Tensor Krylov subspace methods with an invertible linear transform product applied to image processing

Lothar Reichel* and Ugochukwu O. Ugwu†

Department of Mathematical Sciences, Kent State University, OH 44242, USA

Abstract

This paper discusses several transform-based methods for solving linear discrete ill-posed problems for third order tensor equations based on a tensor-tensor product defined by an invertible linear transform. Linear transform-based tensor-tensor products were first introduced in [E. Kernfeld, M. Kilmer, and S. Aeron, Tensor-tensor products with invertible linear transform, Linear Algebra and its Applications, 485 (2015), pp. 545–570]. These tensor-tensor products are applied to derive Tikhonov regularization methods based on Golub-Kahan-type bidiagonalization and Arnoldi-type processes. GMRES-type solution methods based on the latter process also are described. By applying only a fairly small number of steps of these processes, large-scale problems are reduced to problems of small size. The number of steps required by these processes and the regularization parameter are determined by the discrepancy principle. The data tensor is a general third order tensor or a tensor defined by a laterally oriented matrix. A quite general regularization tensor can be applied in Tikhonov regularization. Applications to color image and video restorations illustrate the effectiveness of the proposed methods.

Key words: discrepancy principle, invertible linear transform, linear discrete ill-posed problem, tensor Arnoldi process, tensor bidiagonalization process, tensor Tikhonov regularization.

1 Introduction

We are concerned with the solution of large-scale least squares problem of the form

$$\min_{\mathcal{X}\in\mathbb{R}^{m\times p\times n}} \|\mathcal{A} *_L \mathcal{X} - \mathcal{B}\|_F, \qquad \mathcal{A}\in\mathbb{R}^{\ell\times m\times n}, \qquad \mathcal{B}\in\mathbb{R}^{\ell\times p\times n}, \quad p>1, \tag{1.1}$$

with a third order tensor $\mathcal{A}$, whose singular tubes decay rapidly in the Frobenius norm with increasing index. In particular, $\mathcal{A}$ has ill-determined tubal rank. Many of its singular tubes, which are analogues of the singular values of a matrix, are nonvanishing with tiny Frobenius norm of different orders of magnitude. This makes (1.1) a linear discrete ill-posed problem; cf. Definition 2.2 below, in which the tensor $\mathcal{A}$ specifies the model, the tensor $\mathcal{B}$ represents available data, e.g., a degraded color image, and the operator $*_L$ is a tensor-tensor product defined in a transformed domain for an invertible linear operator L. The $*_L$ product between $\mathcal{A}$ and $\mathcal{X}$ is computed by moving both tensors into the transform domain, evaluating n matrix-matrix products in the transform domain, and computing the inverse transform of the result; cf. Definition 2.1 below. This kind of tensor-tensor product was first described by Kernfeld et al. [16] and has found applications in data compression [18], tensor neural networks [25], image deblurring [16], as well as image recovery by low-rank completion [22]. An extension of the $*_L$ product to k-order tensors for $k>3$ is described by Han [11]. Further details on the operator $*_L$ will be provided in Section 2. Throughout this paper, $\|\cdot\|_F$ denotes the Frobenius norm. For $\mathcal{A}=[a_{ijk}]_{i,j,k=1}^{\ell,m,n}$, we have

$$\|\mathcal{A}\|_F = \sqrt{\sum_{i=1}^{\ell}\sum_{j=1}^{m}\sum_{k=1}^{n} a_{ijk}^2}.$$

* e-mail: reichel@math.kent.edu
† e-mail: uugwu@kent.edu

In applications of interest to us, such as image and video processing, the data tensor $\mathcal{B}$ represents measured data that are contaminated by a measurement error $\mathcal{E} \in \mathbb{R}^{\ell\times p\times n}$, i.e.,

$$\mathcal{B} = \mathcal{B}_{\text{true}} + \mathcal{E},$$

where $\mathcal{B}_{\text{true}} \in \mathbb{R}^{\ell\times p\times n}$ denotes the unknown error-free data tensor. We will assume that the unavailable system of equations

$$\mathcal{A} *_L \mathcal{X} = \mathcal{B}_{\text{true}} \tag{1.2}$$

is consistent and let $\mathcal{X}_{\text{true}} \in \mathbb{R}^{m\times p\times n}$ denote its solution of minimal Frobenius norm. Our aim is to determine an accurate approximation of $\mathcal{X}_{\text{true}}$ given $\mathcal{A}$ and $\mathcal{B}$ in (1.1). The consistency of (1.2) makes it possible to apply the discrepancy principle for this purpose; see below.

Straightforward solution of (1.1) generally does not yield a meaningful approximation of $\mathcal{X}_{\text{true}}$ due to propagation and severe amplification of the error $\mathcal{E}$ in $\mathcal{B}$ into the solution of (1.1). We introduce Tikhonov regularization to reduce this difficulty, i.e., instead of solving (1.1), we solve the penalized least squares problem

$$\min_{\mathcal{X}\in\mathbb{R}^{m\times p\times n}} \left\{ \|\mathcal{A} *_L \mathcal{X} - \mathcal{B}\|_F^2 + \mu^{-1} \|\mathcal{L} *_L \mathcal{X}\|_F^2 \right\}. \tag{1.3}$$

The tensor $\mathcal{L} \in \mathbb{R}^{s\times m\times n}$ is a regularization operator and $\mu > 0$ is a regularization parameter. Let $\mathcal{N}(\mathcal{M})$ denote the null space of the tensor $\mathcal{M}$ under $*_L$, and assume that $\mathcal{L}$ is such that

$$\mathcal{N}(\mathcal{A}) \cap \mathcal{N}(\mathcal{L}) = \{\mathcal{O}\},$$

where $\mathcal{O} \in \mathbb{R}^{m\times p\times n}$ denotes the null tensor. Then (1.3) has a unique solution, $\mathcal{X}_\mu$, for any $\mu > 0$.

We will use the discrepancy principle to determine the regularization parameter. Its application requires that a bound

$$\|\mathcal{E}\|_F \leq \delta \tag{1.4}$$

is known. The discrepancy principle prescribes that $\mu > 0$ be determined so that the solution $\mathcal{X}_\mu$ of (1.3) satisfies the equation

$$\|\mathcal{A} *_L \mathcal{X}_\mu - \mathcal{B}\|_F = \eta\delta, \tag{1.5}$$

where $\eta > 1$ is a user-specified constant that is independent of $\delta > 0$; see Engl et al. [8] for discussions on this approach to determine the regularization parameter μ. We remark that other techniques, such as generalized cross validation [9, 10] and the L-curve criterion [13, 20, 26], also may be used to determine the regularization parameter, in particular, when a bound (1.4) for the error tensor is not known.

We also will discuss the approximate solution of (1.1) by GMRES-type iterative methods when $\mathcal{A} \in \mathbb{R}^{m\times m\times n}$ and $\mathcal{B} \in \mathbb{R}^{m\times p\times n}$. Regularization is achieved by truncating the iterations sufficiently early. The discrepancy principle is used to decide how many iterations to carry out. The GMRES method for the solution of linear discrete ill-posed problems (1.1), when $\mathcal{A}$ is a square matrix and $\mathcal{B}$ a vector, was first described in [4] and is more recently investigated by Neubauer [23]. A variant of the GMRES solution method is discussed in [24].

This paper focuses on tensor-tensor products defined with an invertible linear transform L. These products were introduced by Kernfeld et al. [16] and are denoted by $*_L$. The tensor t-product defined in the seminal work by Kilmer and Martin [19] is a special case of the $*_L$ product. A disadvantage of the t-product is that its efficient evaluation requires the use of the discrete Fourier transform (DFT), whose implementation demands complex arithmetic. The $*_L$ product can be chosen so that no complex arithmetic is required, which may speed up the computations. Moreover, we can use linear transformations that satisfy reflective or periodic boundary conditions when this is appropriate for the problem being solved.

It is the purpose of the present paper to generalize the Arnoldi-type and bidiagonalization-type solution methods for (1.3), that are based on the t-product and are described in [7, 27, 28], to solution methods that use the $*_L$ tensor product defined by an invertible linear transform L. This generalization allows us to consider applications of the $*_L$ product in several contexts, e.g., in image and video processing, and to gain useful insights into the performance of these methods. Iterative Krylov subspace methods defined by the $*_L$ product for image and video processing have so far not received much attention in the literature. The discussion of the current paper builds on the image deblurring model considered by Kernfeld et al. [16].

