rank matrix approximation with a given rank. Furthermore, Tropp et al. [34] proposed a new matrix sketching algorithm to construct a low-rank approximation matrix from streaming data. These matrix sketching algorithms are very effective in reducing storage and computational costs when computing low-rank approximations of large-scale matrices. Subsequently, many researchers have applied matrix sketching algorithms to tensor decomposition and developed various low-rank tensor approximation algorithms. The followings are some low-rank tensor approximation algorithms based on different decompositions [12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24].

Li et al. [13] introduced a random algorithm for CANDECOMP/PARAFAC (CP) tensor decomposition in least-squares regression, aiming to achieve reduced dimensionality and sparsity in randomized linear mapping. Wang et al. [14] developed innovative techniques for performing randomized tensor contractions using the fast Fourier transform (FFT), avoiding explicit formation of tensors. The robust tensor power method based on the tensor sketch (TS-RTPM) can quickly explore the potential features of the tensor, but in some cases its approximation performance is limited. Cao et al. [12] proposed a data-driven framework called TS-RTPM-Net, which improves the accuracy of the estimation of TS-RTPM by jointly training the TS value matrices with the initial

RTPM.

Numerous studies have also been conducted based on Tucker decomposition [15, 16, 17, 18, 19, 20, 23, 24]. For example, Che et al. [15] devised an adaptive randomized approach to approximate Tucker decomposition. Malik et al. [18] proposed two randomized algorithms for low-rank Tucker decomposition, which entail a single pass of the input tensor by integrating sketching. Ravishankar et al. [19] introduced the hybrid Tucker TensorSketch vector quantization (HTTSVQ) algorithm for dynamic light fields. Sun et al. [20] developed a randomized method for Tucker decomposition, which can provide a satisfactory approximation without a second pass on the original tensor data. Minster et al. [23] devised randomized adaptations of the THOSVD and STHOSVD algorithms. Dong et al. [24] presented two practical randomized algorithms for low-rank Tucker approximation of large tensors based on sketching and power scheme, with a rigorous error-bound analysis.

The work based on tensor-train (TT) decomposition can be found in [15, 21, 35, 36]. In particular, Che et al. [15] designed an adaptive random algorithm to calculate the tensor column approximation. Hur et al. [21] introduced a sketching algorithm to construct a TT representation of a probability density from its samples, which can avoid the curse of dimensionality and sample complexities of the recovery problem. Qi and Yu [22] proposed a tensor sketching method based on the t-product, which can quickly obtain a low tubal rank tensor approximation. As pointed out by Kernfeld et al. [37], the t-product has a disadvantage in that, for real tensors, implementation of the t-product and factorizations using the t-product require intermediate complex arithmetic, which, even taking advantage of complex symmetry in the Fourier domain, is more expensive than real arithmetic.

In this paper, based on the $\star_L$-product decomposition [37], we investigate two-sided sketching algorithms for low tubal rank tensor approximation. The main contributions of this paper are as follows. Firstly, we propose a new two-sided sketching method based on transformed domains, which can significantly improve the computational efficiency of the T-Sketch and rt-SVD methods, for low tubal rank approximation. Secondly, we establish a low tubal rank tensor approximation model based on the $\star_L$-product factorization, extending the two-sided matrix sketching algorithm proposed by Tropp et al. [34] with subspace power iteration. Thirdly, a rigorous theoretical analysis is conducted to evaluate the approximation error of the proposed two-sided sketching method. Finally, extensive numerical experiments and comparisons on low-rank approximation of large tensors in different modelities (e.g. synthetic large tensors, color images, and grayscale videos) illustrate the efficiency and effectiveness of the proposed approach in terms of both CPU time and approximation accuracy.

The rest of this paper is organized as follows. Section 2 introduces some common notations and preliminary. Our two-sided sketching algorithms (i.e., Algorithms 1 and 2) based on transformed domains are proposed in Section 3. Section 4 provides strict theoretical guarantees for the approximation error of the proposed algorithms. Section 5 presents detailed numerical experiments and comparisons, demonstrating the efficiency and effectiveness of the proposed algorithms. We conclude in Section 6. Appendix A presents three matrix sketching techniques served as preliminary of the introduction of tensor sketching operators in Section 2.2. Further detailed theoretical guarantees for our proposed algorithms are provided in Appendix B.

2 Notation and Preliminary

In this paper, matrices and tensors are represented by capital letters (e.g. $A, B, \dots$) and curly letters (e.g. $\mathcal{A}, \mathcal{B}, \dots$), respectively. The Matlab command A' can be used to represent the conjugate transpose of the matrix A. $\mathbb{R}$ and $\mathbb{C}$ represent the real number space and the complex number space, respectively. For matrix $A \in \mathbb{C}^{n_1 \times n_2}$, its (i, j)-th element is represented by $a_{i,j}$. For the third-order tensor $\mathcal{A} \in \mathbb{C}^{n_1 \times n_2 \times n_3}$, its (i, j, k)-th element is represented by $a_{i,j,k}$. The Matlab notations $\mathcal{A}(i, :, :)$, $\mathcal{A}(:, i, :)$ and $\mathcal{A}(:, :, i)$ are used to represent the i-th horizontal, lateral and frontal slices of $\mathcal{A}$, respectively. The facial slice $\mathcal{A}(:, :, i)$ is also represented by $\mathcal{A}^{(i)}$. The Frobenius norm of a tensor $\mathcal{A}$ is defined as the square root of the sum of the squares of its elements, i.e.,

$$\|\mathcal{A}\|_F := \|\mathcal{A}(:)\|_2 = \sqrt{\langle \mathcal{A}, \mathcal{A} \rangle} = \sqrt{\sum_{ijk} |a_{ijk}|^2} \, . \tag{2.1}$$

$\mathcal{A}^H$ and $\mathcal{A}^\dagger$ represent the conjugate transpose and pseudo-inverse of $\mathcal{A}$, respectively.

This paper focuses on the low tubal rank tensor approximation that meets the desired accuracy in an efficient manner. For tensor $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, the mathematical model for finding the low-rank approximation $\hat{\mathcal{A}}$ of $\mathcal{A}$ can be expressed as

$$\|\mathcal{A} - \hat{\mathcal{A}}\|_F = \min_{\operatorname{rank}_L(\mathcal{B}) \le l} \|\mathcal{A} - \mathcal{B}\|_F, \tag{2.2}$$

where $l \ll \min\{n_1, n_2\}$ is the target rank, L represents an arbitrary invertible linear transform, and $\operatorname{rank}_L(\mathcal{B})$ denotes the transformed tubal rank of tensor $\mathcal{B}$.

2.1 Transformed Tensor SVD

We below briefly recall the transformed tensor SVD of third-order tensors; more details can be found in [37].

For any third-order tensor $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, let $\bar{\mathcal{A}}_L$ represent a third-order tensor obtained via being multiplied by L (an arbitrary invertible linear transform) on all tubes along the third-dimension of $\mathcal{A}$, i.e.,

$$\bar{\mathcal{A}}_L(i,j,:) = L(\mathcal{A}(i,j,:)),\ i = 1,\dots,n_1,\ j = 1,\dots,n_2. \tag{2.3}$$

Here we write $\bar{\mathcal{A}}_L = L[\mathcal{A}]$. Moreover, one can get $\mathcal{A}$ from $\bar{\mathcal{A}}_L$ by using L^{-1} along the third-dimension of $\bar{\mathcal{A}}_L$, i.e., $\mathcal{A} = L^{-1}[\bar{\mathcal{A}}_L]$. The $\star_L$-product is defined in Definition 2.1 below.

Definition 2.1 *[37] For any two tensors $\mathcal{X} \in \mathbb{C}^{n_1\times n_2\times n_3}$ and $\mathcal{Y} \in \mathbb{C}^{n_2\times n_4\times n_3}$, and an arbitrary invertible linear transform L, the $\star_L$-product of $\mathcal{X}$ and $\mathcal{Y}$ is a tensor $\mathcal{Z} \in \mathbb{C}^{n_1\times n_4\times n_3}$ given by*

$$\mathcal{Z} = \mathcal{X} \star_L \mathcal{Y} = L^H[fold(block(\bar{\mathcal{X}}_L)block(\bar{\mathcal{Y}}_L))], \tag{2.4}$$

where $fold(block(\bar{\mathcal{X}}_L)) = \bar{\mathcal{X}}_L$ and $\bar{X}_L = block(\bar{\mathcal{X}}_L) = \begin{pmatrix} \bar{\mathcal{X}}_L^{(1)} & & & \\ & \bar{\mathcal{X}}_L^{(2)} & & \\ & & \ddots & \\ & & & \bar{\mathcal{X}}_L^{(n_3)} \end{pmatrix}$.

Definition 2.2 *The Kronecker product of matrices $A \in \mathbb{C}^{m\times l}$ and $B \in \mathbb{C}^{p\times r}$ is*

$$A \otimes B = \begin{pmatrix} a_{11}B & a_{12}B & \dots & a_{1l}B \\ a_{21}B & a_{22}B & \dots & a_{2l}B \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1}B & a_{m2}B & \dots & a_{ml}B \end{pmatrix}.$$

The t-product [38] of $\mathcal{A} \in \mathbb{C}^{n_1\times n_2\times n_3}$ and $\mathcal{B} \in \mathbb{C}^{n_2\times n_4\times n_3}$ is a tensor $\mathcal{C} \in \mathbb{C}^{n_1\times n_4\times n_3}$ given by

$$\mathcal{C} = \mathcal{A} * \mathcal{B} = \text{fold}_{\text{vec}}(\text{circ}(\mathcal{A}) \times \text{vec}(\mathcal{B})), \tag{2.5}$$

where

$$\text{vec}(\mathcal{B}) = \begin{pmatrix} \mathcal{B}^{(1)} \\ \mathcal{B}^{(2)} \\ \vdots \\ \mathcal{B}^{(n_3)} \end{pmatrix} \in \mathbb{R}^{n_2 n_3 \times n_4}, \quad \text{fold}_{\text{vec}}(\text{vec}(\mathcal{B})) = \mathcal{B},$$

$$\text{circ}(\mathcal{A}) = \begin{pmatrix} \mathcal{A}^{(1)} & \mathcal{A}^{(n_3)} & \mathcal{A}^{(n_3-1)} & \dots & \mathcal{A}^{(2)} \\ \mathcal{A}^{(2)} & \mathcal{A}^{(1)} & \mathcal{A}^{(n_3)} & \dots & \mathcal{A}^{(3)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathcal{A}^{(n_3)} & \mathcal{A}^{(n_3-1)} & \mathcal{A}^{(n_3-2)} & \dots & \mathcal{A}^{(1)} \end{pmatrix}.$$

Let F_{n_3} and C represent the discrete Fourier transform (DFT) matrix and the discrete cosine transform (DCT) matrix, respectively. The t-product in Eq. (2.5) can be seen as a special case of Definition 2.1. Recall that the block circulant matrix $\text{circ}(\mathcal{A})$ can be diagonalized by the fast Fourier transform matrix F_{n_3}, and the block diagonal matrices are the frontal slices of $\bar{\mathcal{A}}_{F_{n_3}}$, i.e.,

$$\bar{A}_{F_{n_3}} = \text{block}(\bar{\mathcal{A}}_{F_{n_3}}) = (F_{n_3} \otimes I_{n_1}) \times \text{circ}(\mathcal{A}) \times (F_{n_3}^H \otimes I_{n_2}). \tag{2.6}$$

It follows that

$$\begin{aligned}
\mathcal{A} * \mathcal{B} &= \text{fold}_{\text{vec}}(\text{circ}(\mathcal{A}) \times \text{vec}(\mathcal{B})) \\
&= \text{fold}_{\text{vec}}((F_{n_3}^H \otimes I_{n_1}) \times \text{block}(\bar{\mathcal{A}}_{F_{n_3}}) \times (F_{n_3} \otimes I_{n_2}) \times \text{vec}(\mathcal{B})) \\
&= \text{fold}((F_{n_3}^H \otimes I_{n_1}) \times \text{block}(\bar{\mathcal{A}}_{F_{n_3}}) \times \text{block}(\bar{\mathcal{B}}_{F_{n_3}})) \\
&= F_{n_3}^H[\text{fold}(\text{block}(\bar{\mathcal{A}}_{F_{n_3}})\text{block}(\bar{\mathcal{B}}_{F_{n_3}}))] \\
&= F_{n_3}^H[\text{fold}(\bar{A}_{F_{n_3}} \bar{B}_{F_{n_3}})] \\
&= \mathcal{A} \star_{F_{n_3}} \mathcal{B}.
\end{aligned}$$

The definitions of the conjugate transpose of tensor, the identity tensor, the unitary tensor, the invertible tensor, and the diagonal tensor related to the $\star_L$-product are given as follows.

- [39] The conjugate transpose of $\mathcal{A} \in \mathbb{C}^{n_1 \times n_2 \times n_3}$ with respect to L is the tensor $\mathcal{A}^H \in \mathbb{C}^{n_1 \times n_2 \times n_3}$ obtained by $\mathcal{A}^H = L^H[\text{fold}(\text{block}(\bar{\mathcal{A}}_L)^H)] = L^H[\text{fold}(\bar{A}_L^H)]$.
- [37] The identity tensor $\mathcal{I}_L \in \mathbb{C}^{n \times n \times n_3}$ (with respect to L) is defined to be a tensor such that $\mathcal{I}_L = L^H[\mathcal{I}]$, where each frontal slice of $\mathcal{I}$ is the $n \times n$ identity matrix.
- [37] A tensor $\mathcal{Q} \in \mathbb{C}^{n \times n \times n_3}$ is unitary with respect to $\star_L$-product if it satisfies $\mathcal{Q}^H \star_L \mathcal{Q} = \mathcal{Q} \star_L \mathcal{Q}^H = \mathcal{I}_L$, where $\mathcal{I}_L$ is the identity tensor.
- [37] For tensors $\mathcal{A} \in \mathbb{C}^{n \times n \times n_3}$ and $\mathcal{B} \in \mathbb{C}^{n \times n \times n_3}$, if $\mathcal{A} \star_L \mathcal{B} = \mathcal{B} \star_L \mathcal{A} = \mathcal{I}_L$, then tensor $\mathcal{B}$ is the invertible tensor under the $\star_L$-product of tensor $\mathcal{A}$.
- [38] A tensor is a diagonal tensor if each frontal slice of the tensor is a diagonal matrix. For a third-order tensor, if all of its frontal slices are upper or lower triangles, then the tensor is called f-upper or f-lower.

Lemma 2.3 *[37, 38] Suppose that $\mathcal{A} \in \mathbb{R}^{m \times k \times p}$ and $\mathcal{B} \in \mathbb{R}^{k \times n \times p}$ are two arbitrary tensors. Let $\mathcal{Z} = \mathcal{A} \star_L \mathcal{B}$. Then the following properties hold:*

(1) $\|\mathcal{A}\|_F^2 = \frac{1}{p}\|\bar{A}_{F_{n_3}}\|_F^2 = \frac{1}{p}\|\bar{A}_L\|_F^2 = \frac{1}{p}\sum_{i=1}^{p}\|\bar{\mathcal{A}}_L^{(i)}\|_F^2$.

(2) $\mathcal{Z} = \mathcal{A} \star_L \mathcal{B}$ *is equivalent to* $block(\bar{\mathcal{Z}}_L) = block(\bar{\mathcal{A}}_L) block(\bar{\mathcal{B}}_L)$, *i.e.,* $\bar{Z}_L = \bar{A}_L \bar{B}_L$.