The methods of this paper differ from the ones described in [7, 27, 28] in two ways:

i) We modify the generalized global t-product Golub-Kahan bidiagonalization (GG-tGKB) process described in [27] to generate an orthogonal tensor basis for the tensor Krylov (t-Krylov) subspace

$$\mathbb{K}_k(\mathcal{A}^T *_L \mathcal{A}, \mathcal{A}^T *_L \mathcal{B}) = \text{span}\{\mathcal{A}^T *_L \mathcal{B}, (\mathcal{A}^T *_L \mathcal{A}) *_L \mathcal{A}^T *_L \mathcal{B}, \ldots, (\mathcal{A}^T *_L \mathcal{A})^{k-1} *_L \mathcal{A}^T *_L \mathcal{B}\}$$

using the $*_L$ tensor product, where the superscript ^{T} denotes transposition. We refer to the method for generating an orthogonal tensor basis for this t-Krylov subspace as the GG-LtGKB process. This process reduces $\mathcal{A}$ in (1.3) to a small lower bidiagonal matrix by computing a few, say $1 \leq k \ll \min\{\ell, m\}$, steps of the GG-LtGKB process. Each step requires the evaluation of two tensor-tensor products, one with $\mathcal{A}$ and one with $\mathcal{A}^T$. We refer to the solution method for (1.3) based on the GG-LtGKB process as the generalized global $*_L$ tensor Golub-Kahan-Tikhonov (GG-LtGKT) method.

ii) We adjust the T-global Arnoldi process described by El Guide et al. [7] to generate a t-Krylov subspace under the $*_L$ tensor product. Generically, $\ell - 1$ steps of this process determine an orthogonal tensor basis for the t-Krylov subspace

$$\mathbb{K}_\ell(\mathcal{A}, \mathcal{B}) = \text{span}\{\mathcal{B}, \mathcal{A} *_L \mathcal{B}, \ldots, \mathcal{A}^{\ell-1} *_L \mathcal{B}\}. \tag{1.6}$$

Here $\mathcal{A} \in \mathbb{R}^{m\times m\times n}$. The method for determining this tensor basis will be referred to as the generalized global $*_L$ tensor Arnoldi (GG-LtA) process. We assume that $\ell \ll m$. Then, generically, this process reduces the tensor $\mathcal{A}$ in (1.3) to a small $(\ell+1)\times\ell$ upper Hessenberg matrix. Each step requires one tensor-tensor product evaluation (with $\mathcal{A}$). The subspace (1.6) will be applied both in a Tikhonov-type regularization method, which will be referred to as a generalized global $*_L$ tensor Arnoldi-Tikhonov (GG-LtAT) method and in a GMRES-type method. The latter method extends the generalized global t-product GMRES (GG-tGMRES) method recently described in [28] to the $*_L$ tensor product, and will be referred to as the GG-LtGMRES method. Generically, the ℓth iterate determined by this method, $\mathcal{X}_\ell \in \mathbb{K}_\ell(\mathcal{A}, \mathcal{B})$, satisfies

$$\|\mathcal{A} *_L \mathcal{X}_\ell - \mathcal{B}\|_F = \min_{\mathcal{X}\in\mathbb{K}_\ell(\mathcal{A},\mathcal{B})} \|\mathcal{A} *_L \mathcal{X} - \mathcal{B}\|_F, \quad \ell = 1, 2, \ldots, \tag{1.7}$$

where we assume that $\mathcal{X}_0 = \mathcal{O}$, and $\mathcal{O}$ denotes the null tensor. The iterations are terminated by the discrepancy principle, i.e., as soon as the left-hand side of (1.7) is bounded by $\eta\delta$; cf. (1.5).

We also discuss the approximate solution of linear discrete ill-posed problems of the form

$$\min_{\vec{\mathcal{X}}\in\mathbb{R}^{m\times 1\times n}} \left\{ \|\mathcal{A} *_L \vec{\mathcal{X}} - \vec{\mathcal{B}}\|_F^2 + \mu^{-1}\|\mathcal{L} *_L \vec{\mathcal{X}}\|_F^2 \right\}, \tag{1.8}$$

which are obtained when $p = 1$ in (1.1). In this situation, the tensors $\vec{\mathcal{X}} \in \mathbb{R}^{m\times 1\times n}$ and $\vec{\mathcal{B}} \in \mathbb{R}^{\ell\times 1\times n}$ are laterally oriented matrices. Here the data tensor $\vec{\mathcal{B}}$ may represent a degraded laterally oriented gray-scale image. The problem (1.8) has recently been considered in [17, 27, 28] in the special case when $*_L$ is the t-product.

Kernfeld et al. [16] described a solution method for the minimization problem (1.8) with $\mathcal{L}$ the identity tensor, using the discrete cosine transform product, denoted by $*_c$. This product can be computed by the MATLAB function `dct` along the third dimension. The product $*_c$ is a special case of the $*_L$ tensor product. Kernfeld et al. [16] also discuss the tensor product $*_L$, but its performance in the context of solving (1.8) is not considered. Our solution methods for (1.8) differ from the one described by Kernfeld et al. [16] in the following ways: i) We use the global $*_L$ tensor Golub-Kahan bidiagonalization (G-LtGKB) process, ii) we use the global $*_L$ tensor Arnoldi (G-LtA) process, iii) we allow a general regularization operator $\mathcal{L}$ in (1.8), and iv) we determine the regularization parameter with the aid of the discrepancy principle. Further discussions on these topics are provided in Subsections 3.2 and 4.2. Solution methods for (1.8) that are based on the G-LtGKB and G-LtA processes will be referred to as the global $*_L$ tensor Golub-Kahan-Tikhonov (G-LtGKT) and global $*_L$ tensor Arnoldi-Tikhonov (G-LtAT) methods, respectively.

Replacing the data tensor $\mathcal{B}$ in (1.7) by a lateral tensor slice $\vec{\mathcal{B}}$ yields the minimization problems

$$\|\mathcal{A} *_L \vec{\mathcal{X}}_\ell - \vec{\mathcal{B}}\|_F = \min_{\vec{\mathcal{X}}\in\mathbb{K}_\ell(\mathcal{A},\vec{\mathcal{B}})} \|\mathcal{A} *_L \vec{\mathcal{X}} - \vec{\mathcal{B}}\|_F, \quad \ell = 1, 2, \ldots, \tag{1.9}$$

where $\vec{\mathcal{X}}_\ell \in \mathbb{K}_\ell(\mathcal{A}, \vec{\mathcal{B}})$ and $\vec{\mathcal{X}}_0 = \vec{\mathcal{O}} \in \mathbb{R}^{m\times 1\times n}$. We solve (1.9) by a GMRES-type method, which we refer to as the global $*_L$ tensor GMRES (G-LtGMRES) method. Here, $\vec{\mathcal{O}}$ is a tensor determined by an $m \times n$ zero matrix oriented laterally.

We remark that several tensor-based methods, that do not apply the transform-based methods discussed in [7, 16, 17, 27, 28], recently have been described in literature; see, e.g., [2, 3, 6]. The solution schemes of the current paper belong the GKT_BTF and AT_BTF families of methods described by Beik et al. [2, 3]. They involve flattening since they transform the equations (1.3), (1.7), (1.8), and (1.9) to equivalent equations involving matrices and vectors, and they require additional product definitions to the $*_L$ product.

This paper is organized as follows. Section 2 introduces notation and preliminaries associated with the $*_L$ product formalism, and Section 3 describes the GG-LtGKT and G-LtGKT methods. Both methods use a bidiagonalization process to reduce $\mathcal{A} \in \mathbb{R}^{\ell\times m\times n}$ to a small bidiagonal matrix. Section 4 describes the GG-LtAT, G-LtAT, GG-LtGMRES, and G-LtGMRES methods. They are based on reducing $\mathcal{A} \in \mathbb{R}^{m\times m\times n}$ to a small upper Hessenberg matrix by carrying out a few steps of an Arnoldi-type process. The solution methods for (1.3) discussed in Section 3 and 4 can be divided into two groups: those that work with lateral slices $\vec{\mathcal{B}}_j$, $j = 1, 2, \ldots, p$, of the data tensor $\mathcal{B}$ independently, and those that work with these slices simultaneously. When applied to image restoration problems, computed examples in [27, 28] showed the latter approach to require less CPU time than the former, but often gave less accurate restorations. Section 5 presents computed examples that illustrate the performance of the methods described in this paper. Concluding remarks can be found in Section 6.

2 Notation and preliminaries

The tensors in this paper are multidimensional arrays of real scalars of order three. We use notation described in [7, 16, 21]. Thus, third order tensors are denoted by calligraphic script letters, say $\mathcal{A}$, capital letter, say A, stand for matrices, and boldface lower case letters, say $\mathbf{a}$, denote tubal scalars (*tube fibers*). A tube fiber of a third order tensor is a 1D section obtained by fixing two of the indices of $\mathcal{A}$ [21]. Using MATLAB notation, $\mathcal{A}(:, j, k)$, $\mathcal{A}(i, :, k)$, and $\mathcal{A}(i, j, :)$ denote mode-1, mode-2, and mode-3 tube fibers, respectively. A *slice* of a tensor $\mathcal{A}$ is a 2D section obtained by fixing one of the indices [21]. Using MATLAB notation, $\mathcal{A}(i, :, :)$, $\mathcal{A}(:, j, :)$, and $\mathcal{A}(:, :, k)$ stand for the ith horizontal, jth lateral, and kth frontal slices, respectively. The jth lateral slice, also denoted by $\vec{\mathcal{A}}_j$, is a tensor, that sometimes is referred to as a tensor column, whereas the kth frontal slice, oftentimes denoted by $\mathcal{A}^{(k)}$, is a matrix. The tensor-tensor product based on an invertible linear transform L is defined as follows:

Definition 2.1. ($*_L$ product [16]) Let $L : \mathbb{R}^{\ell\times m\times n} \to \mathbb{R}^{\ell\times m\times n}$ be an invertible linear operator, and let $\mathcal{B} \in \mathbb{R}^{\ell\times p\times n}$ and $\mathcal{C} \in \mathbb{R}^{p\times m\times n}$. Then the $*_L$ product of the tensors $\mathcal{B}$ and $\mathcal{C}$ is the tensor $\mathcal{A} \in \mathbb{R}^{\ell\times m\times n}$ given by

$$\mathcal{A} := \mathcal{B} *_L \mathcal{C} = L^{-1}\big(L(\mathcal{B})\triangle L(\mathcal{C})\big), \tag{2.1}$$

where the `face-product` $\triangle$ is defined by

$$(\mathcal{A}\triangle\mathcal{B})^{(i)} = \mathcal{A}^{(i)}\mathcal{B}^{(i)}.$$

The product $*_L$ is associative since $(\mathcal{A}*_L\mathcal{B})*_L\mathcal{C} = \mathcal{A}*_L(\mathcal{B}*_L\mathcal{C})$, and the expression $L(\mathcal{B})\triangle L(\mathcal{C})$ in (2.1) is equivalent to a matrix-matrix product in the transform domain. If we consider a third order tensor $\mathcal{A}$ in (2.1) as a matrix of tubes oriented in the third dimension, then its (i, j)th tube is given by

$$[\mathcal{A}]_{ij} = \sum_{k=1}^{p} \mathcal{B}(i, k, :) *_L \mathcal{C}(k, j, :).$$

This results in a circular convolution between tubes if $*_L$ is the t-product [19]. When instead $*_L$ is the cosine transform product, the resulting multiplication between tubes is the dot-product [16].

Following Kernfeld et al. [16], we denote the transform-domain version of $\mathcal{A}$ by $\hat{\mathcal{A}}$, where $\hat{\mathcal{A}}$ is a tensor whose tube fibers $\hat{\mathbf{a}}$ are computed as

$$\hat{\mathbf{a}}_{ij} = [\hat{\mathcal{A}}]_{ij} := L(\mathbf{a}_{ij}), \quad i = 1, 2, \ldots, \ell, \quad j = 1, 2, \ldots, m.$$

Kernfeld et al. [16] also describe a more efficient way of computing $\hat{\mathcal{A}} = L(\mathcal{A})$ than looping over the row and column indices of $\mathcal{A}$. It is based on computing the `mode-3` matrix product, see [21], between $\mathcal{A} \in \mathbb{R}^{\ell \times m \times n}$ and the invertible matrix $M \in \mathbb{R}^{n \times n}$ associated with the linear transform L according to

$$L(\mathcal{A}) = \mathcal{A} \times_3 M \quad \text{and} \quad L^{-1}(\mathcal{A}) = \mathcal{A} \times_3 M^{-1}, \tag{2.2}$$

and folding the resulting matrix. Note that

$$\mathcal{A} \times_3 M = M\mathcal{A}_{(3)} \in \mathbb{R}^{n \times \ell m},$$

where $\mathcal{A}_{(3)} \in \mathbb{R}^{n \times \ell m}$ is the `mode-3` unfolding of $\mathcal{A}$, which can be obtained by using the `squeeze` operator defined in [17], i.e.,

$$\mathcal{A}_{(3)} = \left[\left(\texttt{squeeze}(\vec{\mathcal{A}}_1) \right)^T, \left(\texttt{squeeze}(\vec{\mathcal{A}}_2) \right)^T, \ldots, \left(\texttt{squeeze}(\vec{\mathcal{A}}_m) \right)^T \right],$$

where $\vec{\mathcal{A}}_j$, $j = 1, 2, \ldots, m$, are the lateral slices of $\mathcal{A}$, and the `squeeze` operator applied to $\vec{\mathcal{X}}$ is identical to the MATLAB squeeze function

$$X = \texttt{squeeze}(\vec{\mathcal{X}}) \implies X(i,k) = \vec{\mathcal{X}}(i,1,k) \quad \forall i,k.$$

In particular, $\texttt{squeeze}(\vec{\mathcal{A}}_j) \in \mathbb{R}^{\ell \times n}$ for $j = 1, 2, \ldots, m$.

The transformation matrix M acts along the tube fibers of $\mathcal{A}$, even though it is written as $M\mathcal{A}_{(3)}$. Using (2.2), an equivalent definition of (2.1) is given by

$$\mathcal{B} *_L \mathcal{C} = L^{-1}(L(\mathcal{B}) \triangle L(\mathcal{C})) = [(\mathcal{B} \times_3 M) \triangle (\mathcal{C} \times_3 M)] \times_3 M^{-1}. \tag{2.3}$$

This definition is described by Kernfeld et al. [16] and used in the computed examples of Section 5.

When M is used as the defining matrix for L in (2.3), the resulting product sometimes is referred to as the *M-product* and denoted by $*_M$; see [18, 25]. Choosing M as the normalized DFT matrix gives the t-product, whereas the DCT matrix yields the cosine transform product; see Kernfeld et al. [16] for a detailed discussion of the cosine transform product, and [1] for an extension to higher order tensors by using the `mode-m` tensor transform.

For any invertible linear operator L, the $*_L$ product between a pair of tensors can be computed by working only in the transform domain using Algorithm 1.

Algorithm 1: $*_L$ product [16]

Input: $\mathcal{B} \in \mathbb{R}^{\ell \times p \times n}$, $\mathcal{C} \in \mathbb{R}^{p \times m \times n}$

$\hat{\mathcal{B}} = L(\mathcal{B})$

$\hat{\mathcal{C}} = L(\mathcal{C})$

for $i = 1$ **to** n **do**

  $\hat{\mathcal{A}}^{(i)} \leftarrow \hat{\mathcal{B}}^{(i)} \hat{\mathcal{C}}^{(i)}$

end

$\mathcal{A} \leftarrow L^{-1}(\hat{\mathcal{A}})$

The following properties of the tensor product $*_L$ have been shown by Kernfeld et al. [16]. Given an invertible linear transform L and a tensor $\mathcal{A} \in \mathbb{R}^{\ell \times m \times n}$, the tensor transpose under $*_L$, $\mathcal{A}^T \in \mathbb{R}^{m \times \ell \times n}$, satisfies

$$[L(\mathcal{A}^T)]^{(i)} = [L(\mathcal{A})^{(i)}]^T, \quad i = 1, 2, \ldots, n.$$

This tensor transpose has similar properties as the matrix transpose. For instance, suppose the tensors $\mathcal{A}$ and $\mathcal{B}$ are such that $\mathcal{A} *_L \mathcal{B}$ and $\mathcal{B}^T *_L \mathcal{A}^T$ are well defined. Then $(\mathcal{A} *_L \mathcal{B})^T = \mathcal{B}^T *_L \mathcal{A}^T$; see [16]. We remark that the tensor transpose is computed by transforming to and from the transform domain using an invertible linear transform matrix, and taking the transpose of each of the frontal slices $\hat{\mathcal{A}}^{(i)}$, $i = 1, 2, \ldots, n$, of $\hat{\mathcal{A}}$ in the transform domain. When the DFT matrix is used to define the $*_L$ product, a conjugate transpose is required to transform back to the spatial domain.

The identity tensor $\mathcal{I} \in \mathbb{R}^{m \times m \times n}$ under L is a tensor such that $\mathcal{I} = L^{-1}(\hat{\mathcal{I}})$, where $\hat{\mathcal{I}}$ is an $m \times m \times n$ tensor, whose frontal slices are the $m \times m$ identity matrix for $i = 1, 2, \ldots, n$. The

diagonal tubes of $\mathcal{I}$ are given by $\mathbf{e}_1 := L^{-1}(\mathbf{e})$, where $\mathbf{e}$ is a $1 \times 1 \times n$ tube fiber of ones and the off-diagonal entries of $\mathcal{I}$ vanish; see [16]. An $m \times m \times n$ tensor $\mathcal{A}$ has an inverse $\mathcal{A}^{-1}$ under $*_L$ provided that $\mathcal{A} *_L \mathcal{A}^{-1} = \mathcal{I}$ and $\mathcal{A}^{-1} *_L \mathcal{A} = \mathcal{I}$; see [16].

A tensor $\mathcal{Q} \in \mathbb{R}^{m\times m\times n}$ is orthogonal if $\mathcal{Q}^T *_L \mathcal{Q} = \mathcal{Q} *_L \mathcal{Q}^T = \mathcal{I}$; see [16]. We remark that the lateral slices of $\mathcal{Q}$ are orthonormal and satisfy

$$\mathcal{Q}^T(:,i,:) *_L \mathcal{Q}(:,j,:) = \begin{cases} \mathbf{e_1} & i = j, \\ \mathbf{0} & i \neq j. \end{cases}$$

The tensor $\mathcal{Q} \in \mathbb{R}^{\ell\times m\times n}$ with $\ell > m$ is said to be partially orthogonal if $\mathcal{Q}^T *_L \mathcal{Q}$ is well defined and equal to the identity tensor $\mathcal{I} \in \mathbb{R}^{m\times m\times n}$.

The tensor singular value decomposition (SVD) $*_L$ factorization of $\mathcal{A} \in \mathbb{R}^{\ell\times m\times n}$ introduced by Kernfeld et al. [16] is given by

$$\mathcal{A} = \mathcal{U} *_L \mathcal{S} *_L \mathcal{V}^T,$$

where $\mathcal{U} \in \mathbb{R}^{\ell\times\ell\times n}$ and $\mathcal{V} \in \mathbb{R}^{m\times m\times n}$ are orthogonal tensors, and the tensor

$$\mathcal{S} = \operatorname{diag}[\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_{\min\{\ell,m\}}] \in \mathbb{R}^{\ell\times m\times n}$$

is f-diagonal with singular tubes $\mathbf{s}_j \in \mathbb{R}^{1\times 1\times n}$, $j = 1, 2, \ldots, \min\{\ell, m\}$, ordered according to

$$\|\mathbf{s}_1\|_F \geq \|\mathbf{s}_2\|_F \geq \cdots \geq \|\mathbf{s}_{\min\{\ell,m\}}\|_F.$$

Note that a tensor is said to be f-diagonal if each frontal slice of the tensor is a diagonal matrix; see [19].

The number of nonzero singular tubes of $\mathcal{A}$ is referred to as the tubal rank of $\mathcal{A}$; see Kilmer et al. [17]. The singular tubes of $\mathcal{A}$ are analogues of the singular values of a matrix A. A linear discrete ill-posed problem with a matrix A has many singular values of different orders of magnitude close to zero. Definition 2.2 describes linear discrete ill-posed tensor problems induced by the $*_L$ product.