Definition 2.4 *(Gaussian Random Tensor) [40] A tensor* $\mathcal{G} \in \mathbb{R}^{m\times n\times p}$ *is called a Gaussian random tensor if the elements of* $\mathcal{G}^{(1)}$ *satisfy the standard normal distribution (i.e., Gaussian with mean zero and variance one) and the other frontal slices are all zero.*

Based on the above definitions, we have the following tensor SVD with respect to L.

Theorem 2.5 *[37]* $\forall \mathcal{A} \in \mathbb{C}^{m\times n\times p}$, *the transformed tensor SVD is given by*

$$\mathcal{A} = \mathcal{U} \star_L \mathcal{S} \star_L \mathcal{V}^H, \tag{2.7}$$

where $\mathcal{U} \in \mathbb{C}^{m\times m\times p}$ *and* $\mathcal{V} \in \mathbb{C}^{n\times n\times p}$ *are unitary tensors with respect to the* $\star_L$*-product, and* $\mathcal{S} \in \mathbb{C}^{m\times n\times p}$ *is a diagonal tensor.*

The transformed tubal rank, denoted as $\text{rank}_L(\mathcal{A})$, is defined as the number of nonzero singular tubes of $\mathcal{S}$, i.e.,

$$\text{rank}_L(\mathcal{A}) = \#\{i : \mathcal{S}(i,i,:) \neq 0\}, \tag{2.8}$$

where # denotes the cardinality of a set. The transformed tensor SVD could be implemented efficiently by the SVDs of the frontal slices in the transformed domain. We also refer the readers to [39] for more details on the computation of the transformed tensor SVD. For the Kernfeld-Kilmer transformed tensor SVD (i.e., Eq. (2.7)), by [37, 38, 41], we have

$$\sum_{k=1}^{p} \mathcal{S}(1,1,k)^2 \geq \sum_{k=1}^{p} \mathcal{S}(2,2,k)^2 \geq \cdots \geq \sum_{k=1}^{p} \mathcal{S}(\min\{m,n\}, \min\{m,n\}, k)^2. \tag{2.9}$$

Definition 2.6 *(Transformed Tensor Singular Values) Suppose* $\mathcal{A} \in \mathbb{R}^{m\times n\times p}$ *with a Kernfeld-Kilmer transformed tensor SVD such that Eq. (2.9) is satisfied. The* i*-th largest transformed tensor singular value of* $\mathcal{A}$ *is defined as*

$$\sigma_i = \sqrt{\sum_{k=1}^{p} \mathcal{S}(i,i,k)^2}, \quad \text{for } i = 1, 2, \ldots, \min\{m,n\}. \tag{2.10}$$

Similarly to the definition of the matrix tail energy, the tail energy of tensor is defined below.

Definition 2.7 *(Tail Energy) For tensor $\mathcal{A} \in \mathbb{R}^{m\times n\times p}$, the i-th largest transformed tensor singular value is σ_i, $i = 1, \dots, \min\{m, n\}$. Then the j-th tail energy of $\mathcal{A}$ is defined as*

$$\tau_j{}^2(\mathcal{A}) := \min_{\mathrm{rank}_L(\mathcal{B})<j} \|\mathcal{A} - \mathcal{B}\|_F^2 = \sum_{i\geq j} \sigma_i^2(\mathcal{A}). \tag{2.11}$$

According to the above definition and using Lemma 2.3 and the linearity, we can obtain the following proposition.

Proposition 2.8 *Suppose $\mathcal{A} \in \mathbb{R}^{m\times n\times p}$ and $\bar{\mathcal{A}}_L$ represents a third-order tensor obtained via being multiplied by L on all tubes along the third-dimension of $\mathcal{A}$. Let j be a positive integer satisfying $j \leq \min\{m, n\}$. Then*

$$\tau_j{}^2(\mathcal{A}) = \frac{1}{p} \sum_{i=1}^{p} \tau_j{}^2(\bar{\mathcal{A}}_L^{(i)}). \tag{2.12}$$

2.2 Tensor Sketching Operator

Using the three matrix sketching techniques (i.e., Gaussian projection, subsampled randomized Hadamard transform (SRHT), and count sketch [43]) with their pseudocodes (i.e., `GaussianProjection`, `SRHT`, and `CountSketch`) introduced in Appendix A, three corresponding tensor sketching operators can be generated. As for an efficient two-sided sketching algorithm, we need to generate four random linear dimension reduction maps, i.e.,

$$\Upsilon \in \mathbb{R}^{k\times m\times p}, \quad \Omega \in \mathbb{R}^{k\times n\times p}, \quad \Phi \in \mathbb{R}^{s\times m\times p}, \quad \text{and } \Psi \in \mathbb{R}^{s\times n\times p}. \tag{2.13}$$

Different ways of generating tensor sketch operators regarding $\mathcal{A} \in \mathbb{R}^{m\times n\times p}$ are shown below.

- **Gaussian tensor sketching operator.** Set
  $\Upsilon = \mathrm{zeros}(k, m, p)$; $\Upsilon(:, :, 1) = \texttt{GaussianProjection}(\mathcal{A}(:, :, 1), k)'$;
  $\Omega = \mathrm{zeros}(k, n, p)$; $\Omega(:, :, 1) = \texttt{GaussianProjection}(\mathcal{A}(:, :, 1)', k)'$;
  $\Phi = \mathrm{zeros}(s, m, p)$; $\Phi(:, :, 1) = \texttt{GaussianProjection}(\mathcal{A}(:, :, 1), s)'$;
  $\Psi = \mathrm{zeros}(s, n, p)$; $\Psi(:, :, 1) = \texttt{GaussianProjection}(\mathcal{A}(:, :, 1)', s)'$.
  Then Υ, Ω, Φ and Ψ are said to be the Gaussian tensor sketching operators.

- **SRHT tensor sketching operator.** Set
  $\Upsilon = \mathrm{zeros}(k, m, p)$; $\Upsilon(:, :, 1) = \texttt{SRHT}(\mathcal{A}(:, :, 1), k)'$;
  $\Omega = \mathrm{zeros}(k, n, p)$; $\Omega(:, :, 1) = \texttt{SRHT}(\mathcal{A}(:, :, 1)', k)'$;
  $\Phi = \mathrm{zeros}(s, m, p)$; $\Phi(:, :, 1) = \texttt{SRHT}(\mathcal{A}(:, :, 1), s)'$;
  $\Psi = \mathrm{zeros}(s, n, p)$; $\Psi(:, :, 1) = \texttt{SRHT}(\mathcal{A}(:, :, 1)', s)'$.
  Then Υ, Ω, Φ and Ψ are said to be the SRHT tensor sketching operators.

- **Count tensor sketching operator.** For $i = 1, 2, \ldots, p$, set
  $\Upsilon = \text{zeros}(k, m, p)$; $\Upsilon(:, :, i) = \texttt{CountSketch}(\mathcal{A}(:, :, 1), k)'$;
  $\Omega = \text{zeros}(k, n, p)$; $\Omega(:, :, i) = \texttt{CountSketch}(\mathcal{A}(:, :, 1)', k)'$;
  $\Phi = \text{zeros}(s, m, p)$; $\Phi(:, :, i) = \texttt{CountSketch}(\mathcal{A}(:, :, 1), s)'$;
  $\Psi = \text{zeros}(s, n, p)$; $\Psi(:, :, i) = \texttt{CountSketch}(\mathcal{A}(:, :, 1)', s)'$.
  Then Υ, Ω, Φ and Ψ are said to be the count tensor sketching operators.

3 The Proposed Two-Sided Sketching Algorithms

The two-sided tensor sketching algorithm proposed by Qi and Yu [22] only considers the range and co-range of the input tensor. On this basis, we here also consider the core sketch. The core sketch contains new information that improves our estimates of the transformed tensor singular values and the transformed tensor singular vectors of the input tensor, and is responsible for the superior performance of the algorithms. We below firstly present the framework of our efficient two-sided sketching method based on the transformed domains for low tubal rank tensor approximation, followed by the principle interpreting its rationale and its extension by using the power iteration technique.

3.1 Method

Given the input tensor $\mathcal{A} \in \mathbb{R}^{m\times n\times p}$ and the objective tubal rank k, using the appropriate tensor sketching operators Υ, Ω, Φ and Ψ in Eq. (2.13), we can realize the randomized sketches $(\mathcal{X}, \mathcal{Y}, \mathcal{Z})$ such as

$$\mathcal{X} := \Upsilon \star_L \mathcal{A} \in \mathbb{R}^{k\times n\times p}, \tag{3.14}$$

$$\mathcal{Y} := \mathcal{A} \star_L \Omega^H \in \mathbb{R}^{m\times k\times p}, \tag{3.15}$$

$$\mathcal{Z} := \Phi \star_L \mathcal{A} \star_L \Psi^H \in \mathbb{R}^{s\times s\times p}. \tag{3.16}$$

The first two tensor sketches $\mathcal{X}$ and $\mathcal{Y}$ respectively capture the co-range and range of $\mathcal{A}$, and the core sketch $\mathcal{Z}$, as we mentioned before, contains new information that improves our estimates of the transformed tensor singular values and vectors of $\mathcal{A}$ and is also responsible for further method performance enhancement.

Once the sketches $(\mathcal{X}, \mathcal{Y}, \mathcal{Z})$ of the input tensor $\mathcal{A}$ are obtained by Eq. (3.14)–(3.16), we can find the low-rank approximation $\hat{\mathcal{A}}$ by following the below proposed three-step process.

(I) Form an L-orthogonal-triangular factorization

$$\mathcal{X}^H := \mathcal{P} \star_L \mathcal{R}_1, \quad \mathcal{Y} := \mathcal{Q} \star_L \mathcal{R}_2, \tag{3.17}$$

where $\mathcal{P} \in \mathbb{R}^{n\times k\times p}$ and $\mathcal{Q} \in \mathbb{R}^{m\times k\times p}$ are partially orthogonal tensors, and $\mathcal{R}_1 \in \mathbb{R}^{k\times k\times p}$ and $\mathcal{R}_2 \in \mathbb{R}^{k\times k\times p}$ are f-upper triangular tensors, in the sense of the $\star_L$-product operation.

(II) Using the known $\mathcal{P}, \mathcal{Q}, \Phi, \Psi$, and $\mathcal{Z}$, calculate

$$\mathcal{C} := (\Phi \star_L \mathcal{Q})^\dagger \star_L \mathcal{Z} \star_L ((\Psi \star_L \mathcal{P})^\dagger)^H \in \mathbb{R}^{k\times k\times p} . \tag{3.18}$$

The above formula of calculating $\mathcal{C}$ is equivalent to solving the below least-squares problem based on the $\star_L$-product, i.e.,

$$\min_{\mathcal{C}} \frac{1}{2} \|\mathcal{G} \star_L \mathcal{C} \star_L \mathcal{M}^H - \mathcal{Z}\|_F^2 , \tag{3.19}$$

where $\mathcal{G} = \Phi \star_L \mathcal{Q}$ and $\mathcal{M} = \Psi \star_L \mathcal{P}$. According to Lemma 2.3, the above problem (3.19) can be reformulated as

$$\min_{\bar{\mathcal{C}}_L} \frac{1}{2p} \|\bar{\mathcal{G}}_L \bar{\mathcal{C}}_L \bar{\mathcal{M}}_L^H - \bar{\mathcal{Z}}_L\|_F^2 , \tag{3.20}$$

whose solution is $\bar{\mathcal{C}}_L^{(i)} = (\bar{\mathcal{G}}_L^{(i)})^\dagger \bar{\mathcal{Z}}_L^{(i)} ((\bar{\mathcal{M}}_L^{(i)})^\dagger)^H$ for $i = 1, \dots, p$.

(III) Construct the transformed tensor tubal rank k approximation

$$\hat{\mathcal{A}} := \mathcal{Q} \star_L \mathcal{C} \star_L \mathcal{P}^H . \tag{3.21}$$

Algorithm 1 L Transformed Randomized Projection Sketching Algorithm

```
1:  Input: Input tensor $\mathcal{A} \in \mathbb{R}^{m\times n\times p}$ and sketch size parameters $k, s$ with $k \le s$.
2:  function L-TRP-SKETCH($\mathcal{A}, k$)
3:  Select the appropriate tensor sketching operators from Eq. (2.13), i.e.,
    $\Upsilon \in \mathbb{R}^{k\times m\times p}, \Omega \in \mathbb{R}^{k\times n\times p}, \Phi \in \mathbb{R}^{s\times m\times p}, \Psi \in \mathbb{R}^{s\times n\times p}$;
4:  $\bar{\mathcal{A}}_L = L[\mathcal{A}]$, $\bar{\Upsilon}_L = L[\Upsilon]$, $\bar{\Omega}_L = L[\Omega]$, $\bar{\Phi}_L = L[\Phi]$, $\bar{\Psi}_L = L[\Psi]$;
5:  for $i \leftarrow 1$ to $p$
6:      $\bar{\mathcal{X}}_L^{(i)} = \bar{\Upsilon}_L^{(i)} \bar{\mathcal{A}}_L^{(i)}$, $\bar{\mathcal{Y}}_L^{(i)} = \bar{\mathcal{A}}_L^{(i)} (\bar{\Omega}_L^{(i)})^H$, $\bar{\mathcal{Z}}_L^{(i)} = \bar{\Phi}_L^{(i)} (\bar{\mathcal{A}}_L^{(i)}) (\bar{\Psi}_L^{(i)})^H$;
7:      $[\bar{\mathcal{P}}_L^{(i)}, \bar{\mathcal{R}}_L^{(i)}] = \mathtt{qr}((\bar{\mathcal{X}}_L^{(i)})^H, 0)$, $[\bar{\mathcal{Q}}_L^{(i)}, \bar{\mathcal{M}}_L^{(i)}] = \mathtt{qr}(\bar{\mathcal{Y}}_L^{(i)}, 0)$;
8:      $\bar{\mathcal{C}}_L^{(i)} = (\bar{\Phi}_L^{(i)} \bar{\mathcal{Q}}_L^{(i)})^\dagger \bar{\mathcal{Z}}_L^{(i)} ((\bar{\Psi}_L^{(i)} \bar{\mathcal{P}}_L^{(i)})^\dagger)^H$;
9:      $\tilde{\mathcal{A}}_L^{(i)} = \bar{\mathcal{Q}}_L^{(i)} \bar{\mathcal{C}}_L^{(i)} (\bar{\mathcal{P}}_L^{(i)})^H$;
10: end
11: return $\hat{\mathcal{A}}_L = L^H[\tilde{\mathcal{A}}_L]$ and $(\bar{\mathcal{Q}}_L, \bar{\mathcal{C}}_L, \bar{\mathcal{P}}_L)$.
```

We call the above three-step process our proposed two-sided sketching method, i.e., *L transformed randomized projection sketching (L-TRP-SKETCH)* algorithm. The pseudocode of our L-TRP-SKETCH algorithm is given in Algorithm 1. The storage cost for the sketches $(\mathcal{X}, \mathcal{Y}, \mathcal{Z})$ is $p(nk + mk + s^2)$ floating point numbers. The storage complexity of Algorithm 3.1 for the original data $\mathcal{A} \in \mathbb{R}^{m\times n\times p}$ is $\mathcal{O}(mkp + nkp + kkp)$ floating point numbers. Regarding the selection of the invertible transformation operator L, in addition to the DFT matrix F_{n_3} (here $n_3 = p$) and the DCT matrix C, we can also choose the U transformation matrix as in [39]. In this case, the original data $\mathcal{A} \in \mathbb{R}^{m\times n\times p}$ modulo-3 is expanded into a matrix $W \in \mathbb{R}^{p\times mn}$, and then SVD is applied on W to obtain the unitary transformation matrix $U \in \mathbb{R}^{p\times p}$.