Definition 2.2. The tensor least squares problem (1.1) is said to be a linear discrete ill-posed problem for third order tensors under $*_L$ if $\mathcal{A}$ has ill-determined tubal rank, i.e., the Frobenius norm of the singular tubes of $\mathcal{A}$ decays rapidly to zero without a significant gap with increasing index, and there are many nonvanishing singular tubes of tiny Frobenius norm of different orders of magnitude.

We remark that this definition describes a property of the whole tensor $\mathcal{A}$, i.e., the singular tubes of $\mathcal{A}$ which are computed by finding the SVD of each frontal slice $\hat{\mathcal{A}}^{(i)}$, $i = 1, 2, \ldots, n$, of $\hat{\mathcal{A}}$ in the transform domain; see [16] for details.

We conclude this section by introducing notation from El Guide et al. [7]. Let

$$\mathbb{V}_k := [\mathcal{V}_1, \mathcal{V}_2, \ldots, \mathcal{V}_k] \in \mathbb{R}^{m\times kp\times n} \quad \text{and} \quad \mathcal{V}_k := [\vec{\mathcal{V}}_1, \vec{\mathcal{V}}_2, \ldots, \vec{\mathcal{V}}_k] \in \mathbb{R}^{m\times k\times n}$$

with $\mathcal{V}_j \in \mathbb{R}^{m\times p\times n}$ and $\vec{\mathcal{V}}_j \in \mathbb{R}^{m\times 1\times n}$ for $j = 1, 2, \ldots, p$. Let $y = [y_1, y_2, \ldots, y_k]^T \in \mathbb{R}^k$. Then El Guide el al. [7] defined the product $\circledast$ as

$$\mathbb{V}_k \circledast y = \sum_{j=1}^{k} y_j \mathcal{V}_j, \qquad \mathcal{V}_k \circledast y = \sum_{j=1}^{k} y_j \vec{\mathcal{V}}_j.$$

It is readily shown that for any orthogonal tensors $\mathbb{V} \in \mathbb{R}^{\ell\times kp\times n}$ and $\mathcal{V} \in \mathbb{R}^{\ell\times k\times n}$,

$$\|\mathbb{V} \circledast y\|_F = \|y\|_2, \qquad \|\mathcal{V} \circledast y\|_F = \|y\|_2, \tag{2.4}$$

where $\|\cdot\|_2$ denotes the Euclidean vector norm; see [7] for details.

Consider the tensors $\mathcal{C} = [c_{ijk}], \mathcal{D} = [d_{ijk}] \in \mathbb{R}^{m\times p\times n}$ and their lateral slices $\vec{\mathcal{C}} = [c_{i1k}], \vec{\mathcal{D}} = [d_{i1k}] \in \mathbb{R}^{m\times 1\times n}$, and define the scalar products

$$\langle \mathcal{C}, \mathcal{D} \rangle = \sum_{i=1}^{m}\sum_{j=1}^{p}\sum_{k=1}^{n} c_{ijk} d_{ijk}, \qquad \langle \vec{\mathcal{C}}, \vec{\mathcal{D}} \rangle = \sum_{i=1}^{m}\sum_{k=1}^{n} c_{i1k} d_{i1k}.$$

Suppose

$$\mathbb{A} := [\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_m] \in \mathbb{R}^{\ell \times sm \times n} \quad \text{and} \quad \mathbb{B} := [\mathcal{B}_1, \mathcal{B}_2, \ldots, \mathcal{B}_p] \in \mathbb{R}^{\ell \times sp \times n},$$

$$\mathcal{A} := [\vec{\mathcal{A}}_1, \vec{\mathcal{A}}_2, \ldots, \vec{\mathcal{A}}_m] \in \mathbb{R}^{\ell \times m \times n} \quad \text{and} \quad \mathcal{B} := [\vec{\mathcal{B}}_1, \vec{\mathcal{B}}_2, \ldots, \vec{\mathcal{B}}_p] \in \mathbb{R}^{\ell \times p \times n},$$

where $\mathcal{A}_i \in \mathbb{R}^{\ell \times s \times n}$, $\vec{\mathcal{A}}_i \in \mathbb{R}^{\ell \times 1 \times n}$, $i = 1, 2, \ldots, m$, and $\mathcal{B}_j \in \mathbb{R}^{\ell \times s \times n}$, $\vec{\mathcal{B}}_j \in \mathbb{R}^{\ell \times 1 \times n}$, $j = 1, 2, \ldots, p$. Then the T-diamond products [7] denoted by $\mathbb{A}^T \Diamond \mathbb{B}$ and $\mathcal{A}^T \Diamond \mathcal{B}$ result in $m \times p$ matrices with entries

$$[\mathbb{A}^T \Diamond \mathbb{B}]_{ij} = \langle \mathcal{A}_i, \, \mathcal{B}_j \rangle, \qquad [\mathcal{A}^T \Diamond \mathcal{B}]_{ij} = \langle \vec{\mathcal{A}}_i, \, \vec{\mathcal{B}}_j \rangle, \quad i = 1, 2, \ldots, m, \quad j = 1, 2, \ldots, p.$$

The naming scheme for the solution methods for (1.3), (1.7), (1.8), and (1.9), that are described in Sections 3 and 4, is summarized in Table 1.

3 Solution methods for (1.3) and (1.8) based on bidiagonalization processes

Abbreviation	Method	Described in Section
GG-LtGKT	generalized global $*_L$ tensor Golub-Kahan-Tikhonov	3.1
GG-LtGKB	generalized global $*_L$ tensor Golub-Kahan bidiagonalization	3.1
GG-tQR	generalized global tensor QR	3.1, 4.1
G-LtGKT	global $*_L$ tensor Golub-Kahan-Tikhonov	3.2
G-LtGKB	global $*_L$ tensor Golub-Kahan bidiagonalization	3.2
G-tQR	global tensor QR	3.2, 4.2
GG-LtAT	generalized global $*_L$ tensor Arnoldi-Tikhonov	4.1
GG-LtA	generalized global $*_L$ tensor Arnoldi	4.1
GG-LtGMRES	generalized global $*_L$ tensor GMRES	4.1
G-LtAT	global $*_L$ tensor Arnoldi-Tikhonov	4.2
G-LtA	global $*_L$ tensor Arnoldi	4.2
G-LtGMRES	global $*_L$ tensor GMRES	4.2
G-LtGKT$_p$	G-LtGKT applied p times to solve (1.3)	3.2
G-LtAT$_p$	G-LtAT applied p times to solve (1.3)	4.2
G-LtGMRES$_p$	G-LtGMRES applied p times to solve (1.7)	4.2

Table 1: The prefix GG indicates that the method for the solution of tensor least squares problems (1.3) and (1.7) with a general data tensor $\mathcal{B} \in \mathbb{R}^{\ell \times p \times n}$, $p > 1$, works with the whole tensor $\mathcal{B}$ at a time, while the prefix G indicates that the method is designed for the solution of tensor least squares problems (1.8) and (1.9) with a data tensor slice $\vec{\mathcal{B}} \in \mathbb{R}^{\ell \times 1 \times n}$ that is a laterally oriented matrix of size $\ell \times n$. The last three methods with subscript $_p$ work with the lateral slices $\vec{\mathcal{B}}_j$, $j = 1, 2, \ldots, p$, of the data tensor $\mathcal{B}$ independently.

This section describes the generalized global $*_L$ tensor Golub-Kahan-Tikhonov (GG-LtGKT) method and the global $*_L$ tensor Golub-Kahan-Tikhonov (G-LtGKT) method. The G-LtGKT method applied to the solution of (1.3) works with the lateral slices $\vec{\mathcal{B}}_j$, $j = 1, 2, \ldots, p$, of $\mathcal{B}$ independently; when applied to the solution of (1.8), there is only one data tensor slice $\vec{\mathcal{B}}$. The GG-LtGKT method for the solution of (1.3) works with all lateral slices of $\mathcal{B}$ simultaneously.

3.1 The GG-LtGKT method for the solution of (1.3)

This subsection extends the generalized global t-product Golub-Kahan-Tikhonov (GG-tGKT) method for the solution of (1.3) described in [27] to the $*_L$ product. The latter method will be referred to as the GG-LtGKT method. A variant of the GG-tGKT method has recently been presented in [7].

Algorithm 2 extends the generalized global t-product Golub-Kahan bidiagonalization (GG-tGKB) process described in [27] to the $*_L$ product. This algorithm will be referred to as the generalized global $*_L$ tensor Golub-Kahan bidiagonalization (GG-LtGKB) process. We assume that

the number of steps, k, is chosen small enough to avoid breakdown. Then Algorithm 2 produces the partial GG-LtGKB decompositions

$$\mathcal{A} *_L \mathbb{W}_k = \mathbb{V}_{k+1} \circledast \bar{P}_k, \quad \mathcal{A}^T *_L \mathbb{V}_k = \mathbb{W}_k \circledast P_k^T, \tag{3.1}$$

where

$$\mathbb{W}_k := [\mathcal{W}_1, \mathcal{W}_2, \ldots, \mathcal{W}_k] \in \mathbb{R}^{m\times kp\times n}, \;\; \mathbb{V}_j := [\mathcal{V}_1, \mathcal{V}_2, \ldots, \mathcal{V}_j] \in \mathbb{R}^{\ell\times jp\times n}, \quad j \in \{k, k+1\}$$

and

$$\begin{array}{rcl} \mathcal{A} *_L \mathbb{W}_k & := & [\mathcal{A} *_L \mathcal{W}_1, \mathcal{A} *_L \mathcal{W}_2, \ldots, \mathcal{A} *_L \mathcal{W}_k] \in \mathbb{R}^{\ell\times kp\times n}, \\ \mathbb{V}_{k+1} \circledast \bar{P}_k & := & [\mathbb{V}_{k+1} \circledast \bar{P}_k(:,1), \mathbb{V}_{k+1} \circledast \bar{P}_k(:,2), \ldots, \mathbb{V}_{k+1} \circledast \bar{P}_k(:,k)] \in \mathbb{R}^{\ell\times kp\times n}. \end{array} \tag{3.2}$$

The tensors $\mathcal{A}^T *_L \mathbb{V}_k$ and $\mathbb{W}_k \circledast P_k^T$ are similarly defined to (3.2). The matrix

$$\bar{P}_k = \begin{bmatrix} \alpha_1 & & & \\ \beta_2 & \alpha_2 & & \\ & \ddots & \ddots & \\ & & \beta_k & \alpha_k \\ & & & \beta_{k+1} \end{bmatrix} \in \mathbb{R}^{(k+1)\times k} \tag{3.3}$$

is lower bidiagonal, where P_k is the leading $k \times k$ submatrix of $\bar{P}_k$. The tensors $\mathcal{V}_j \in \mathbb{R}^{\ell\times p\times n}$ and $\mathcal{W}_j \in \mathbb{R}^{m\times p\times n}$, for $j = 1, 2, \ldots, k$, generated by Algorithm 2 form orthogonal tensor bases for the t-Krylov subspaces $\mathbb{K}_k(\mathcal{A} *_L \mathcal{A}^T, \mathcal{B})$ and $\mathbb{K}_k(\mathcal{A}^T *_L \mathcal{A}, \mathcal{A}^T *_L \mathcal{B})$, respectively. The relations

$$\mathcal{B} = \mathcal{V}_1\|\mathcal{B}\|_F = \mathbb{V}_{k+1} \circledast e_1\|\mathcal{B}\|_F = \mathbb{V}_{k+1} \circledast e_1\beta_1, \quad e_1 = [1, 0, \ldots, 0]^T, \tag{3.4}$$

follow from Algorithm 2.

Algorithm 2: The partial generalized global $*_L$ tensor Golub-Kahan bidiagonalization (GG-LtGKB) process

Input: $\mathcal{A} \in \mathbb{R}^{\ell\times m\times n}$, $\mathcal{B} \in \mathbb{R}^{\ell\times p\times n}$, such that $\mathcal{A}^T *_L \mathcal{B} \neq \mathcal{O}$

Set $\beta_1 \leftarrow \|\mathcal{B}\|_F, \mathcal{V}_1 \leftarrow \frac{1}{\beta_1}\mathcal{B}$, $\mathcal{W}_0 \leftarrow \mathcal{O}$

for $j = 1, 2, \ldots, k$ **do**

  $\mathcal{W} \leftarrow \mathcal{A}^T *_L \mathcal{V}_j - \beta_j\mathcal{W}_{j-1}$

  $\alpha_j \leftarrow \|\mathcal{W}\|_F$, `If` $\alpha_j = 0$, `stop else`

  $\mathcal{W}_j \leftarrow \mathcal{W}/\alpha_j$

  $\mathcal{V} \leftarrow \mathcal{A} *_L \mathcal{W}_j - \alpha_j\mathcal{V}_j$

  $\beta_{j+1} \leftarrow \|\mathcal{V}\|_F$, `If` $\beta_{j+1} = 0$, `stop else`

  $\mathcal{V}_{j+1} \leftarrow \mathcal{V}/\beta_{j+1}$

end

Let $\mathcal{X} = \mathbb{W}_k \circledast y$, substitute the left-hand side of (3.1) into (1.3), and use (3.4) to obtain the reduced minimization problem

$$\min_{y\in\mathbb{R}^k} \{\|\mathbb{V}_{k+1} \circledast \bar{P}_k \circledast y - \mathbb{V}_{k+1} \circledast e_1\beta_1\|_F^2 + \mu^{-1}\|\mathcal{L} *_L \mathbb{W}_k \circledast y\|_F^2\}. \tag{3.5}$$

Following an approach that is analogous to the one described in [15], we use Algorithm 3 to compute the generalized global tensor QR (GG-tQR) factorization

$$\mathcal{L} *_L \mathbb{W}_k = \mathbb{V}_{\mathcal{L},k} \circledast R_{\mathcal{L},k}, \tag{3.6}$$

where $R_{\mathcal{L},k} \in \mathbb{R}^{k\times k}$ is an upper triangular matrix, and $\mathbb{V}_{\mathcal{L},k} \in \mathbb{R}^{s\times kp\times n}$ has k orthogonal tensor columns. The factorization (3.6) can be evaluated by updating the available GG-tQR factorization of $\mathcal{L} *_L \mathbb{W}_{k-1}$. The regularization operators $\mathcal{L}$ used in the computed examples are described in Section 5.

Algorithm 3: Generalized global tensor QR (GG-tQR) factorization [27]

Input: $\mathbb{A} := [\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_k] \in \mathbb{R}^{\ell \times km \times n}, \mathcal{A}_j \in \mathbb{R}^{\ell \times m \times n}, j = 1, \ldots, k,\ \ell \geq m.$

Output: $\mathbb{V} := [\mathcal{V}_1, \mathcal{V}_2, \ldots, \mathcal{V}_k] \in \mathbb{R}^{\ell \times km \times n}, \mathcal{V}_j \in \mathbb{R}^{\ell \times m \times n}, j = 1, \ldots, k,\ R = [r_{ij}] \in \mathbb{R}^{k \times k}$ such that $\mathbb{A} = \mathbb{V} \circledast R$, and $\mathbb{V}^T \lozenge \mathbb{V} = I_k$

Set $r_{11} \leftarrow \langle \mathcal{A}_1, \mathcal{A}_1 \rangle^{1/2}$, $\mathcal{V}_1 \leftarrow \frac{1}{r_{11}} \mathcal{A}_1$
for $j = 1, 2 \ldots, k$ **do**
  $\mathcal{W} \leftarrow \mathcal{A}_j$
  for $i = 1, 2, \ldots, j-1$ **do**
    $r_{ij} \leftarrow \langle \mathcal{V}_i, \mathcal{W} \rangle$
    $\mathcal{W} \leftarrow \mathcal{W} - r_{ij} \mathcal{V}_i$
  end
  $r_{jj} \leftarrow \langle \mathcal{W}, \mathcal{W} \rangle^{1/2}$
  $\mathcal{V}_j \leftarrow \mathcal{W} / r_{jj}$
end

Substitute (3.6) into (3.5), and use the left-hand side of (2.4) to obtain

$$\min_{y \in \mathbb{R}^k} \{ \| \bar{P}_k y - e_1 \beta_1 \|_2^2 + \mu^{-1} \| R_{\mathcal{L},k} y \|_2^2 \}. \tag{3.7}$$

We would like to transform (3.7) into a Tikhonov minimization problem in standard form. With this aim, define the quantities

$$z := R_{\mathcal{L},k} y, \quad \widetilde{P}_k := \bar{P}_k R_{\mathcal{L},k}^{-1}, \tag{3.8}$$

where we assume the matrix $R_{\mathcal{L},k}$ to be invertible and not very ill-conditioned. This holds for many regularization operators $\mathcal{L}$, and in particular for the ones used in the computed examples of Section 5. Then the transformation (3.8) is attractive to apply. Substitution into (3.7) yields the Tikhonov minimization problem in standard form,

$$\min_{z \in \mathbb{R}^k} \{ \| \widetilde{P}_k z - e_1 \beta_1 \|_2^2 + \mu^{-1} \| z \|_2^2 \}. \tag{3.9}$$

The normal equations associated with (3.9) are given by

$$(\widetilde{P}_k^T \widetilde{P}_k + \mu^{-1} I) z = \widetilde{P}_k^T e_1 \beta_1, \tag{3.10}$$

and their solution, for any $\mu > 0$, can be written as

$$z_{\mu,k} = (\widetilde{P}_k^T \widetilde{P}_k + \mu^{-1} I)^{-1} \widetilde{P}_k^T e_1 \beta_1. \tag{3.11}$$

It follows that the computed approximate solution to the Tikhonov minimization problem (1.3) can be expressed as

$$\mathcal{X}_{\mu,k} = \mathbb{W}_k \circledast R_{\mathcal{L},k}^{-1} (\widetilde{P}_k^T \widetilde{P}_k + \mu^{-1} I)^{-1} \widetilde{P}_k^T e_1 \beta_1.$$

We remark that we compute the vector $z_{\mu,k}$ in (3.11) by solving the least squares problem

$$\min_{z \in \mathbb{R}^k} \left\| \begin{bmatrix} \widetilde{P}_k \\ \mu^{-1/2} I \end{bmatrix} z - \begin{bmatrix} e_1 \beta_1 \\ 0 \end{bmatrix} \right\|_2,$$

because the condition number of the matrix in this problem is the square root of the condition number of the matrix $\widetilde{P}_k^T \widetilde{P}_k + \mu^{-1} I$ in (3.10).

The regularization parameter and the required number of steps k by the GG-LtGKB process are determined by the discrepancy principle (1.5), which prescribes that $\mu > 0$ be chosen so that the solution (3.11) satisfies

$$\| \widetilde{P}_k z_{\mu,k} - e_1 \beta_1 \|_2 = \eta \delta; \tag{3.12}$$

see Proposition 3.1 below. We choose k as small as possible so that the above equality can be satisfied. Define the function

$$\phi_k(\mu) := \| \widetilde{P}_k z_{\mu,k} - e_1 \beta_1 \|_2^2, \tag{3.13}$$

substitute (3.11) into (3.13), and use the identity

$$I - \widetilde{P}_k(\widetilde{P}_k^T\widetilde{P}_k + \mu^{-1}I)^{-1}\widetilde{P}_k^T = (\mu\widetilde{P}_k\widetilde{P}_k^T + I)^{-1}$$

to obtain

$$\phi_k(\mu) = \beta_1^2 e_1^T(\mu\bar{P}_k\bar{P}_k^T + I)^{-2}e_1.$$

It is readily shown that $\phi_k(\mu)$ is decreasing and convex with $\phi_k(0) = \beta_1^2$; see [27] for details. A zero finder such as bisection or Newton's method can be used to determine the solution μ_k of

$$\phi_k(\mu) - \eta^2\delta^2 = 0. \tag{3.14}$$

We will use the bisection method in the computed examples of Section 5.