Naming. When, based on the U transformed domain, linear dimensionality reduction mappings are chosen as the count tensor sketching operator, the Gaussian tensor sketching operator, and the SRHT tensor sketching operator, Algorithm 1 is then referred to as the U-Count-Sketch, the U-Gaussian-Sketch, and the U-SRHT-Sketch algorithms, respectively. When, based on the DFT transformed domain, linear dimensionality reduction mappings are chosen as the count tensor sketching operator, the Gaussian tensor sketching operator, and the SRHT tensor sketching operator, Algorithm 1 is then referred to as the DFT-Count-Sketch, the DFT-Gaussian-Sketch, and the DFT-SRHT-Sketch algorithms, respectively. Similarly, when, based on the DCT transformed domain, linear dimensionality reduction mappings are chosen as the count tensor sketching operator, the Gaussian tensor sketching operator, and the SRHT tensor sketching operator, Algorithm 1 is then referred to as the DCT-Count-Sketch, the DCT-Gaussian-Sketch, and the DCT-SRHT-Sketch algorithms, respectively. Finally, if the transformed domain is not specified, we just use L in the names of these algorithms; e.g., U-Gaussian-Sketch will be called L-Gaussian-Sketch.

3.2 Principle

For

$$\mathcal{A} \approx \mathcal{Q} \star_L (\mathcal{Q}^H \star_L \mathcal{A} \star_L \mathcal{P}) \star_L \mathcal{P}^H, \tag{3.22}$$

the core tensor $\mathcal{Q}^H \star_L \mathcal{A} \star_L \mathcal{P}$ cannot be calculated directly from the linear sketch since $\mathcal{P}$ and $\mathcal{Q}$ are functions of $\mathcal{A}$. Using the representation in Eq. (3.22), the core sketch $\mathcal{Z}$ estimating the core tensor can be achieved by

$$\mathcal{Z} = \Phi \star_L \mathcal{A} \star_L \Psi^H \approx (\Phi \star_L \mathcal{Q}) \star_L (\mathcal{Q}^H \star_L \mathcal{A} \star_L \mathcal{P}) \star_L (\mathcal{P}^H \star_L \Psi^H). \tag{3.23}$$

Transferring the external matrix to the left-hand side, the core approximation $\mathcal{C}$ defined in Eq. (3.18) is found to satisfy

$$\mathcal{C} = (\Phi \star_L \mathcal{Q})^\dagger \star_L \mathcal{Z} \star_L ((\Psi \star_L \mathcal{P})^\dagger)^H \approx \mathcal{Q}^H \star_L \mathcal{A} \star_L \mathcal{P}. \tag{3.24}$$

Given Eq. (3.21), (3.22) and (3.24), we have

$$\mathcal{A} \approx \mathcal{Q} \star_L (\mathcal{Q}^H \star_L \mathcal{A} \star_L \mathcal{P}) \star_L \mathcal{P}^H \approx \mathcal{Q} \star_L \mathcal{C} \star_L \mathcal{P}^H = \hat{\mathcal{A}}. \tag{3.25}$$

The error in the last relation depends on the error in the best transformed tubal rank k approximation of $\mathcal{A}$.

Algorithm 2 DCT-Gaussian-Sketch-PI

```
1: Input: Input tensor $\mathcal{A} \in \mathbb{R}^{m\times n\times p}$ and sketch size parameters $k, s, q$ with $k \leq s$.
2: function DCT-Gaussian-Sketch-PI($\mathcal{A}, k, q$)
3: Select the Gaussian tensor sketching operators from Eq. (2.13), i.e.,
   $\Upsilon \in \mathbb{R}^{k\times m\times p}, \Omega \in \mathbb{R}^{k\times n\times p}, \Phi \in \mathbb{R}^{s\times m\times p}, \Psi \in \mathbb{R}^{s\times n\times p}$;
4: $\bar{\mathcal{A}}_C = C[\mathcal{A}]$; $\bar{\Upsilon}_C = C[\Upsilon]$; $\bar{\Omega}_C = C[\Omega]$; $\bar{\Phi}_C = C[\Phi]$; $\bar{\Psi}_C = C[\Psi]$;
5: for $i \leftarrow 1$ to $p$
6:     $\bar{\mathcal{X}}_C^{(i)} = \bar{\Upsilon}_C^{(i)} \bar{\mathcal{A}}_C^{(i)}$; $\bar{\mathcal{Y}}_C^{(i)} = \bar{\mathcal{A}}_C^{(i)} (\bar{\Omega}_C^{(i)})^H$; $\bar{\mathcal{Z}}_C^{(i)} = \bar{\Phi}_C^{(i)} (\bar{\mathcal{A}}_C^{(i)}) (\bar{\Psi}_C^{(i)})^H$;
7:     $[\bar{\mathcal{P}}_C^{(i)}, \bar{\mathcal{R}}_C^{(i)}] = \mathtt{qr}((\bar{\mathcal{X}}_C^{(i)})^H, 0), [\bar{\mathcal{Q}}_C^{(i)}, \bar{\mathcal{M}}_C^{(i)}] = \mathtt{qr}(\bar{\mathcal{Y}}_C^{(i)}, 0)$;
8:     for $j \leftarrow 1$ to $q$
9:         $\tilde{\mathcal{Y}}_C^{(i)} = (\bar{\mathcal{A}}_C^{(i)})^H \bar{\mathcal{Q}}_C^{(i)}, [\tilde{\mathcal{Q}}_C^{(i)}, \sim] = \mathtt{qr}(\tilde{\mathcal{Y}}_C^{(i)}, 0)$;
10:        $\hat{\mathcal{Y}}_C^{(i)} = \bar{\mathcal{A}}_C^{(i)} \tilde{\mathcal{Q}}_C^{(i)}, [\hat{\mathcal{Q}}_C^{(i)}, \sim] = \mathtt{qr}(\hat{\mathcal{Y}}_C^{(i)}, 0)$;
11:        $\tilde{\mathcal{X}}_C^{(i)} = \bar{\mathcal{A}}_C^{(i)} \bar{\mathcal{P}}_C^{(i)}, [\tilde{\mathcal{P}}_C^{(i)}, \sim] = \mathtt{qr}(\tilde{\mathcal{X}}_C^{(i)}, 0)$;
12:        $\hat{\mathcal{X}}_C^{(i)} = (\bar{\mathcal{A}}_C^{(i)})^H \tilde{\mathcal{P}}_C^{(i)}, [\hat{\mathcal{P}}_C^{(i)}, \sim] = \mathtt{qr}(\hat{\mathcal{X}}_C^{(i)}, 0)$;
13:        $\bar{\mathcal{Q}}_C^{(i)} = \hat{\mathcal{Q}}_C^{(i)}, \bar{\mathcal{P}}_C^{(i)} = \hat{\mathcal{P}}_C^{(i)}$;
14:    end
15:    $\bar{\mathcal{C}}_C^{(i)} = (\bar{\Phi}_C^{(i)} \bar{\mathcal{Q}}_C^{(i)})^\dagger \bar{\mathcal{Z}}_C^{(i)} ((\bar{\Psi}_C^{(i)} \bar{\mathcal{P}}_C^{(i)})^\dagger)^H$;
16:    $\tilde{\mathcal{A}}_C^{(i)} = \bar{\mathcal{Q}}_C^{(i)} \bar{\mathcal{C}}_C^{(i)} (\bar{\mathcal{P}}_C^{(i)})^H$;
17: end
18: return $\hat{\mathcal{A}} = C^H[\tilde{\mathcal{A}}_C]$ and $(\bar{\mathcal{Q}}_C, \bar{\mathcal{C}}_C, \bar{\mathcal{P}}_C)$.
```

3.3 Extension

We now showcase one extension of the proposed two-sided sketching algorithms by exploiting the power iteration technique. As shown in [42], the power iteration technique is useful to improve sketching algorithms for low-rank matrix approximation. Here, as an example, we can combine the power iteration technique with the DCT-Gaussian-Sketch algorithm, in which we exploit the third order tensor say $\mathcal{B} = (\mathcal{A} \star_L \mathcal{A}^H)^q \star_L \mathcal{A}$ (where q is a nonnegative integer) instead of the original tensor $\mathcal{A}$; and the DCT-Gaussian-Sketch algorithm is applied to the new tensor $\mathcal{B}$. According to the transformed tensor SVD, i.e., $\mathcal{A} = \mathcal{U} \star_L \mathcal{S} \star_L \mathcal{V}^H$, we have

$$\mathcal{B} = \mathcal{U} \star_L (\mathcal{S})^{2q+1} \star_L \mathcal{V}^H. \tag{3.26}$$

Therefore, the transformed tensor singular values of $\mathcal{B}$ have a faster decay rate. This way can improve the solution obtained by the DCT-Gaussian-Sketch algorithm. The scheme of the proposed algorithm DCT-Gaussian-Sketch with power iteration, named DCT-Gaussian-Sketch-PI, is summarized in Algorithm 2.

4 Theoretical Analysis

The error bound of the proposed two-sided sketching algorithms is given in Theorems 4.1 and 4.2 below. The detailed proofs used in the proof of Theorems 4.1 and 4.2 can be found in Appendix B.

Theorem 4.1 *Assume that the sketch parameters satisfy $s \geq 2k+1$ and $\hat{\mathcal{A}}$ is the transformed tensor tubal rank-k approximation of $\mathcal{A}$ defined by the L-Gaussian-Sketch algorithm (i.e., Algorithm 1 with the Gaussian tensor sketching operator selected). Then*

$$\mathbb{E}\|\mathcal{A}-\hat{\mathcal{A}}\|_F^2 \leq (1+f(k,s))(1+\frac{2\varrho}{k-\varrho-1})\tau_{\varrho+1}^2(\mathcal{A}\star_L\mathcal{A}^H), \tag{4.27}$$

where ϱ is a natural number less than $k-1$, $f(\varrho,k) := \varrho/(k-\varrho-1)$, and the tail energy $\tau_{\varrho+1}^2$ is defined by Definition 2.7.

proof We have

$$\begin{aligned}
&\mathbb{E}\|\mathcal{A}-\hat{\mathcal{A}}\|_F^2\\
=\ &\mathbb{E}_\Upsilon\mathbb{E}_\Omega\|\mathcal{A}-\mathcal{Q}\star_L\mathcal{Q}^H\star_L\mathcal{A}*\mathcal{P}\star_L\mathcal{P}^H\|_F^2+\mathbb{E}\|\mathcal{C}-\mathcal{Q}^H\star_L\mathcal{A}\star_L\mathcal{P}\|_F^2\\
=\ &(1+f(k,s))\mathbb{E}_\Upsilon\mathbb{E}_\Omega\|\mathcal{A}-\mathcal{Q}\star_L\mathcal{Q}^H\star_L\mathcal{A}\star_L\mathcal{P}\star_L\mathcal{P}^H\|_F^2\\
&+\frac{k(2k+1-s)}{(s-k-1)^2}\mathbb{E}\|\mathcal{Q}_\perp^H\star_L\mathcal{A}\star_L\mathcal{P}_\perp\|_F^2\\
\leq\ &(1+f(k,s))(1+\frac{2\varrho}{k-\varrho-1})\tau_{\varrho+1}^2(\mathcal{A}\star_L\mathcal{A}^H)\\
&+\frac{k(2k+1-s)}{(s-k-1)^2}\mathbb{E}\|\mathcal{Q}_\perp^H\star_L\mathcal{A}\star_L\mathcal{P}_\perp\|_F^2\\
\leq\ &(1+f(k,s))(1+\frac{2\varrho}{k-\varrho-1})\tau_{\varrho+1}^2(\mathcal{A}\star_L\mathcal{A}^H).
\end{aligned}$$

In particular, the first equation is known by Proposition 6.3 in Appendix B; the second is known by Lemma 6.6 in Appendix B; the first inequality is known by Theorem 6.4 in Appendix B; and the last inequality is because we require $s \geq 2k+1$, and thus the missing item $\frac{k(2k+1-s)}{(s-k-1)^2}\mathbb{E}\|\mathcal{Q}_\perp^H\star_L\mathcal{A}\star_L\mathcal{P}_\perp\|_F^2$ is negative. This completes the proof.

Theorem 4.2 *Assume that the sketch parameters satisfy $s \geq 2k+1$ and $\hat{\mathcal{A}}$ is the transformed tensor tubal rank-k approximation of $\mathcal{A}$ defined by Algorithm 2. Then*

$$\mathbb{E}\|\mathcal{A}-\hat{\mathcal{A}}\|_F^2 \leq (1+f(k,s))(1+\frac{2\varrho}{k-\varrho-1})\tau_{\varrho+1}^2((\mathcal{A}\star_L\mathcal{A}^H)^{(2q+1)}), \tag{4.28}$$

where q is a nonnegative integer, ϱ is a natural number less than $k-1$, $f(\varrho,k) := \varrho/(k-\varrho-1)$, the tail energy $\tau_{\varrho+1}^2$ is defined by Definition 2.7, and L here is C in Algorithm 2.

proof We exploit the third order tensor $(\mathcal{A}\star_L\mathcal{A}^H)^q\star_L\mathcal{A}$ instead of the original tensor $\mathcal{A}$. Following the proof in Theorem 4.1, we have

$$\begin{aligned}
&\mathbb{E}\|\mathcal{A}-\hat{\mathcal{A}}\|_F^2\\
=\,&\mathbb{E}_\Upsilon\mathbb{E}_\Omega\|\mathcal{A}-\mathcal{Q}\star_L\mathcal{Q}^H\star_L(\mathcal{A}\star_L\mathcal{A}^H)^q\star_L\mathcal{A}\star_L\mathcal{P}\star_L\mathcal{P}^H\|_F^2\\
&+\mathbb{E}\|\mathcal{C}-\mathcal{Q}^H\star_L(\mathcal{A}\star_L\mathcal{A}^H)^q\star_L\mathcal{A}\star_L\mathcal{P}\|_F^2\\
=\,&(1+f(k,s))\mathbb{E}_\Upsilon\mathbb{E}_\Omega\|\mathcal{A}-\mathcal{Q}\star_L\mathcal{Q}^H\star_L(\mathcal{A}\star_L\mathcal{A}^H)^q\star_L\mathcal{A}\star_L\mathcal{P}\star_L\mathcal{P}^H\|_F^2\\
&+\frac{k(2k+1-s)}{(s-k-1)^2}\mathbb{E}\|\mathcal{Q}_\perp^H\star_L(\mathcal{A}\star_L\mathcal{A}^H)^q\star_L\mathcal{A}\star_L\mathcal{P}_\perp\|_F^2\\
\le\,&(1+f(k,s))(1+\frac{2\varrho}{k-\varrho-1})\tau_{\varrho+1}^2\big((\mathcal{A}\star_L\mathcal{A}^H)^q\star_L\mathcal{A}\star_L((\mathcal{A}\star_L\mathcal{A}^H)^q\star_L\mathcal{A})^H\big)\\
&+\frac{k(2k+1-s)}{(s-k-1)^2}\mathbb{E}\|\mathcal{Q}_\perp^H\star_L(\mathcal{A}\star_L\mathcal{A}^H)^q\star_L\mathcal{A}\star_L\mathcal{P}_\perp\|_F^2\\
\le\,&(1+f(k,s))(1+\frac{2\varrho}{k-\varrho-1})\tau_{\varrho+1}^2((\mathcal{A}\star_L\mathcal{A}^H)^{2q+1}).
\end{aligned}$$

This completes the proof.

5 Numerical Experiments

This section showcases numerical experiments validating the efficiency and effectiveness of the proposed two-sided sketching algorithms, in comparison with the state-of-the-art algorithms including the T-Sketch algorithm [22, Algorithm 2], T-Sketch-PI algorithm ($q=1$) [22], truncated-t-SVD algorithm [38], and rt-SVD algorithm (i.e., a one-sided randomized algorithm based on t-SVD in [40, Algorithm 6]). The sketch size parameters are set to $s=2k+1$. The following relative error ϵ_{err} and the peak signal-to-noise ratio (PSNR) ρ_{psnr} are used as metrics of the low-rank approximation $\hat{\mathcal{A}}$ to the input tensor data $\mathcal{A}$, i.e.,

$$\epsilon_{\text{err}} := ||\mathcal{A}-\hat{\mathcal{A}}||_F^2/||\mathcal{A}||_F^2, \quad \rho_{\text{psnr}} := 10\log_{10}\frac{n_1n_2n_3||\mathcal{A}||_\infty^2}{||\mathcal{A}-\hat{\mathcal{A}}||_F^2}. \tag{5.29}$$

5.1 Synthetic Experiments

We firstly conduct numerical tests on some synthetic input tensors $\mathcal{A}\in\mathbb{R}^{10^3\times10^3\times10}$ with decaying spectrum.

Polynomial decay: These tensors are f-diagonal tensors. Considering their j-th frontal slices with the form

$$\mathcal{A}^{(j)} = \operatorname{diag}\big(\underbrace{1,\dots,1}_{\min(r,j)}, 2^{-p}, 3^{-p}, 4^{-p}, \dots, (n-\min(r,j)+1)^{-p}\big) \in \mathbb{R}^{n\times n}, \qquad (5.30)$$

we study two examples, i.e., PolyDecaySlow ($p = 0.5$) and PolyDecayFast ($p = 2$).

Figures 1–3 give the results of the algorithms compared in terms of the relative error, CPU time, and PSNR, as the size of the sketch parameter k varies; see more details below.

5.1.1 Experiments Regarding the Transformed Domains

We first examine the performance of different transformed domains in our two-sided sketching method. Figure 1 shows the results of different methods and our method via different transformed domains (specifically, U-Gaussian-Sketch, DCT-Gaussian-Sketch, and DFT-Gaussian-Sketch algorithms) in terms of the relative error, CPU time, and PSNR. Figure 1 illustrates that among our two-sided Gaussian sketching algorithms with different transformed domains, they all perform similarly in terms of the relative error and PSNR. Regarding the CPU time, our DCT-Gaussian-Sketch and U-Gaussian-Sketch algorithms outperform all other methods including the rt-SVD, truncated-t-SVD, and T-Sketch algorithms. This means, regarding the two-sided Gaussian sketching algorithms employing various transformation operators, the DCT and U transforms emerge as the most efficient.

As shown in the second row of Figure 1, our two-sided sketching method with different transformed domains all surpasses the accuracy of the rt-SVD and T-Sketch algorithms for input tensors exhibiting a fast decay spectrum. Consequently, with reduced storage requirements and manipulation, our two-sided sketching method with different transformed domains provides superior accuracy in low-rank approximations. It is worth highlighting that, for input tensors exhibiting a fast decay spectrum, our method is the second only to the truncated-t-SVD algorithm in terms of accuracy, but our method offers a significant speed advantage, i.e, the truncated-t-SVD is far slower than our method (see the middle plot of the second column of Figure 1).

5.1.2 Experiments Regarding the Tensor Sketching Operators

We now examine the performance of different tensor sketching operators in our two-sided sketching method. As an example, our method here is adopted with the DCT transformed domain. Figure 2 shows the results of different methods and our method via different tensor sketching operators (specifically, DCT-Gaussian-Sketch, DCT-SRHT-Sketch, DCT-Count-Sketch algorithms) in terms of the relative error, CPU time, and

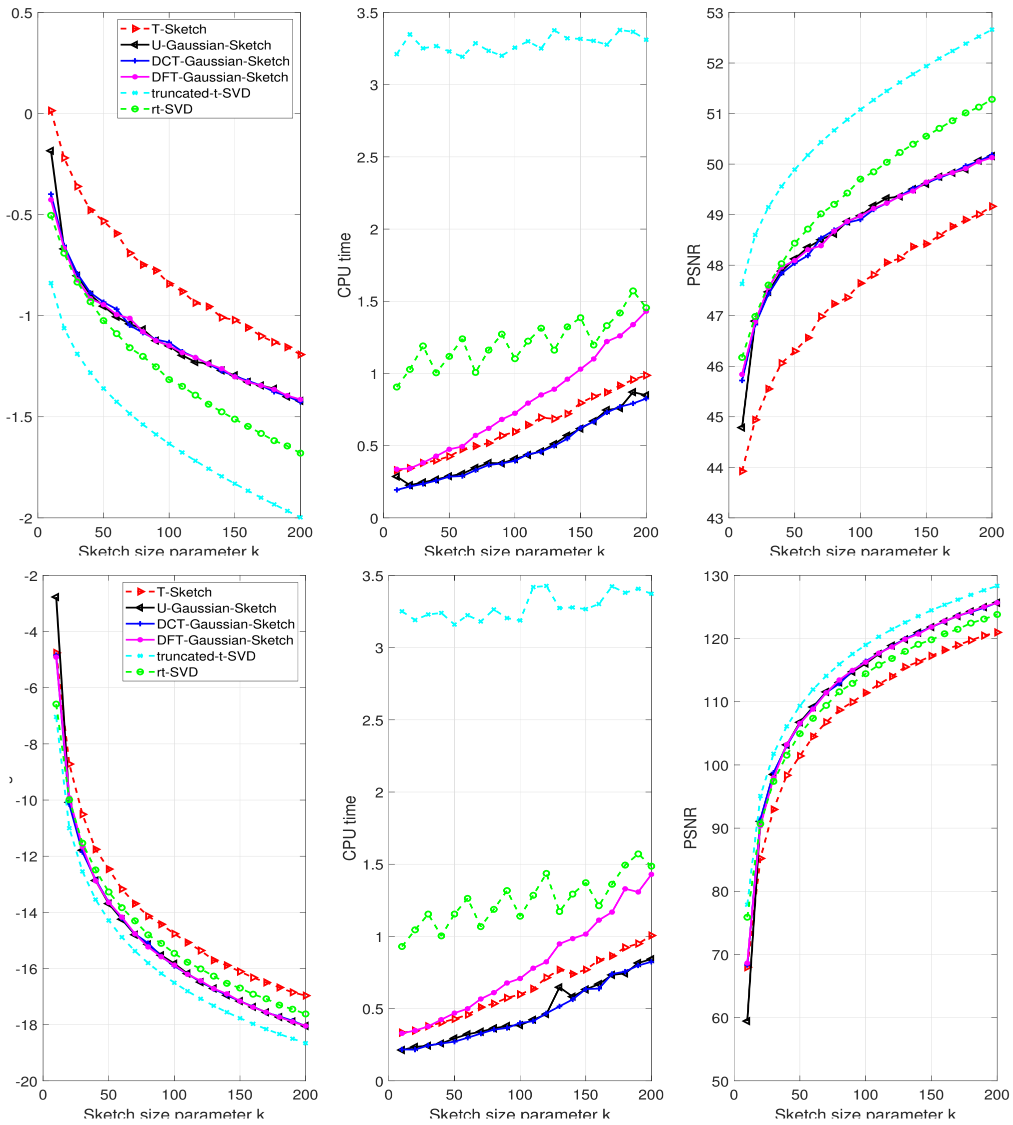


Figure 1: Performance of different methods and our method via different transformed domains (i.e., U, DCT, and DFT transforms) in terms of the relative error (*left* column), CPU time (*middle* column), and PSNR (*right* column). In particular, the first row is for tensor data (PolyDecaySlow) with slow decaying spectrum and the second row is for tensor data (PolyDecayFast) with fast decaying spectrum.

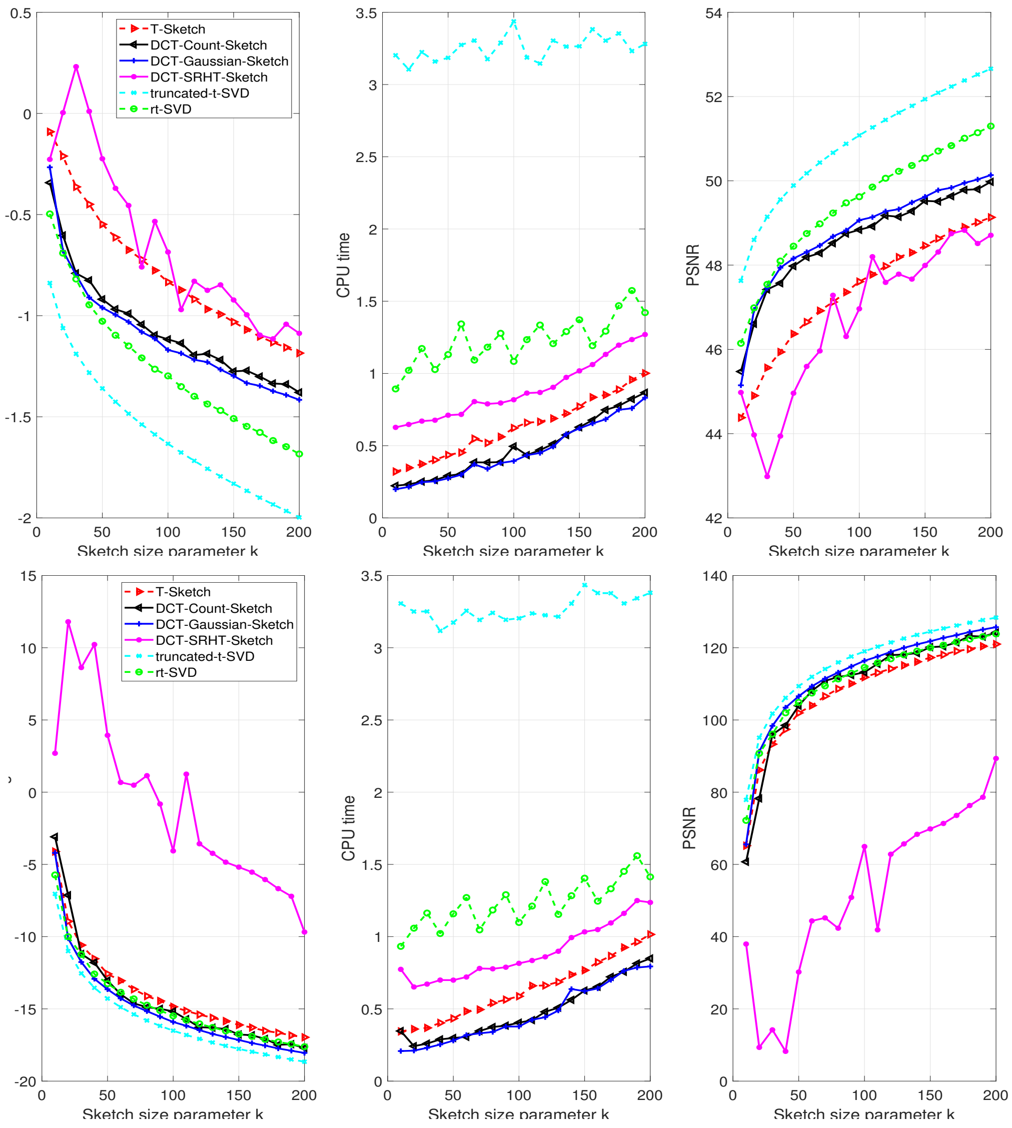


Figure 2: Performance of different methods and our method based on the DCT transformed domain and via different tensor sketching operators (i.e., Gaussian, SRHT, and count sketching operators) in terms of the relative error (*left* column), CPU time (*middle* column), and PSNR (*right* column). In particular, the first row is for tensor data (PolyDecaySlow) with slow decaying spectrum and the second row is for tensor data (PolyDecayFast) with fast decaying spectrum.

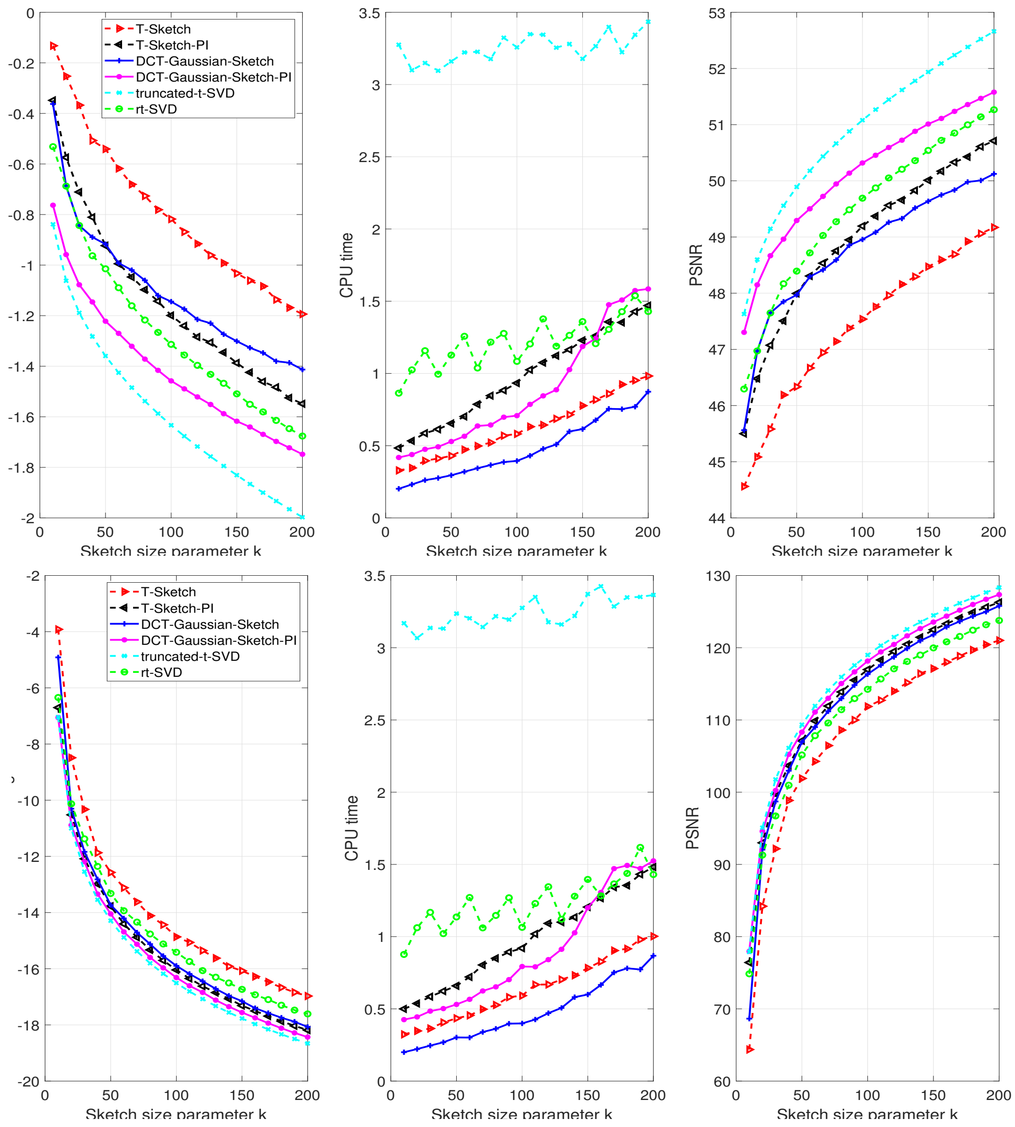


Figure 3: Performance of different methods and our method based on the DCT transformed domain and the Gaussian tensor sketching operator with and without the power iteration technique in terms of the relative error (*left* column), CPU time (*middle* column), and PSNR (*right* column). In particular, the first row is for tensor data (PolyDecaySlow) with slow decaying spectrum and the second row is for tensor data (PolyDecayFast) with fast decaying spectrum.