The following result shows that we can apply the discrepancy principle (1.5) to the reduced problem (3.9) to determine $\mu > 0$ that satisfy (3.12); see [27] for a proof of a related result.

Proposition 3.1. Let $\mu = \mu_k$ solve (3.14) and suppose $z_{\mu,k}$ is the solution of the normal equations (3.10). Let $y_{\mu,k}$ and $z_{\mu,k}$ be related by (3.8). Then the associated approximate solution $\mathcal{X}_{\mu,k} = \mathbb{W}_k \circledast y_{\mu,k}$ of (1.3) satisfies

$$\|\mathcal{A} *_L \mathcal{X}_{\mu,k} - \mathcal{B}\|_F^2 = \beta_1^2 e_1^T(\mu\widetilde{P}_k\widetilde{P}_k^T + I)^{-2}e_1.$$

We refer to the above solution method as the GG-LtGKT method. It is implemented by using [27, Algorithm 10] with the t-product replaced by the $*_L$ product.

3.2 The G-LtGKT method for the solution of (1.8) and (1.3)

This subsection extends the global t-product Golub-Kahan-Tikhonov (G-tGKT) method described in [27] for the approximate solution of (1.8) and (1.3) to the $*_L$ product. The methods obtained will be referred to as the G-LtGKT and G-LtGKT$_p$ methods, respectively. Algorithm 4 extends the global t-product Golub-Kahan bidiagonalization (G-tGKB) process described in [27] to the $*_L$ product. The latter method is referred to as the G-LtGKB process.

Assume that the number of steps, k, with the G-LtGKB process is small enough to avoid breakdown. This is the generic situation. Then Algorithm 4 yields the G-LtGKB decompositions

$$\mathcal{A} *_L \mathcal{W}_k = \mathcal{Q}_{k+1} \circledast \bar{B}_k, \quad \mathcal{A}^T *_L \mathcal{Q}_k = \mathcal{W}_k \circledast B_k^T, \tag{3.15}$$

where

$$\mathcal{W}_k := [\vec{\mathcal{W}}_1, \vec{\mathcal{W}}_2, \ldots, \vec{\mathcal{W}}_k] \in \mathbb{R}^{m\times k\times n}, \quad \mathcal{Q}_j := [\vec{\mathcal{Q}}_1, \vec{\mathcal{Q}}_2, \ldots, \vec{\mathcal{Q}}_j] \in \mathbb{R}^{\ell\times j\times n}, \quad j \in \{k, k+1\}.$$

The expressions $\mathcal{A} *_L \mathcal{W}_k$, $\mathcal{Q}_{k+1} \circledast \bar{B}_k$, $\mathcal{A}^T *_L \mathcal{Q}_k$, and $\mathcal{W}_k \circledast B_k^T$ are analogous to those in (3.2) and the lower bidiagonal matrix $\bar{B}_k \in \mathbb{R}^{(k+1)\times k}$ is of the form (3.3), where B_k is the leading $k \times k$ submatrix of $\bar{B}_k$. The tensor columns $\vec{\mathcal{Q}}_j \in \mathbb{R}^{\ell\times 1\times n}$ and $\vec{\mathcal{W}}_j \in \mathbb{R}^{m\times 1\times n}$, $j = 1, 2, \ldots, k$, generated by Algorithm 4 make up orthonormal tensor bases for the t-Krylov subspaces $\mathbb{K}_k(\mathcal{A} *_L \mathcal{A}^T, \vec{\mathcal{B}})$ and $\mathbb{K}_k(\mathcal{A}^T *_L \mathcal{A}, \mathcal{A}^T *_L \vec{\mathcal{B}})$, respectively. It can easily be deduced from Algorithm 4 that

$$\vec{\mathcal{B}} = \mathcal{Q}_{k+1} \circledast e_1\beta_1. \tag{3.16}$$

Algorithm 4: The partial global $*_L$ tensor Golub-Kahan bidiagonalization (G-LtGKB) process

Input: $\mathcal{A} \in \mathbb{R}^{\ell\times m\times n}$, $\vec{\mathcal{B}} \in \mathbb{R}^{\ell\times 1\times n}$, $\mathcal{A}^T *_L \vec{\mathcal{B}} \neq \vec{\mathcal{O}}$

Set $\beta_1 \leftarrow \|\vec{\mathcal{B}}\|_F$, $\vec{\mathcal{Q}}_1 \leftarrow \frac{1}{\beta_1}\vec{\mathcal{B}}$, $\vec{\mathcal{W}}_0 \leftarrow \vec{\mathcal{O}}$

for $j = 1, 2, \ldots, k$ **do**

  $\vec{\mathcal{W}} \leftarrow \mathcal{A}^T *_L \vec{\mathcal{Q}}_j - \beta_j\vec{\mathcal{W}}_{j-1}$

  $\alpha_j \leftarrow \|\vec{\mathcal{W}}\|_F$. `If` $\alpha_j = 0$, `stop else`

  $\vec{\mathcal{W}}_j \leftarrow \vec{\mathcal{W}}/\alpha_j$

  $\vec{\mathcal{Q}} \leftarrow \mathcal{A} *_L \vec{\mathcal{W}}_j - \alpha_j\vec{\mathcal{Q}}_j$

  $\beta_{j+1} \leftarrow \|\vec{\mathcal{Q}}\|_F$. `If` $\beta_{j+1} = 0$, `stop else`

  $\vec{\mathcal{Q}}_{j+1} \leftarrow \vec{\mathcal{Q}}/\beta_{j+1}$

end

Let $\vec{\mathcal{X}} = \mathcal{W}_k \circledast y$. We follow a similar approach as in Subsection 3.1 to reduce (1.8) to a problem of small size. Thus, substitute the left-side of (3.15) as well as (3.16) into (1.8). Then compute the G-tQR factorization

$$\mathcal{L} *_L \mathcal{W}_k = \mathcal{Q}_{\mathcal{L},k} \circledast \bar{R}_{\mathcal{L},k}$$

by Algorithm 5, and use the right-hand side of (2.4) to obtain the minimization problem

$$\min_{y\in\mathbb{R}^k} \{\|\breve{P}_k z - e_1\beta_1\|_2^2 + \mu^{-1}\|z\|_2^2\}, \tag{3.17}$$

where

$$z := \bar{R}_{\mathcal{L},k} y, \quad \breve{P}_k := \bar{B}_k \bar{R}_{\mathcal{L},k}^{-1}. \tag{3.18}$$

Here we assume the matrix $R_{\mathcal{L},k}$ to be invertible and not very ill-conditioned. This holds for the regularization operator $\mathcal{L}$ used in the computed examples in Section 5. The solution method for (3.17) is analogous to the method in Subsection 3.1, and is referred to as the G-LtGKT method. It can be implemented by [27, Algorithm 13] with $p = 1$.

Algorithm 5: Global tensor QR (G-tQR) factorization [27]

Input: $\mathcal{A} = [\vec{\mathcal{A}}_1, \vec{\mathcal{A}}_2, \ldots, \vec{\mathcal{A}}_m] \in \mathbb{R}^{\ell\times m\times n}$, $\ell \geq m$
Output: $\mathcal{Q} = [\vec{\mathcal{Q}}_1, \vec{\mathcal{Q}}_2, \ldots, \vec{\mathcal{Q}}_m] \in \mathbb{R}^{\ell\times m\times n}$, $R = [r_{ij}] \in \mathbb{R}^{m\times m}$ such that $\mathcal{A} = \mathcal{Q} \circledast \bar{R}$ and $\mathcal{Q}^T \lozenge \mathcal{Q} = I_m$

$r_{11} \leftarrow \langle \vec{\mathcal{A}}_1, \vec{\mathcal{A}}_1\rangle^{1/2}$, $\vec{\mathcal{Q}}_1 \leftarrow \frac{1}{r_{11}}\vec{\mathcal{A}}_1$
for $j = 1, 2, \ldots, m$ **do**
  $\vec{\mathcal{W}} \leftarrow \vec{\mathcal{A}}_j$
  for $i = 1, 2, \ldots, j-1$ **do**
    $r_{ij} \leftarrow \langle \vec{\mathcal{Q}}_i, \vec{\mathcal{W}}\rangle$
    $\vec{\mathcal{W}} \leftarrow \vec{\mathcal{W}} - r_{ij}\vec{\mathcal{Q}}_i$
  end
  $r_{jj} \leftarrow \langle \vec{\mathcal{W}}, \vec{\mathcal{W}}\rangle^{1/2}$
  $\vec{\mathcal{Q}}_j \leftarrow \vec{\mathcal{W}}/r_{jj}$
end

We determine the regularization parameter $\mu > 0$ and the number of steps of the G-LtGKB process similarly as in Subsection 3.1. Thus, let a bound for the error $\vec{\mathcal{E}}$ in $\vec{\mathcal{B}}$ be known, i.e.,

$$\|\vec{\mathcal{E}}\|_F \leq \delta.$$

The discrepancy principle prescribes that $\mu > 0$ be determined so that the solution $z_{\mu,k}$ of (3.17) satisfies

$$\|\breve{P}_k z_{\mu,k} - e_1\beta_1\|_2 = \eta\delta$$

for some constant $\eta > 1$ that is independent of δ. Define

$$\psi_k(\mu) := \|\breve{P}_k z_{\mu,k} - e_1\beta_1\|_2^2.$$

We obtain similarly as in Subsection 3.1 that

$$\psi_k(\mu) = \beta_1^2 e_1^T(\mu\breve{P}_k\breve{P}_k^T + I)^{-2}e_1.$$

The following result is analogous to Proposition 3.1 and can be shown in a similar fashion.