PSNR. Figure 2 illustrates that among our two-sided sketching algorithms based on the DCT transformed domain, the DCT-Gaussian-Sketch algorithm outperforms the other two in terms of accuracy, indicating the better effectiveness of the Gaussian tensor sketching operator. Regarding the CPU time, our DCT-Gaussian-Sketch and DCT-Count-Sketch algorithms outperform all other methods including the rt-SVD, truncated-t-SVD, and T-Sketch algorithms. This means, regarding the two-sided sketching algorithms (based on the DCT transformed domain) employing various tensor sketching operators, the Gaussian and count sketching operators emerge as the most efficient. Considering both the accuracy and speed, the DCT-Gaussian-Sketch algorithm is the best among our method with different tensor sketching operators.

As shown in the second row of Figure 2, consistent results are obtained for our DCT-Gaussian-Sketch algorithm as that in Figure 1. It is worth highlighting that, for input tensors exhibiting a fast decay spectrum, our DCT-Gaussian-Sketch algorithm is the second only to the truncated-t-SVD algorithm in terms of accuracy, but it offers a significant speed advantage, i.e., the truncated-t-SVD is far slower than our DCT-Gaussian-Sketch algorithm (see the middle plot of the second column of Figure 2).

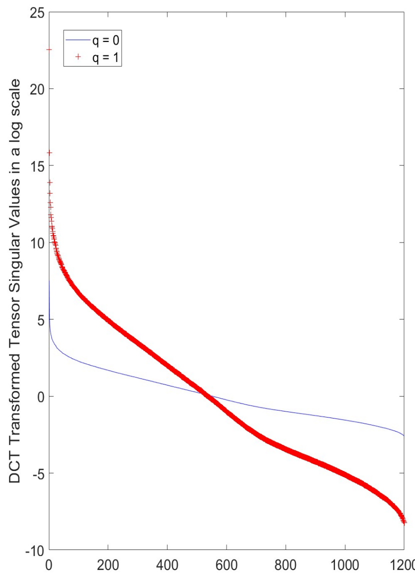
5.1.3 Experiments Regarding the Power Iteration Technique

Figure 3 shows the results of different methods and our method based on the DCT transformed domain and the Gaussian tensor sketching operator with and without the power iteration technique (i.e., DCT-Gaussian-Sketch and DCT-Gaussian-Sketch-PI ($q = 1$)) in terms of the relative error, CPU time, and PSNR. For the input tensors exhibiting both slow and fast decay spectra, Figure 3 shows that DCT-Gaussian-Sketch-PI can indeed outperform DCT-Gaussian-Sketch in terms of accuracy, with a litter sacrifice of speed, indicating the effectiveness of the power iteration technique in approximation quality enhancement. Moreover, for the comparison between our DCT-Gaussian-Sketch-PI algorithm, the T-Sketch algorithm with the power iteration technique (i.e., T-Sketch-PI), and the rt-SVD algorithm, our method is superior in all metrics.

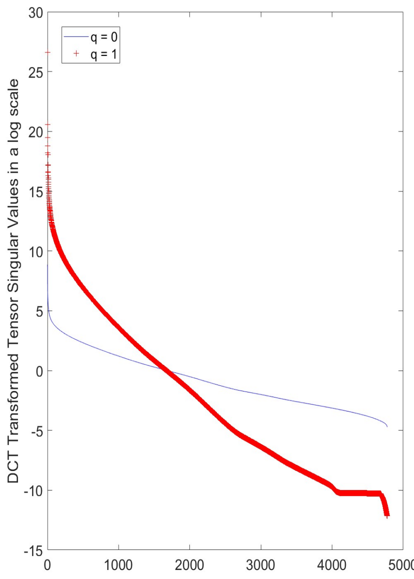
With reduced storage requirements and manipulation, as shown in the second row of Figure 3 for input tensors exhibiting a fast decay spectrum, our DCT-Gaussian-Sketch-PI algorithm is the second only to the truncated-t-SVD algorithm in terms of accuracy, but our algorithm again offers a significant speed advantage, i.e, the truncated-t-SVD is far slower than our DCT-Gaussian-Sketch-PI algorithm (see the middle plot of the second column of Figure 3).

(a) HDU image (size: $1,200 \times 1,800$)

(b) London image (size: $4,775 \times 7,155$)



(c) Transformed tensor singular values of (a)



(d) Transformed tensor singular values of (b)

Figure 4: Test real-world color images and the DCT transformed tensor singular values in terms of their decaying spectrum. Row one: the original color images. Row two: the transformed tensor singular values of the power iteration ($q = 0$) and ($q = 1$) for the given color images.

5.2 Real-world Data

We now conduct experiments on real-world data including color images and grayscale videos. For our method, we select the DCT-Gaussian-Sketch and DCT-Gaussian-Sketch-PI ($q = 1$) algorithms, given their great performance demonstrated in the previous section.

5.2.1 Color Images

Two large size color images, i.e., HDU picture[1] with size of $1,200 \times 1,800 \times 3$ and London picture with size of $4,775 \times 7,155 \times 3$, are employed, see the first row of Figure

[1]https://www.hdu.edu.cn/landscape

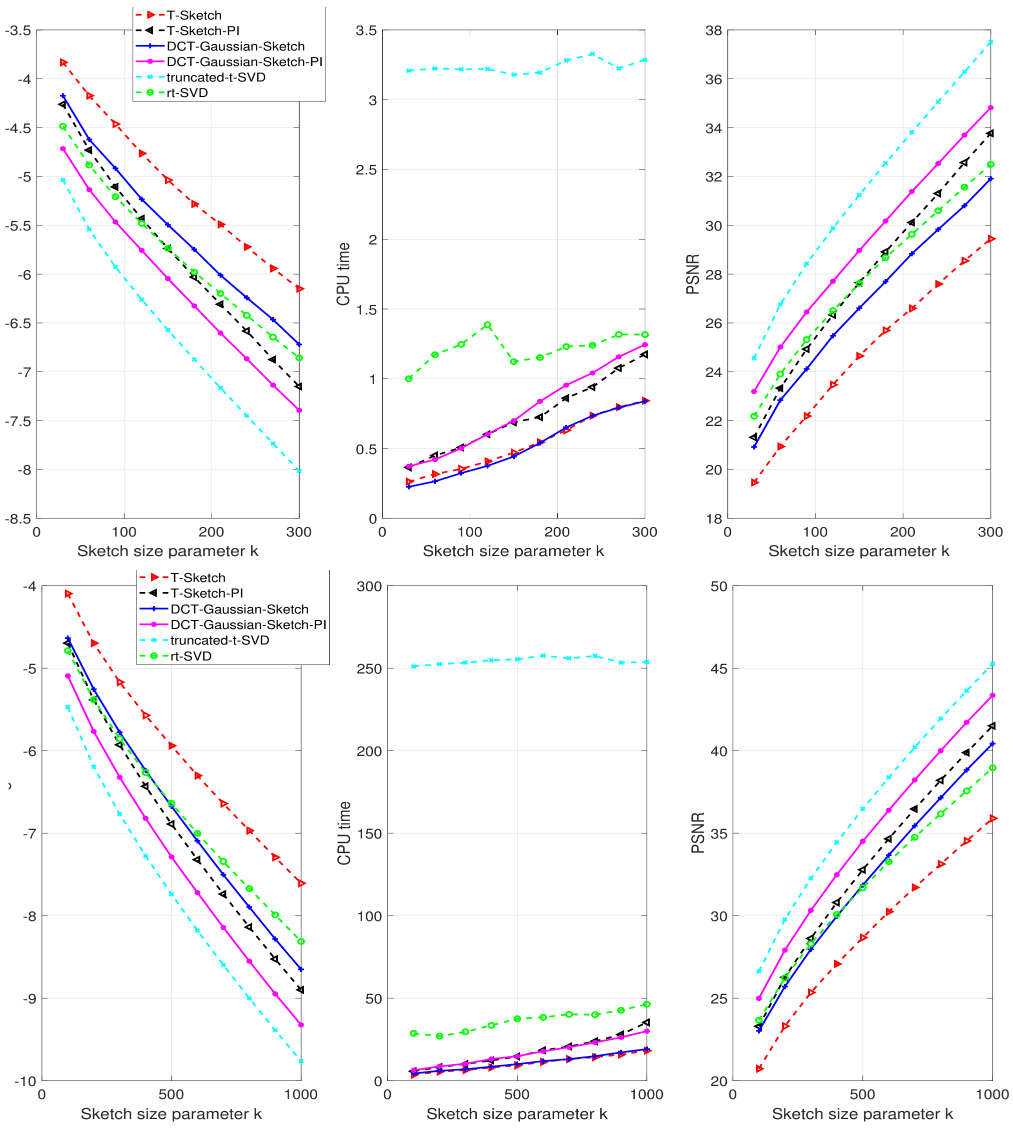


Figure 5: Low-rank approximation performance of different methods on the HDU image (*first* row) and the London image (*second* row) in terms of the relative error (*left* column), CPU time (*middle* column), and PSNR (*right* column).

4.

The second row of Figure 4 gives the ordered transformed tensor singular values (indicating the decaying spectrum) using the DCT transformed tensor singular values decomposition on the two original color images. It clearly shows that the one with power iteration achieves a faster decaying spectrum compared to that without power iteration, demonstrating the great effectiveness of the power iteration technique.

(a1) T-Sketch
CPU:0.84; PSNR: 29.44

(b1) T-Sketch-PI
CPU:1.17; PSNR: 33.77

(c1) DCT-Gaussian-Sketch
CPU: 0.84; PSNR: 31.91

(d1) DCT-Gaussian-Sketch-PI
CPU: 1.24; PSNR: 34.82

(e1) truncated-t-SVD
CPU: 3.29; PSNR: 37.51

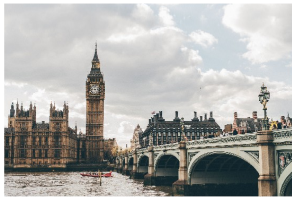
(f1) rt-SVD
CPU: 1.32; PSNR: 32.49

(a2) T-Sketch
CPU:11.55; PSNR: 30.20

(b2) T-Sketch-PI
CPU:17.32; PSNR: 34.67

(c2) DCT-Gaussian-Sketch
CPU: 11.12; PSNR: 33.67

(d2) DCT-Gaussian-Sketch-PI
CPU: 17.06; PSNR: 36.38


(e2) truncated-t-SVD
CPU: 252.40; PSNR: 38.38

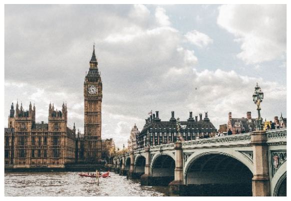

(f2) rt-SVD
CPU: 38.81; PSNR: 33.26

Figure 6: Qualitative results of the low-rank approximation performance of different methods on the HDU image (rows 1 and 2) with the sketch size $k = 300$ and the London image (rows 3 and 4) with the sketch size $k = 600$ in terms of the CPU time and PSNR.

Figures 5 and 6 respectively give the quantitative and qualitative results of the low-rank approximation performance of different methods on the test color images in terms of different metrics. It is evident that as the sketch parameter size k increases, the low-rank approximation performance of all the methods improves. Consistent results are obtained as that obtained from the previous section for synthetic tensors. Regarding the CPU time, our DCT-Gaussian-Sketch algorithm and the T-Sketch algorithm outperform all others. In particular, the second row of Figure 5 (results for the London image) shows that the DCT-Gaussian-Sketch algorithm surpasses both the rt-SVD and T-Sketch algorithms in terms of accuracy, with reduced storage and manipulation requirements. The DCT-Gaussian-Sketch-PI algorithm achieves better accuracy compared to the T-Sketch-PI algorithm with similar speed. The DCT-Gaussian-Sketch-PI algorithm is only second to the truncated-t-SVD algorithm in accuracy, but it is remarkably faster than the truncated-t-SVD algorithm.

5.2.2 Color Image with Gaussian White Noise

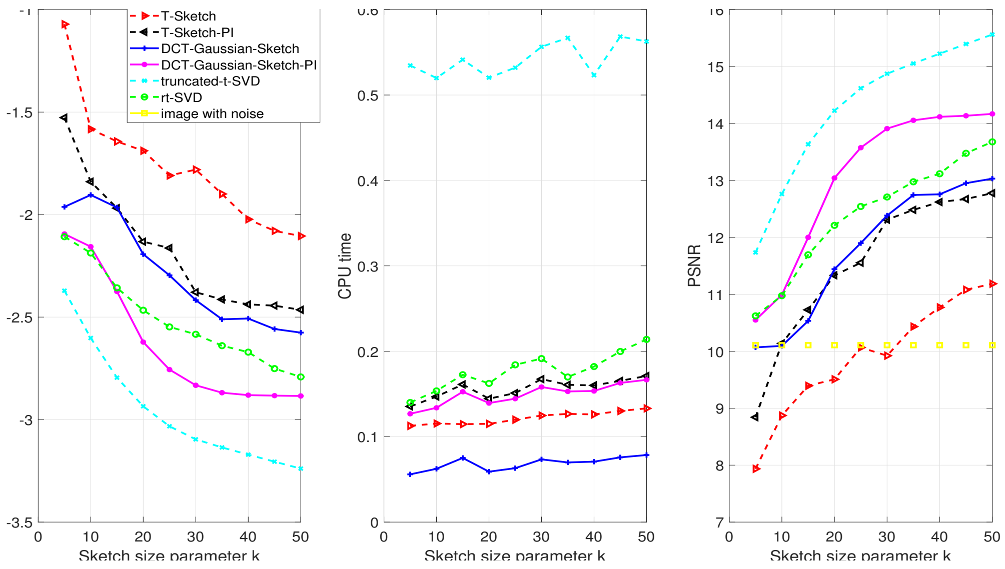
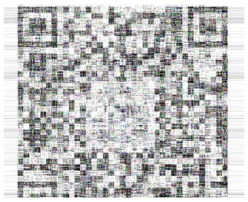


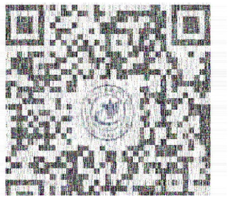
Figure 7: Low-rank approximation performance of different methods on the HDU 2D code image with Gaussian white noise in terms of the relative error (*left* column), CPU time (*middle* column), and PSNR (*right* column). In particular, the yellow line named 'image with noise' is utilized to plot the PSNR of the HDU 2D code image with Gaussian white noise.

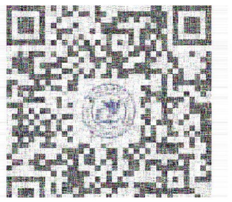
We now inspect the impact of noise on the performance of different methods. Specifically, the input tensor is randomly generated as

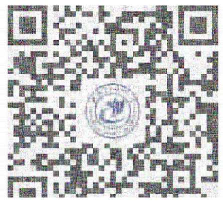
$$\mathcal{A} \leftarrow \mathcal{A} + \sigma \cdot \mathcal{E},$$

(a) Image with noise
PSNR: 10.11


(b) T-Sketch
CPU:0.13; PSNR: 11.19


(c) T-Sketch-PI
CPU: 0.17; PSNR: 12.78


(d) DCT-Gaussian-Sketch
CPU: 0.08; PSNR: 13.03


(e) DCT-Gaussian-Sketch-PI
CPU: 0.17; PSNR: 14.17

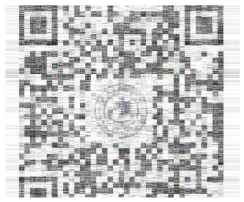

(f) rt-SVD
CPU: 0.21; PSNR: 13.68

Figure 8: Qualitative results of the low-rank approximation performance of different methods on the HDU 2D code image with Gaussian white noise and the sketch size $k = 50$ in terms of the CPU time and PSNR.

where $\mathcal{E}$ represents the Gaussian noise tensor following the standard Gaussian distribution, and σ controls the noise level and is set to 5 here.

Figures 7 and 8 respectively give the quantitative and qualitative results of the low-rank approximation performance of different methods on the HDU 2D code image (with size of $800 \times 800 \times 3$) with Gaussian white noise in terms of different metrics. It is again evident that as the sketch parameter size k increases, the low-rank approximation performance of all the methods improves. Regarding the CPU time, our DCT-Gaussian-Sketch algorithm outperforms all others. The DCT-Gaussian-Sketch algorithm also surpasses the T-Sketch algorithm in terms of accuracy, with reduced storage and manipulation requirements. The DCT-Gaussian-Sketch-PI algorithm achieves better accuracy by a large margin compared to the T-Sketch-PI algorithm with faster speed. Moreover, the DCT-Gaussian-Sketch-PI algorithm is only second to the truncated-t-SVD algorithm in accuracy, but it is remarkably faster than the truncated-t-SVD algorithm. In addition, from the right column of Figure 7, it can be seen that when $k \geq 10$, the PSNR obtained by the DCT-Gaussian-Sketch and the DCT-Gaussian-Sketch-PI algorithms is higher than the original image with Gaussian white noise, indicating the effectiveness of our method both in tensor approximation and the denoising capacity.

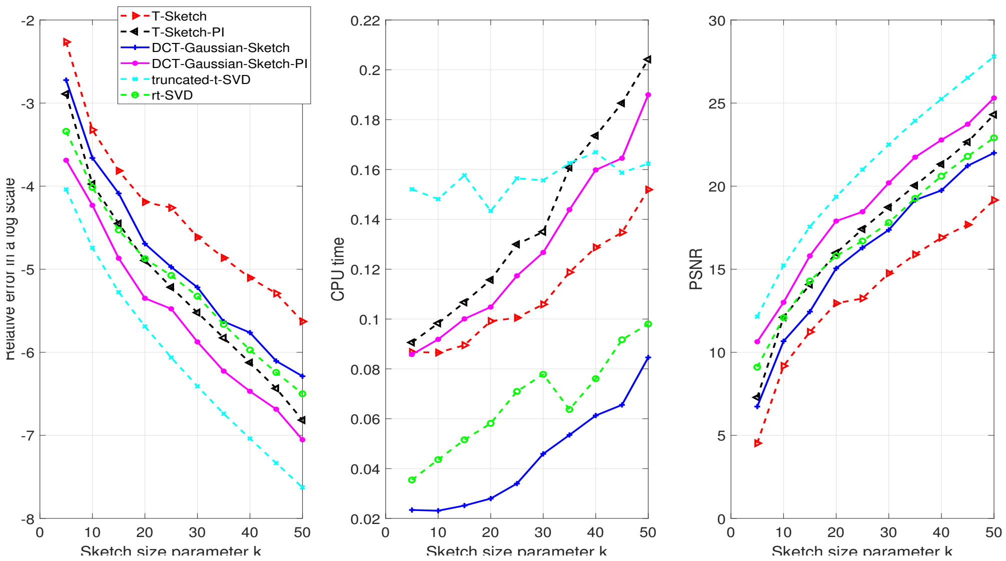
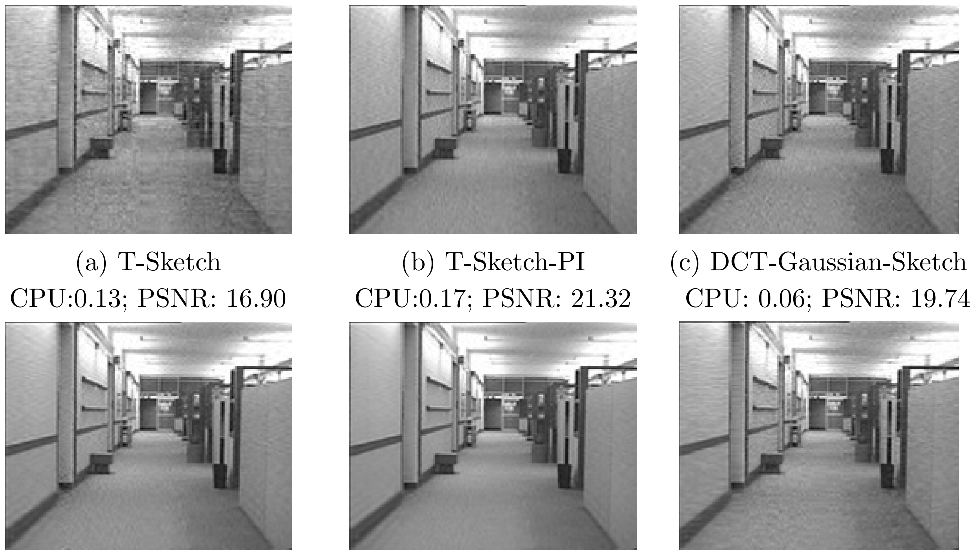


Figure 9: Low-rank approximation performance of different methods on the 'hall monitor' grayscale video clip (size: $144 \times 176 \times 30$) in terms of the relative error (*left* column), CPU time (*middle* column), and PSNR (*right* column).



(a) T-Sketch
CPU:0.13; PSNR: 16.90

(b) T-Sketch-PI
CPU:0.17; PSNR: 21.32

(c) DCT-Gaussian-Sketch
CPU: 0.06; PSNR: 19.74

(d) DCT-Gaussian-Sketch-PI
CPU: 0.16; PSNR: 22.78

(e) truncated-t-SVD
CPU: 0.17; PSNR: 25.24

(f) rt-SVD
CPU: 0.08; PSNR: 20.60

Figure 10: Qualitative results of the low-rank approximation performance of different methods on the first frame of the 'hall monitor' grayscale video clip with the sketch size $k = 40$ in terms of the CPU time and PSNR.

5.2.3 Grayscale Video

We ultimately assess the proposed sketching algorithms and the peers using the extensively adopted YUV Video Sequences[2]. As an example, we utilize the 'hall monitor' video clip and its first 30 frames to create a three-order tensor with dimensions $144 \times 176 \times 30$ for this experiment.

Figures 9 and 10 respectively give the quantitative and qualitative results of the low-rank approximation performance of different methods on the 'hall monitor' graysc-ale video clip in terms of different metrics. Consistent results are obtained as that obtained from the previous section for synthetic tensors and real-world images. For example, regarding the CPU time, our DCT-Gaussian-Sketch algorithm again outperforms all others. Moreover, it also clearly surpasses the T-Sketch algorithm in terms of accuracy, with reduced storage and manipulation requirements. The DCT-Gaussian-Sketch-PI algorithm achieves better accuracy compared to the T-Sketch-PI algorithm with faster speed. The DCT-Gaussian-Sketch-PI algorithm is only second to the truncated-t-SVD algorithm in accuracy, but it is faster than the truncated-t-SVD algorithm.

6 Conclusion

In this paper, we focused on large-scale tensor decomposition and proposed a novel two-sided sketching method based on the $\star_L$-product decomposition and transformed domains. Different transformed domains including the U, DCT, and DFT domains were investigated, together with different tensor sketching operators and the extension with the power iteration technique. A rigorous theoretical analysis was also conducted to assess the approximation error of the proposed method, particularly for the case of using the DCT transformation and the Gaussian tensor sketching operator with and without power iteration. Extensive numerical experiments and comparisons on low-rank approximation of synthetic large tensors and real-world data like color images and grayscale videos demonstrated the efficiency and effectiveness of the proposed approach in terms of both CPU time and low-rank approximation effects.

Appendix A

This appendix is about random projection. Three matrix sketching techniques, i.e., Gaussian projection, subsampled randomized Hadamard transform (SRHT), and count sketch [43], are introduced below.

[2]http://trace.eas.asu.edu/yuv/index.html

A.1. Gaussian Projection

The Gaussian random projection matrix $S \in \mathbb{R}^{n\times s}$ is a matrix formed by $S = G/\sqrt{s}$, where each entry of G is sampled i.i.d. from $\mathcal{N}(0, 1)$. For matrix $A \in \mathbb{R}^{m\times n}$, the time complexity of Gaussian projection is $\mathcal{O}(mns)$. Algorithm 3 presents the Gaussian projection process.

Algorithm 3 Gaussian Projection

```
1: Input: $A \in \mathbb{R}^{m\times n}$, and parameter $s$.
2: function GaussianProjection($A, s$)
3: Generate Gaussian random matrix $G \in \mathbb{R}^{n\times s}$;
4: $S = \frac{1}{\sqrt{s}} G$;
5: $C = AS$;
6: return $C \in \mathbb{R}^{m\times s}$.
```

A.2. Subsampled Randomized Hadamard Transform

The SRHT matrix is defined by $S = DH_nP/\sqrt{sn} \in \mathbb{R}^{n\times s}$, where

- $D \in \mathbb{R}^{n\times n}$ is a diagonal matrix with diagonal entries sampled uniformly from $\{+1, 1\}$.
- $H_n \in \mathbb{R}^{n\times n}$ is the Hadamard matrix defined recursively by

$$H_n = \begin{pmatrix} H_{\frac{n}{2}} & H_{\frac{n}{2}} \\ H_{\frac{n}{2}} & -H_{\frac{n}{2}} \end{pmatrix} \quad \text{and} \quad H_2 = \begin{pmatrix} +1 & +1 \\ +1 & -1 \end{pmatrix}.$$

  Note that the order of the Hadamard matrix is usually a power of 2. $\forall y \in \mathbb{R}^n$, the matrix vector product $y^H H_n$ can be performed in $\mathcal{O}(n \log n)$ by the fast Walsh-Hadamard transform algorithm in a divide-and-conquer fashion.
- $P \in \mathbb{R}^{n\times s}$ is a matrix that essentially samples s columns from DH_n.

Algorithm 4 presents the SRHT process.

A.3. Count Sketch

There are two approaches, namely "map-reduce fashion" and "streaming fashion", to implement count sketch. Both approaches are equivalent, and we will focus on the streaming fashion here. The streaming fashion involves two steps: (i) the $m \times s$ matrix C is initialized to zero; and (ii) for each column of A, its sign is randomly flipped with

Algorithm 4 Subsampled Randomized Hadamard Transform (SRHT)

```
1: Input: $A \in \mathbb{R}^{m \times n}$, and parameter $s$.
2: function SRHT$(A, s)$
3: Generate matrices $D \in \mathbb{R}^{n \times n}$, $H_n \in \mathbb{R}^{n \times n}$ and $P \in \mathbb{R}^{n \times s}$;
4: $S = \frac{1}{\sqrt{sn}} D H_n P$;
5: $C = AS$;
6: return $C \in \mathbb{R}^{m \times s}$.
```

a probability of 0.5 and the flipped column is then added to a randomly chosen column of C.

The count sketch process in the streaming fashion is detailed in Algorithm 5. The streaming fashion keeps matrix C in memory and processes the data A in a single pass. When A does not fit into memory, this approach is more efficient than the map-reduce fashion because it processes columns sequentially. Furthermore, if A is a sparse matrix, sequentially accessing columns can be more efficient than random access. It is noticed that the count sketch does not explicitly form the sketching matrix S like the Gaussian projection and the SHRT processes. In fact, S for count sketch is such a matrix that its each row has only one nonzero entry. The time complexity of count sketch is $\mathcal{O}(\text{nnz}(A))$, where $\text{nnz}(A)$ represents the number of nonzeros of matrix A.

Algorithm 5 Count Sketch in the Streaming Fashion

```
1: Input: $A \in \mathbb{R}^{m \times n}$, and parameter $s$.
2: function CountSketch$(A, s)$
3: Initialize $C$ to be an $m \times s$ all-zero matrix;
4: for $i = 1$ to $n$ do
5:     sample $l$ from the set $[s]$ uniformly at random;
6:     sample $g$ from the set $\{+1, -1\}$ uniformly at random;
7:     update the $l$-th column of $C$ by $C(:, l) \leftarrow C(:, l) + gA(:, i)$;
8: end
9: return $C \in \mathbb{R}^{m \times s}$.
```

Appendix B

In this appendix, we provide the proofs which are used to derive the error bound of the proposed two-sided sketching algorithms in Theorems 4.1 and 4.2.

B.1. Facts about Random Tensors

First, let us state a useful formula below that allows us to compute some expectations involving a Gaussian random tensor.

Proposition 6.1 *Assume $t > q+1$. Let $\mathcal{G}_1 \in \mathbb{R}^{t\times q\times p}$ and $\mathcal{G}_2 \in \mathbb{R}^{t\times l\times p}$ be Gaussian random tensors. For any tensor $\mathcal{B}$ with conforming dimensions,*

$$\mathbb{E}\| \mathcal{G}_1^{\dagger} \star_L \mathcal{G}_2 \star_L \mathcal{B} \|_F^2 = \frac{q}{t-q-1} \| \mathcal{B} \|_F^2 \ .$$

proof By Lemma 2.3 and the linearity of the expectation, we have

$$\mathbb{E}\| \mathcal{G}_1^{\dagger} \star_L \mathcal{G}_2 \star_L \mathcal{B} \|_F^2 = \frac{1}{p}\left(\sum_{i=1}^{p} \mathbb{E}\left\| \bar{\mathcal{G}_1^{\dagger}}^{(i)} \bar{\mathcal{G}_2}^{(i)} \bar{\mathcal{B}}^{(i)} \right\|_F^2 \right) .$$

By using [34, A.1], we have

$$\mathbb{E}\left\| \bar{\mathcal{G}_1^{\dagger}}^{(i)} \bar{\mathcal{G}_2}^{(i)} \bar{\mathcal{B}}^{(i)} \right\|_F^2 = \frac{q}{t-q-1} \| \bar{\mathcal{B}}^{(i)} \|_F^2 \ ,$$

which yields

$$\mathbb{E}\| \mathcal{G}_1^{\dagger} \star_L \mathcal{G}_2 \star_L \mathcal{B} \|_F^2 = \frac{q}{t-q-1} \| \mathcal{B} \|_F^2 \ .$$

This completes the proof.