Proposition 3.2. Let $\mu = \mu_k$ solve $\psi_k(\mu) = \eta^2\delta^2$ and suppose that $z_{\mu,k}$ is the solution of the normal equations of (3.17), i.e., of

$$(\breve{P}_k^T\breve{P}_k + \mu^{-1}I)z = \breve{P}_k^T e_1\beta_1.$$

Let $y_{\mu,k}$ and $z_{\mu,k}$ be related by (3.18). Then the associated approximate solution $\vec{\mathcal{X}}_{\mu,k} = \mathcal{W}_k \circledast y_{\mu,k}$ of (1.8) satisfies

$$\|\mathcal{A} *_L \vec{\mathcal{X}}_{\mu,k} - \vec{\mathcal{B}}\|_F^2 = \beta_1^2 e_1^T(\mu\breve{P}_k\breve{P}_k^T + I)^{-2}e_1.$$

Finally, we discuss the solution of (1.3) by applying the G-LtGKB process and the G-LtGKT method to each one of the p Tikhonov minimization problems

$$\min_{\vec{\mathcal{X}}_j \in \mathbb{R}^{m\times 1\times n}} \left\{ \|\mathcal{A} *_L \vec{\mathcal{X}}_j - \vec{\mathcal{B}}_j\|_F^2 + \mu^{-1} \|\mathcal{L} *_L \vec{\mathcal{X}}_j\|_F^2 \right\}, \quad j = 1, 2, \ldots, p, \quad p > 1, \tag{3.19}$$

separately. We will refer to this solution approach as the G-LtGKT$_p$ method. It is implemented by replacing the t-product in [27, Algorithm 13] by the $*_L$ product. The solution method for (3.19) obtained in this manner provides an alternative to the GG-LtGKT method of Subsection 3.1. Its performance is illustrated in Section 5.

4 Solution methods for (1.3) and (1.8) based on Arnoldi processes

This section describes the generalized global $*_L$ tensor Arnoldi-Tikhonov (GG-LtAT) method, which works with all lateral slices of $\mathcal{B}$ simultaneously. We also present the global $*_L$ tensor Arnoldi-Tikhonov (G-LtAT) method for the approximate solution of (1.3). The latter method is also applied to determine the solution of (1.8) by working with the lateral slices $\vec{\mathcal{B}}_j$, $j = 1, 2, \ldots, p$, of $\mathcal{B}$ independently.

4.1 The GG-LtAT method for the approximate solution of (1.3)

This subsection extends the generalized global t-product Arnoldi-Tikhonov (GG-tAT) method described in [28] to the $*_L$ product. We refer to this solution scheme as the GG-LtAT method. It works with the whole data tensor $\mathcal{B}$ at a time. This method applies an extension of the T-global Arnoldi process described by El Guide et al. [7] to the $*_L$ product. The resulting process is described by Algorithm 6 below and will be referred to as the generalized global $*_L$ tensor Arnoldi (GG-LtA) process.

Let $\mathcal{A} \in \mathbb{R}^{m\times m\times n}$ and assume that the number of steps, ℓ, is small enough to avoid breakdown. This is the generic situation. Algorithm 6 yields the GG-LtA decomposition

$$\mathcal{A} *_L \mathbb{V}_\ell = \mathbb{V}_{\ell+1} \circledast \bar{H}_\ell, \tag{4.1}$$

where

$$\mathbb{V}_j := [\mathcal{V}_1, \mathcal{V}_2, \ldots, \mathcal{V}_j] \in \mathbb{R}^{m\times jp\times n}, \quad j \in \{\ell, \ell+1\},$$

and

$$\bar{H}_\ell = \begin{bmatrix} h_{11} & & & \cdots & h_{1\ell} \\ h_{21} & h_{22} & & & \\ & h_{32} & h_{33} & & \vdots \\ & & \ddots & \ddots & \\ & & & h_{\ell,\ell-1} & h_{\ell,\ell} \\ O & & & & h_{\ell+1,\ell} \end{bmatrix} \in \mathbb{R}^{(\ell+1)\times\ell}$$

is an upper Hessenberg matrix. The tensors $\mathcal{V}_j \in \mathbb{R}^{m\times p\times n}$, $j = 1, 2, \ldots, \ell$, generated by Algorithm 6 form an orthogonal tensor basis for the t-Krylov subspace (1.6). The tensors $\mathcal{A} *_L \mathbb{V}_\ell$ and $\mathbb{V}_{\ell+1} \circledast \bar{H}_\ell$ are defined similarly as in (3.2). The relation

$$\mathcal{B} = \mathbb{V}_{\ell+1} \circledast e_1 \beta \tag{4.2}$$

follows from Algorithm 6.

Algorithm 6: The generalized global $*_L$ tensor Arnoldi (GG-LtA) process

Input: $\mathcal{A} \in \mathbb{R}^{m\times m\times n}$, $\mathcal{B} \in \mathbb{R}^{m\times p\times n} \neq \mathcal{O}$

Set $\beta \leftarrow \|\mathcal{B}\|_F, \mathcal{V}_1 \leftarrow \frac{1}{\beta}\mathcal{B}$
for $j = 1, 2, \ldots, \ell$ **do**
  $\mathcal{W} \leftarrow \mathcal{A} *_L \mathcal{V}_j$
  for $i = 1, 2, \ldots, j$ **do**
    $h_{ij} \leftarrow \langle \mathcal{V}_i, \mathcal{W} \rangle$
    $\mathcal{W} \leftarrow \mathcal{W} - h_{ij}\mathcal{V}_i$
  end
  $h_{j+1,j} \leftarrow \|\mathcal{W}\|_F$, `If` $h_{j+1,j} = 0$, `stop`; `else`
  $\mathcal{V}_{j+1} \leftarrow \mathcal{W}/h_{j+1,j}$
end

We proceed in the same manner as in Section 3.1. Let $\mathcal{X} = \mathbb{V}_\ell \circledast y$ and substitute (4.1) and (4.2) into (1.3) to obtain an analogue of (3.5). Compute a GG-tQR factorization of $\mathcal{L} *_L \mathbb{Q}_\ell$, which is analogous to (3.6), by Algorithm 3. Then using the left-hand side of (2.4), we obtain

$$\min_{y\in\mathbb{R}^\ell} \{\|\bar{H}_\ell y - e_1\beta\|_2^2 + \mu^{-1}\|R_{\mathcal{L},\ell}y\|_2^2\}. \tag{4.3}$$

When the matrix $R_{\mathcal{L},\ell}$ is invertible and not very ill-conditioned, we introduce the quantities

$$z := R_{\mathcal{L},\ell}y, \quad \widetilde{H}_\ell := \bar{H}_\ell R_{\mathcal{L},\ell}^{-1},$$

which allow us to transform the problem (4.3) to a Tikhonov minimization problem in standard form,

$$\min_{y\in\mathbb{R}^\ell} \{\|\widetilde{H}_\ell z - e_1\beta\|_2^2 + \mu^{-1}\|z\|_2^2\}.$$

This problem can be solved similarly as the method described in Section 3.1. We refer to this solution method as the GG-LtAT method. It can be implemented in the same manner as [28, Algorithm 11], which uses the t-product.

We remark that when $\mu = \infty$ in (4.3), we obtain the minimization problem that is solved in step ℓ of the generalized global $*_L$ tensor GMRES (GG-LtGMRES) method for the approximate solution of (1.7). The special case of this method when the matrix that defines the linear transform L is the normalized DFT matrix has been described in [28]; its implementation is given by [28, Algorithm 12]. The GG-LtGMRES method can be implemented in the same fashion.

4.2 The G-tLAT method for the solution of (1.3) and (1.8)

We extend the global t-product Arnoldi-Tikhonov (G-tAT) regularization method described in [28] to the $*_L$ product. The method so obtained will be referred to as the G-LtAT method.

The G-LtAT method for the solution of (1.3) provides an alternative approach to the GG-LtAT method described in Subsection 4.1. It is applied to each lateral slice $\vec{\mathcal{B}}_j$, $j = 1, 2, \ldots, p$, of $\mathcal{B}$ in (1.3) independently. We will refer to this approach of solving (3.19) as the G-LtAT$_p$ method. This method has been described in [28] when using the t-product, i.e., when the defining matrix for the linear transform L in (3.19) is the normalized DFT matrix. The implementation of the G-LtAT$_p$ method is similar to [28, Algorithm 14] with the main difference that the global t-Arnoldi process is replaced by the global $*_L$ Arnoldi (G-LtA) process described by Algorithm 7. As already mentioned, our interest in solution methods that work with the tensor slices $\vec{\mathcal{B}}_j$ independently is that they were found in [27, 28] to yield approximate solutions of (1.3) of higher quality than when working with these lateral slices simultaneously.