B.2. Results from Randomized Linear Algebra

Proposition 6.2 *Fix $\mathcal{A} \in \mathbb{R}^{m\times n\times p}$, let $\varrho < k$ be a natural number, and $f(\varrho,k) = \varrho/(k-\varrho-1)$. Then the tensor $\mathcal{Q} \in \mathbb{R}^{m\times k\times p}$ calculated by Eq. (3.17) satisfies*

$$\mathbb{E}_{\Omega}\|\mathcal{A} - \mathcal{Q} \star_L \mathcal{Q}^H \star_L \mathcal{A}\|_F^2 \le (1 + f(\varrho,k))\tau_{\varrho+1}^2(\mathcal{A} \star_L \mathcal{A}^H) \ . \tag{6.31}$$

An analogous result holds for the tensor $\mathcal{P} \in \mathbb{R}^{n\times k\times p}$ computed by Eq. (3.17), i.e.,

$$\mathbb{E}_{\Upsilon}\|\mathcal{A} - \mathcal{A} \star_L \mathcal{P} \star_L \mathcal{P}^H\|_F^2 \le (1 + f(\varrho,k))\tau_{\varrho+1}^2(\mathcal{A} \star_L \mathcal{A}^H) \ . \tag{6.32}$$

proof By Lemma 2.3 and the linearity of the expectation, we have

$$\mathbb{E}_{\Omega}\|\mathcal{A} - \mathcal{Q}_L \star_L \mathcal{Q}_L^H \star_L \mathcal{A}\|_F^2 = \frac{1}{p}\left(\sum_{i=1}^{p} \mathbb{E}\left\| \bar{\mathcal{A}}_L^{(i)} - \bar{\mathcal{Q}}_L^{(i)} \left(\bar{\mathcal{Q}}_L^{(i)}\right)^H \bar{\mathcal{A}}_L^{(i)} \right\|_F^2 \right) .$$

By [42, Theorem 10.5], we have

$$\mathbb{E}\|\bar{\mathcal{A}}_{F_{n_3}}^{(i)} - \bar{\mathcal{Q}}_{F_{n_3}}^{(i)} (\bar{\mathcal{Q}}_{F_{n_3}}^{(i)})^H \bar{\mathcal{A}}_{F_{n_3}}^{(i)} \|_F^2 \le (1 + f(\varrho,k)) \cdot \tau_{\varrho+1}^2(\bar{\mathcal{A}}_{F_{n_3}}^{(i)} (\bar{\mathcal{A}}_{F_{n_3}}^{(i)})^H) \ .$$

Similarly, we know from [37] that multiplying by a unitary matrix will not change the Frobenius norm, i.e., the Frobenius norm keeps the unitary matrix unchanged. Thus we have

$$\mathbb{E}\|\bar{\mathcal{A}}_L^{(i)} - \bar{\mathcal{Q}}_L^{(i)}(\bar{\mathcal{Q}}_L^{(i)})^H\bar{\mathcal{A}}_L^{(i)}\|_F^2 \leq (1+f(\varrho,k))\cdot\tau_{\varrho+1}^2(\bar{\mathcal{A}}_L^{(i)}(\bar{\mathcal{A}}_L^{(i)})^H)$$

and

$$\begin{aligned}\mathbb{E}_\Omega\|\mathcal{A} - \mathcal{Q}_L \star_L \mathcal{Q}_L^H \star_L \mathcal{A}\|_F^2 &\leq \frac{1}{p}(1+f(\varrho,k))\cdot\big(\sum_{i=1}^p \tau_{\varrho+1}^2(\bar{\mathcal{A}}_L^{(i)}(\bar{\mathcal{A}}_L^{(i)})^H)\big)\\ &= (1+f(\varrho,k))\cdot\tau_{\varrho+1}^2(\mathcal{A}\star_L\mathcal{A}^H)\ .\end{aligned}$$

Similarly, there is

$$\mathbb{E}_\Upsilon\|\mathcal{A} - \mathcal{A}\star_L\mathcal{P}\star_L\mathcal{P}^H\|_F^2 \leq (1+f(\varrho,k))\tau_{\varrho+1}^2(\mathcal{A}\star_L\mathcal{A}^H)\ , \tag{6.33}$$

which completes the proof.

Let $\hat{\mathcal{A}}$ be the tubal rank k approximation of $\mathcal{A}$ obtained by the L-Gaussian-Sketch algorithm. We now split the error $\|\mathcal{A} - \hat{\mathcal{A}}\|_F^2$ into two parts by Proposition 6.3 below.

Proposition 6.3 *Let $\hat{\mathcal{A}}$ be the tubal rank k approximation of $\mathcal{A} \in \mathbb{R}^{m\times n\times p}$ obtained by the L-Gaussian-Sketch algorithm, with $\mathcal{Q}$, $\mathcal{C}$ and $\mathcal{P}$ being the intermediate tensors obtained by the L-Gaussian-Sketch algorithm satisfying $\mathcal{Q}^H\star_L\mathcal{Q} = \mathcal{I}_{kkp}$ and $\mathcal{P}\star_L\mathcal{P}^H = \mathcal{I}_{nnp}$. Then*

$$\|\mathcal{A} - \hat{\mathcal{A}}\|_F^2 = \|\mathcal{A} - \mathcal{Q}\star_L\mathcal{Q}^H\star_L\mathcal{A}\star_L\mathcal{P}\star_L\mathcal{P}^H\|_F^2 + \|\mathcal{C} - \mathcal{Q}^H\star_L\mathcal{A}\star_L\mathcal{P}\|_F^2\ . \tag{6.34}$$

proof Since $\hat{\mathcal{A}} = \mathcal{Q}\star_L\mathcal{C}\star_L\mathcal{P}^H$, we have

$$\begin{aligned}&\|\mathcal{A} - \hat{\mathcal{A}}\|_F^2\\ =\ & \frac{1}{p}\sum_{i=1}^p \|\bar{\mathcal{A}}_L - \bar{\hat{\mathcal{A}}}_L\|_F^2\\ =\ & \frac{1}{p}\sum_{i=1}^p(\|\bar{\mathcal{A}}_L - \bar{\mathcal{Q}}_L\bar{\mathcal{Q}}_L^H\bar{\mathcal{A}}_L\bar{\mathcal{P}}_L\bar{\mathcal{P}}^H\|_F^2 + \|\bar{\mathcal{C}}_L - \bar{\mathcal{Q}}_L^H\bar{\mathcal{A}}_L\bar{\mathcal{P}}_L\|_F^2)\\ =\ & \|\mathcal{A} - \mathcal{Q}\star_L\mathcal{Q}^H\star_L\mathcal{A}\star_L\mathcal{P}\star_L\mathcal{P}^H\|_F^2 + \|\mathcal{C} - \mathcal{Q}^H\star_L\mathcal{A}\star_L\mathcal{P}\|_F^2\ .\end{aligned}$$

The first and last equations are known by Lemma 2.3, and the second equation is known by [34, A.6]. This completes the proof.

The error of the first part of Eq. (6.34) can be given in the below Theorem 6.4.

Theorem 6.4 *For any natural number $\varrho < k-1$, it holds that*

$$\mathbb{E}_\Upsilon\mathbb{E}_\Omega\|\mathcal{A} - \mathcal{Q}\star_L\mathcal{Q}^H\star_L\mathcal{A}\star_L\mathcal{P}\star_L\mathcal{P}^H\|_F^2 \leq (1+\frac{2\varrho}{k-\varrho-1})\tau_{\varrho+1}^2(\mathcal{A}\star_L\mathcal{A}^H)\ . \tag{6.35}$$

proof We have

$$\begin{aligned}
&\mathbb{E}_\Upsilon \mathbb{E}_\Omega \|\mathcal{A} - \mathcal{Q} \star_L \mathcal{Q}^H \star_L \mathcal{A} \star_L \mathcal{P} \star_L \mathcal{P}^H\|_F^2 \\
= \;& \frac{1}{p} \sum_{i=1}^{p} \mathbb{E}_\Upsilon \mathbb{E}_\Omega \|\bar{A}_L - \bar{Q}_L \bar{Q}_L^H \bar{A}_L \bar{P}_L \bar{P}^H\|_F^2 \\
\leq \;& (1 + \frac{2\varrho}{k - \varrho - 1}) \tau_{\varrho+1}^2 (\mathcal{A} \star_L \mathcal{A}^H) \; .
\end{aligned}$$

The first equation is known by Lemma 2.3, and the inequality is known by [34, A.5]. This completes the proof.

B.3. Decomposition of the Core Tensor Approximation Error

We now obtain a formula for the error in the approximation $(\mathcal{C} - \mathcal{Q}^H \star_L \mathcal{A} \star_L \mathcal{P})$ (i.e., the second part of Eq. (6.34)).

Note that the core tensor $\mathcal{C} \in \mathbb{R}^{k \times k \times p}$ is defined in Eq. (3.18), and the orthonormal tensors $\mathcal{P} \in \mathbb{R}^{n \times k \times p}$ and $\mathcal{Q} \in \mathbb{R}^{m \times k \times p}$ are constructed in Eq. (3.17). Let us introduce tensors $\mathcal{P}_\perp \in \mathbb{R}^{n \times (n-k) \times p}$ and $\mathcal{Q}_\perp \in \mathbb{R}^{m \times (m-k) \times p}$ whose ranges are complementary to those of $\mathcal{P}$ and $\mathcal{Q}$, respectively, i.e.,

$$\mathcal{P}_\perp \star_L \mathcal{P}_\perp^H = \mathcal{I}_{mmp} - \mathcal{P} \star_L \mathcal{P}^H, \tag{6.36}$$

$$\mathcal{Q}_\perp \star_L \mathcal{Q}_\perp^H = \mathcal{I}_{mmp} - \mathcal{Q} \star_L \mathcal{Q}^H, \tag{6.37}$$

where the columns of $\mathcal{P}_\perp$ and $\mathcal{Q}_\perp$ are orthonormal, separately. Next, we introduce the subtensors, i.e.,

$$\begin{aligned}
\Phi_1 := \Phi \star_L \mathcal{Q} \in \mathbb{R}^{s \times k \times p}, \quad \Phi_2 := \Phi \star_L \mathcal{Q}_\perp \in \mathbb{R}^{s \times (m-k) \times p}, \\
\Psi_1^H := \mathcal{P}^H \star_L \Psi^H \in \mathbb{R}^{k \times s \times p}, \quad \Psi_2^H := \mathcal{P}_\perp^H \star_L \Psi^H \in \mathbb{R}^{(n-k) \times s \times p}.
\end{aligned} \tag{6.38}$$

With these notations at hand, we can state and prove Lemma 6.5 below.

Lemma 6.5 *(Decomposition of the Core Tensor Approximation) Assume the tub- al rank of* Φ_1 *and* Ψ_1 *is* k *and* s*, respectively. Then*

$$\begin{aligned}
&\mathcal{C} - \mathcal{Q}^H \star_L \mathcal{A} \star_L \mathcal{P} \\
= \;& \Phi_1^\dagger \star_L \Phi_2 \star_L (\mathcal{Q}_\perp^H \star_L \mathcal{A} \star_L \mathcal{P}) + (\mathcal{Q}^H \star_L \mathcal{A} \star_L \mathcal{P}_\perp) \star_L \Psi_2^\dagger \star_L (\Psi_1^\dagger)^H \\
&+ \Phi_1^\dagger \star_L \Phi_2 \star_L (\mathcal{Q}_\perp^H \star_L \mathcal{A} \star_L \mathcal{P}_\perp) \star_L \Psi_2^\dagger \star_L (\Psi_1^\dagger)^H .
\end{aligned}$$

proof Adding and subtracting terms, we write the core sketch $\mathcal{Z}$ as

$$\begin{aligned}
\mathcal{Z} &= \Phi \star_L \mathcal{A} \star_L \Psi^H \\
&= \Phi \star_L (\mathcal{A} - \mathcal{Q} \star_L \mathcal{Q}^H \star_L \mathcal{A} \star_L \mathcal{P} \star_L \mathcal{P}^H) \star_L \Psi^H \\
&\quad + (\Phi \star_L \mathcal{Q}) \star_L (\mathcal{Q}^H \star_L \mathcal{A} \star_L \mathcal{P}) \star_L (\mathcal{P}^H \star_L \Psi^H).
\end{aligned}$$

Using Eq. (6.38), we identify the tensors Φ_1 and Ψ_1. For the above $\mathcal{Z}$, after left-multiplying it by $\Phi_1^\dagger$ and right-multiplying it by $(\Psi_1^\dagger)^H$, we have

$$\begin{aligned}\mathcal{C} =\ & \Phi_1^\dagger \star_L \mathcal{Z} \star_L (\Psi_1^\dagger)^H \\ =\ & \Phi_1^\dagger \star_L \Phi \star_L (\mathcal{A} - \mathcal{Q} \star_L \mathcal{Q}^H \star_L \mathcal{A} \star_L \mathcal{P} \star_L \mathcal{P}^H) \star_L \Psi^H \star_L (\Psi_1^\dagger)^H \\ & + \mathcal{Q}^H \star_L \mathcal{A} \star_L \mathcal{P},\end{aligned}$$

which identifies the core tensor $\mathcal{C}$ defined in (3.18). For the above representation of $\mathcal{C}$, moving the term $\mathcal{Q}^H \star_L \mathcal{A} \star_L \mathcal{P}$ to the left-hand side will give the approximation error. Using Eq. (6.38) again, we have

$$\begin{aligned}\Phi_1^\dagger \star_L \Phi &= \Phi_1^\dagger \star_L \Phi \star_L \mathcal{Q} \star_L \mathcal{Q}^H + \Phi_1^\dagger \star_L \Phi \star_L \mathcal{Q}_\perp \star_L \mathcal{Q}_\perp^H \\ &= \mathcal{Q}^H + \Phi_1^\dagger \star_L \Phi_2 \star_L \mathcal{Q}_\perp^H\ ,\end{aligned}$$

and

$$\begin{aligned}\Psi^H \star_L (\Psi_1^\dagger)^H &= \mathcal{P} \star_L \mathcal{P}^H \star_L \Psi^H \star_L (\Psi_1^\dagger)^H + \mathcal{P}_\perp \star_L \mathcal{P}_\perp^H \star_L \Psi^H \star_L (\Psi_1^\dagger)^H \\ &= \mathcal{P} + \mathcal{P}_\perp \star_L \Psi_2^H \star_L (\Psi_1^\dagger)^H.\end{aligned}$$

Combining the last three displays, we have

$$\begin{aligned}& \mathcal{C} - \mathcal{Q}^H \star_L \mathcal{A} \star_L \mathcal{P} \\ =\ & (\mathcal{Q}^H + \Phi_1^\dagger \star_L \Phi_2 \star_L \mathcal{Q}_\perp^H) \star_L (\mathcal{A} - \mathcal{Q} \star_L \mathcal{Q}^H \star_L \mathcal{A} \star_L \mathcal{P} \star_L \mathcal{P}^H) \star_L (\mathcal{P} \\ & + \mathcal{P}_\perp \star_L \Psi_2^H \star_L (\Psi_1^\dagger)^H).\end{aligned}$$

Expanding the expression and using the orthogonality relations $\mathcal{Q}^H \star_L \mathcal{Q} = \mathcal{I}_{kkp}$, $\mathcal{Q}_\perp^H \star_L \mathcal{Q} = \mathcal{O}$, $\mathcal{P}^H \star_L \mathcal{P} = \mathcal{I}_{kkp}$, and $\mathcal{P}_\perp^H \star_L \mathcal{P} = \mathcal{O}$, we complete the proof.

B.4. Probabilistic Analysis of the Core Tensor

We can then study the probabilistic behavior of the error $(\mathcal{C} - \mathcal{Q}^H \star_L \mathcal{A} \star_L \mathcal{P})$, conditional on $\mathcal{Q}$ and $\mathcal{P}$.

Lemma 6.6 *(Probabilistic Analysis of the Core Tensor) Assume that the dimension reduction tensors Φ and Ψ are Gaussian linear sketching operators. When $s \geq k$, it holds that*

$$\mathbb{E}_{\Phi,\Psi}[\mathcal{C} - \mathcal{Q}^H \star_L \mathcal{A} \star_L \mathcal{P}] = \mathcal{O}. \tag{6.39}$$

When $s > k + 1$, the error can be expressed as

$$\begin{aligned}\mathbb{E}_{\Phi,\Psi}\|\mathcal{C} - \mathcal{Q}^T \star_L \mathcal{A} \star_L \mathcal{P}\|_F^2 &= \frac{k}{s-k-1}\|\mathcal{A} - \mathcal{Q} \star_L \mathcal{Q}^H \star_L \mathcal{A} \star_L \mathcal{P} \star_L \mathcal{P}^H\|_F^2 \\ &\quad + \frac{k(2k+1-s)}{(s-k-1)^2}\|\mathcal{Q}_\perp^H \star_L \mathcal{A} \star_L \mathcal{P}_\perp\|_F^2\ .\end{aligned}$$

In particular, when $s < 2k+1$, the last term is nonnegative; and when $s \geq 2k+1$, the last term is nonpositive.

proof Since Φ is a Gaussian linear sketching operator, the orthogonal subtensors Φ_1 and Φ_2 are also Gaussian linear sketching operators because of the marginal property of the normal distribution. Likewise, Ψ_1 and Ψ_2 are Gaussian linear sketching operators. Provided that $s \geq k$, the tensor transformed tubal rank of Φ_1 and Ψ_1 is k. Using the decomposition of the approximation error from Lemma 6.5, we have

$$\begin{aligned}\mathbb{E}_{\Phi,\Psi}[\mathcal{C} - \mathcal{Q}^H \star_L \mathcal{A} \star_L \mathcal{P}] &= \mathbb{E}_{\Phi_1}\mathbb{E}_{\Phi_2}[\Phi_1^\dagger \star_L \Phi_2 \star_L (\mathcal{Q}_\perp^H \star_L \mathcal{A} \star_L \mathcal{P})] \\ &\quad + \mathbb{E}_{\Psi_1}\mathbb{E}_{\Psi_2}[(\mathcal{Q}^H \star_L \mathcal{A} \star_L \mathcal{P}_\perp) \star_L \Psi_2^H \star_L (\Psi_1^\dagger)^H] \\ &\quad + \mathbb{E}[\Phi_1^\dagger \star_L \Phi_2 \star_L (\mathcal{Q}_\perp^H \star_L \mathcal{A} \star_L \mathcal{P}_\perp) \star_L \Psi_2^H \star_L (\Psi_1^\dagger)^H] \ .\end{aligned}$$

Then we invoke independence to write the expectations as iterated expectations. Since Φ_2 and Ψ_2 have mean zero. This formula makes it clear that the approximation error has mean zero.

To study the fluctuations, applying the independence and zero-mean property of Φ_2 and Ψ_2, we have

$$\begin{aligned}&\mathbb{E}_{\Phi,\Psi}\|\mathcal{C} - \mathcal{Q}^H \star_L \mathcal{A} \star_L \mathcal{P}\|_F^2 \\ &= \mathbb{E}_{\Phi}\|\Phi_1^\dagger \star_L \Phi_2 \star_L (\mathcal{Q}_\perp^H \star_L \mathcal{A} \star_L \mathcal{P})\|_F^2 \\ &\quad + \mathbb{E}_{\Psi}\|(\mathcal{Q}^H \star_L \mathcal{A} \star_L \mathcal{P}_\perp) \star_L \Psi_2^H \star_L (\Psi_1^\dagger)^H\|_F^2 \\ &\quad + \mathbb{E}_{\Phi}\mathbb{E}_{\Psi}\|\Phi_1^\dagger \star_L \Phi_2 \star_L (\mathcal{Q}_\perp^H \star_L \mathcal{A} \star_L \mathcal{P}_\perp) \star_L \Psi_2^H \star_L (\Psi_1^\dagger)^H\|_F^2 \ .\end{aligned}$$

Invoking Proposition 6.1 yields

$$\begin{aligned}&\mathbb{E}_{\Phi,\Psi}\|\mathcal{C} - \mathcal{Q}^H \star_L \mathcal{A} \star_L \mathcal{P}\|_F^2 \\ &= \frac{k}{s-k-1}\Big[\|\mathcal{Q}_\perp^H \star_L \mathcal{A} \star_L \mathcal{P}\|_F^2 + \|\mathcal{Q}^H \star_L \mathcal{A} \star_L \mathcal{P}_\perp\|_F^2 \\ &\quad + \frac{k}{s-k-1}\|\mathcal{Q}_\perp^H \star_L \mathcal{A} \star_L \mathcal{P}_\perp\|_F^2\Big] \\ &= \frac{k}{s-k-1}\Big[\|\mathcal{Q}_\perp^H \star_L \mathcal{A} \star_L \mathcal{P}\|_F^2 + \|\mathcal{Q}^H \star_L \mathcal{A} \star_L \mathcal{P}_\perp\|_F^2 + \|\mathcal{Q}_\perp^H \star_L \mathcal{A} \star_L \mathcal{P}_\perp\|_F^2 \\ &\quad + \frac{2k+1-s}{s-k-1}\|\mathcal{Q}_\perp^H \star_L \mathcal{A} \star_L \mathcal{P}_\perp\|_F^2\Big] \ .\end{aligned}$$

Using the Pythagorean Theorem to combine the terms in the above equation, we have

$$\begin{aligned}&\mathbb{E}_{\Phi,\Psi}\|\mathcal{C} - \mathcal{Q}^H \star_L \mathcal{A} \star_L \mathcal{P}\|_F^2 \\ &= \frac{k}{s-k-1}\|\mathcal{A} - \mathcal{Q} \star_L \mathcal{Q}^H \star_L \mathcal{A} \star_L \mathcal{P} \star_L \mathcal{P}^H\|_F^2 \\ &\quad + \frac{k(2k+1-s)}{(s-k-1)^2}\|\mathcal{Q}_\perp^H \star_L \mathcal{A} \star_L \mathcal{P}_\perp\|_F^2 \ .\end{aligned}$$

This completes the proof.

References

[1] T. Jiang, T. Huang, X. Zhao, L. Deng, and Y. Wang. A novel tensor-based video rain streaks removal approach via utilizing discriminatively intrinsic priors. Conference on Computer Vision and Pattern Recognition, pp. 4057–4066, 2017.

[2] T. Kim, S. Wong, and R. Cipolla. Tensor canonical correlation analysis for action classification. Conference on Computer Vision and Pattern Recognition, pp. 1–8, 2007.

[3] B. Du, M. Zhang, L. Zhang, R. Hu, and D. Tao. Pltd: Patch-based low-rank tensor decomposition for hyperspectral images. IEEE Transactions on Multimedia, 19(1):67–79, 2016.

[4] N. Renard, S. Bourennane, and J. Blanc-Talon. Denoising and dimensionality reduction using multilinear tools for hyperspectral images. IEEE Geoscience and Remote Sensing Letters, 5(2):138–142, 2008.

[5] W. Wu, T. Huang, X. Zhao, et al. Hyperspectral image denoising via tensor low-rank prior and unsupervised deep spatial–spectral prior. IEEE Transactions on Geoscience and Remote Sensing, 60:1–14, 2022.

[6] J. Wang, T. Huang, X. Zhao, et al. CoNoT: coupled nonlinear transform-based low-rank tensor representation for multidimensional image completion. IEEE Transactions on Neural Networks and Learning Systems, 35(7):8969–8983, 2024.

[7] L. De Lathauwer. Signal processing based on multilinear algebra. Ph.D. dissertation, Katholieke Universiteit Leuven Leuven, 1997.

[8] P. Comon. Tensor decompositions, state of the art and applications. Mathematics in Signal Processing, pp. 1–24, 2002.

[9] E. Papalexakis, K. Pelechrinis, and C. Faloutsos. Spotting misbehaviors in location-based social networks using tensors. Proceedings of the 23rd International Conference on World Wide Web, pp. 551–552, 2014.

[10] M. Nakatsuji, Q. Zhang, X. Lu, B. Makni, and J. A. Hendler. Semantic social network analysis by cross-domain tensor factorization. IEEE Transactions on Computational Social Systems, 4(4):207–217, 2017.

[11] Y. Liu, J. Liu, Z. Long, and C. Zhu. Tensor Sketch[M]//Tensor Computation for Data Analysis. Springer, Cham, pp. 299–321, 2022.

[12] X. Cao, X. Zhang, C. Zhu C, et al. TS-RTPM-Net: Data-driven tensor sketching for efficient CP decomposition. IEEE Transactions on Big Data, 10(1):1–11, 2024.

[13] X. Li, J. Haupt, and D. Woodruff. Near optimal sketching of low-rank tensor regression. Advances in Neural Information Processing Systems, 30, 2017.

[14] Y. Wang, H. Tung, A. Smola, et al. Fast and guaranteed tensor decomposition via sketching. Advances in neural information processing systems, 28, 2015.

[15] M. Che and Y. Wei. Randomized algorithms for the approximations of Tucker and the tensor train decompositions. Adv. in Comput. Math, 45:395–428, 2019.

[16] M. Che, Y. Wei, and H. Yan. Randomized algorithms for the low multilinear rank approximations of tensors. Journal of Computational and Applied Mathematics, 390(2):113380, 2021.

[17] M. Che, Y. Wei, and H. Yan. An efficient randomized algorithm for computing the approximate Tucker decomposition. Journal of Scientific Computing, DOI: https://doi.org/10.1007/s10915-021-01545-5, 2021.

[18] O. Malik and S. Becker. Low-rank tucker decomposition of large tensors using tensor sketch. Advances in neural information processing systems, 31, 2018.

[19] J. Ravishankar, M. Sharma, and S. Khaidem. A novel compression scheme based on hybrid Tucker-vector quantization via tensor sketching for dynamic light fields acquired through coded aperture camera. IEEE International Conference on 3D Immersion (IC3D), pp. 1–8, 2021.

[20] Y. Sun, Y. Guo, C. Luo, et al. Low-rank Tucker approximation of a tensor from streaming data. SIAM Journal on Mathematics of Data Science, 2(4):1123–1150, 2020.

[21] Y. Hur, J. Hoskins, M. Lindsey, et al. Generative modeling via tensor train sketching. arXiv preprint arXiv:2202.11788, 2022.

[22] L. Qi and G. Yu, T-singular values and T-Sketching for third order tensors. arXiv:2013.00976, 2021.

[23] R. Minster, A. Saibaba, and M. Kilmer. Randomized algorithms for low-rank tensor decompositions in the Tucker format. SIAM Journal on Mathematics of Data Science, 2(1):189–215, 2020.

[24] W. Dong, G. Yu, L. Qi, and X. Cai. Practical sketching algorithms for low-rank Tucker approximation of large tensors. Journal of Scientific Computing, 95(2):52, 2023.

[25] H. Diao, R. Jayaram, Z. Song, W. Sun, and D. Woodruff. Optimal sketching for kronecker product regression and low rank approximation. Advances in neural information processing systems, 32, 2019.

[26] I. Han, H. Avron, and J. Shin. Polynomial tensor sketch for element-wise function of low-rank matrix. International Conference on Machine Learning, pp. 3984–3993, 2020.

[27] S. Kasiviswanathan, N. Narodytska, and H. Jin. Network approximation using tensor sketching. IJCAI, pp. 2319–2325, 2018.

[28] M. Ma, Q. Zhang, J. Ho, and L. Xiong. Spatio-temporal tensor sketching via adaptive sampling. Joint European Conference on Machine Learning and Knowledge Discovery in Databases, pp. 490–506, 2020.

[29] Y. Shi and A. Anandkumar A. Higher-order count sketch: Dimensionality reduction that retains efficient tensor operations. arXiv preprint arXiv:1901.11261, 2019.

[30] B. Yang, A. Zamzam, N. Sidiropoulos. Parasketch: Parallel tensor factorization via sketching. SIAM International Conference on Data Mining. pp. 396–404, 2018.

[31] Y. Carmon, J. Duchi, S. Aaron, et al. A rank-1 sketch for matrix multiplicative weights. Conference on Learning Theory, pp. 589–623, 2019.

[32] D. Woodruff. Sketching as a tool for numerical linear algebra. Foundations and Trends in Theoretical Computer Science, 10(12):1–157, 2014.

[33] J. Tropp, A. Yurtsever, M. Udell, and V. Cevher, Practical sketching algorithms for low-rank matrix approximation. SIAM Journal on Matrix Analysis and Applications, 38:1454–1485, 2017.

[34] J. Tropp, A. Yurtsever, M. Udell, and V. Cevher, Streaming low-rank matrix approximation with an application to scientific simulation. SIAM J. Sci. Comput., 41:A2430–A2463, 2019.

[35] D. Kressner, B. Vandereycken, and R. Voorhaar. Streaming tensor train approximation. arXiv preprint arXiv:2208.02600, 2022.

[36] M. Che, Y. Wei, and Y. Xu. Randomized algorithms for the computation of multilinear rank-(μ_1, μ_2, μ_3) approximations. Journal of Global Optimization, pp. 1–31, 2022.

[37] E. Kernfeld, M. Kilmer, and S. Aeron. Tensor-tensor products with invertible linear transforms. Linear Algebra and Its Applications, 485:545–570, 2015.

[38] M. Kilmer and C. Martin. Factorization strategies for third-order tensors. Linear Algebra and Its Applications, 435:641–658, 2011.

[39] G. Song, M. Ng, and X. Zhang. Robust tensor completion using transformed tensor singular value decomposition. Numerical Linear Algebra with Applications, 27(3):e2299, 2020.

[40] J. Zhang, A. Saibaba, M. Kilmer, and S. Aeron. A randomized tensor singular value decomposition based on the t-product. Numerical Linear Algebra with Applications, 25:e2179, 2018.

[41] M. Kilmer, C.D. Martin, and L. Perrone, A third-order generalization of the matrix SVD as a product of third-order tensors. Tech. Report TR-2008-4 Tufts University, Computer Science Department, 2008.

[42] N. Halko, P. Martinsson, and J. Tropp. Find structure with randomness: Probablistic algorithms for constructing approximate matrix decompositions. SIAM Review, 53:217–288, 2011.

[43] S. Wang. A practical guide to randomized matrix computations with MATLAB implementations. arXiv preprint arXiv:1505.07570, 2015.