We first consider the G-LtAT method for the solution of (1.8). As usual, we assume that the number of steps, ℓ, is small enough to avoid breakdown. Algorithm 7 yields the G-LtA decomposition

$$\mathcal{A} *_L \mathcal{Q}_\ell = \mathcal{Q}_{\ell+1} \circledast \bar{\bar{H}}_\ell, \tag{4.4}$$

where

$$\mathcal{Q}_j := [\vec{\mathcal{Q}}_1, \vec{\mathcal{Q}}_2, \ldots, \vec{\mathcal{Q}}_j] \in \mathbb{R}^{m\times j\times n}, \quad j \in \{\ell, \ell+1\}.$$

The tensor columns $\vec{\mathcal{Q}}_j \in \mathbb{R}^{m\times 1\times n}$, $j = 1, 2, \ldots, \ell$, form an orthonormal tensor basis for the t-Krylov subspace $\mathbb{K}_\ell(\mathcal{A}, \vec{\mathcal{B}})$. The tensors $\mathcal{A} *_L \mathcal{Q}_\ell \in \mathbb{R}^{m\times k\times n}$ and $\mathcal{Q}_{\ell+1} \circledast \bar{H}_\ell \in \mathbb{R}^{m\times(\ell+1)\times n}$ are defined similarly as (3.2) and the matrix $\bar{\bar{H}} \in \mathbb{R}^{(\ell+1)\times\ell}$ is of upper Hessenberg form. We can deduce from Algorithm 7 that

$$\vec{\mathcal{B}} = \mathcal{Q}_{\ell+1} \circledast e_1\beta.$$

Algorithm 7: The global $*_L$ tensor Arnoldi (G-LtA) process

Input: $\mathcal{A} \in \mathbb{R}^{m\times m\times n}$, $\vec{\mathcal{B}} \in \mathbb{R}^{m\times 1\times n} \neq \vec{\mathcal{O}}$

Set $\beta \leftarrow \|\vec{\mathcal{B}}\|_F$, $\vec{\mathcal{Q}}_1 \leftarrow \frac{1}{\beta}\vec{\mathcal{B}}$
for $j = 1, 2, \ldots, \ell$ **do**
  $\vec{\mathcal{W}} \leftarrow \mathcal{A} *_L \vec{\mathcal{Q}}_j$
  for $i = 1, 2, \ldots, j$ **do**
    $h_{ij} \leftarrow \langle \vec{\mathcal{Q}}_i, \vec{\mathcal{W}} \rangle$
    $\vec{\mathcal{W}} \leftarrow \vec{\mathcal{W}} - h_{ij}\vec{\mathcal{Q}}_i$
  end
  $h_{j+1,j} \leftarrow \|\vec{\mathcal{W}}\|_F$, `If` $h_{j+1,j} = 0$, `stop`; `else`
  $\vec{\mathcal{Q}}_{j+1} \leftarrow \vec{\mathcal{W}}/h_{j+1,j}$
end

Suppose that $\vec{\mathcal{X}} = \mathcal{Q}_\ell \circledast y$ and assume, as usual, that ℓ is chosen small enough so that the factorization (4.4) with the stated properties exists. Then, similarly as in Section 3.2, we reduce (1.8) to the minimization problem

$$\min_{y\in\mathbb{R}^\ell} \{\|\bar{\bar{H}}_\ell y - e_1\beta\|_2^2 + \mu^{-1}\|\bar{R}_{\mathcal{L},\ell} y\|_2^2\}. \tag{4.5}$$

Compute the G-tQR factorization of $\mathcal{L} *_L \mathcal{Q}_\ell$ analogously to (3.6) by Algorithm 5, and introduce the quantities

$$z := \bar{R}_{\mathcal{L},\ell} y, \quad \breve{H}_\ell := \bar{\bar{H}}_\ell \bar{R}_{\mathcal{L},\ell}^{-1},$$

which we assume exist. Then similarly as in Section 3, we obtain the Tikhonov minimization problem in standard form

$$\min_{y\in\mathbb{R}^\ell} \{\|\breve{H}_\ell z - e_1\beta\|_2^2 + \mu^{-1}\|z\|_2^2\},$$

which we solve for z. We refer to this solution scheme as the G-LtAT method. It can be implemented similarly as [28, Algorithm 14] with $p = 1$ and the t-product replaced by the $*_L$ product.

Similarly as above, we note that setting $\mu = \infty$ in (4.5) results in the minimization problem that is solved at step ℓ of the global $*_L$ tensor GMRES (G-LtGMRES) method for the approximate solution of (1.9). An implementation of this method with the t-product is described by [28, Algorithm 15] with $p = 1$. The G-LtGMRES method can be implemented similarly.

We also apply the G-LtGMRES method to the solution of the minimization problem (1.7) using one lateral slice $\vec{\mathcal{B}}_j$, $j = 1, 2, \ldots, p$, of $\mathcal{B}$ at a time, i.e., we solve separately the p minimization problems

$$\|\mathcal{A} *_L \vec{\mathcal{X}}_{j,\ell} - \vec{\mathcal{B}}_j\|_F = \min_{\vec{\mathcal{X}}\in\mathbb{K}_\ell(\mathcal{A},\vec{\mathcal{B}})} \|\mathcal{A} *_L \vec{\mathcal{X}}_j - \vec{\mathcal{B}}_j\|_F, \quad j = 1, 2, \ldots, p, \quad p > 1, \tag{4.6}$$

for $\vec{\mathcal{X}}_{j,\ell} \in \mathbb{K}_\ell(\mathcal{A}, \vec{\mathcal{B}})$. This solution method is referred to as the G-LtGMRES$_p$ method. Analogues of the G-LtGMRES and G-LtGMRES$_p$ methods, that use the t-product, have been described in [28].

5 Numerical examples

This section compares and illustrates the effectiveness of the methods described in Sections 3 and 4. Applications to color image and video restoration are considered. All computations were carried

in MATLAB 2019b on a Lenovo computer with an Intel Core i3 processor and 4 GB RAM running Windows 10.

We use three different regularization operators in the computed examples with Tikhonov regularization: the identity operator $\mathcal{L} = \mathcal{I}$, the operator $\mathcal{L}_1 \in \mathbb{R}^{(m-2)\times m\times n}$ with a tridiagonal first frontal slice

$$\mathcal{L}_1^{(1)} = \frac{1}{4}\begin{bmatrix} -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \end{bmatrix} \in \mathbb{R}^{(m-2)\times m} \tag{5.1}$$

and the remaining frontal slices $\mathcal{L}_1^{(i)} \in \mathbb{R}^{(m-2)\times m}$, $i = 2, 3, \ldots, n$, equal to the zero matrix, as well as the regularization operator $\mathcal{L}_2 \in \mathbb{R}^{(m-1)\times m\times n}$ with a bidiagonal first frontal slice

$$\mathcal{L}_2^{(1)} = \frac{1}{2}\begin{bmatrix} 1 & -1 & & & \\ & 1 & -1 & & \\ & & \ddots & \ddots & \\ & & & 1 & -1 \end{bmatrix} \in \mathbb{R}^{(m-1)\times m} \tag{5.2}$$

and the remaining frontal slices $\mathcal{L}_2^{(i)} \in \mathbb{R}^{(m-1)\times m}$, $i = 2, 3, \ldots, n$, equal to the zero matrix. The results for GMRES-type methods are independent of the regularization operator $\mathcal{L}$; cf. (1.7) and (1.9). The construction of the blurring operator $\mathcal{A}$ follows a similar approach as described by Kernfeld et al. [16].

We apply the `twist` and `squeeze` operators defined by Kilmer et al. [17] to associate an image stored as an $m\times n$ matrix to a tensor column of size $m\times 1\times n$. Additionally, we use the `multi_twist` and `multi_squeeze` operators described in [27, 28] to relate an $m \times p \times n$ image to a tensor of size $m \times n \times p$ for $p > 1$.

Example 5.1. This example uses an orthogonal DCT matrix from MATLAB as the defining matrix for the linear transform L to define the $*_L$-product. This gives the $*_c$-product introduced by Kernfeld [16]. We apply the regularization operator $\mathcal{L}_1$ defined above, and consider the restoration of the `peppers` image shown in Figure 1 (left). This image is stored as a tensor $\mathcal{B} \in \mathbb{R}^{300\times 3\times 300}$ by using the `multi_twist` operator and is blurred by the operator $\mathcal{A} \in \mathbb{R}^{300\times 300\times 300}$, which is generated by applying the function `blur` in [12]. Specifically, we use the MATLAB commands below, and the operator `ten` described in [16] to generate $\mathcal{A}$:

$$\begin{aligned} &\texttt{z} = [\texttt{exp}(-([\texttt{0}:\texttt{band}-\texttt{1}].^{\texttt{2}})/(2\sigma^2)), \texttt{zeros}(\texttt{1},\texttt{N}-\texttt{band})], \quad N = 300, \quad \sigma = 3, \quad \texttt{band} = 12, \\ &\texttt{y} = \texttt{zeros}(\texttt{length}(\texttt{z}),\texttt{1}), \quad \texttt{y}(\texttt{1}:\texttt{length}(\texttt{z})-\texttt{1}) = \texttt{z}(\texttt{2}:\texttt{end}), \\ &A_1 = \texttt{toeplitz}(\texttt{z}), \quad A_2 = \texttt{flip}(\texttt{flip}(\texttt{hankel}(\texttt{y}))') + \texttt{hankel}(\texttt{y}), \\ &A = A_1 + A_2, \quad \mathcal{A} = \frac{1}{\sqrt{2\pi\sigma}}\texttt{ten}(A\otimes A, 300, 300, 300), \end{aligned} \tag{5.3}$$

where $A_1 \in \mathbb{R}^{300\times 300}$ is a Toeplitz matrix and $A_2 \in \mathbb{R}^{300\times 300}$ is a Hankel matrix. Reflective boundary conditions are employed; see [14, Chapter 4] for discussions.

The condition number with regard to the spectral matrix norm of the frontal slices $\mathcal{A}^{(i)}$ is $1.2\cdot 10^6$ for $i = 1, 2, \ldots, 12$, and the condition number of the remaining slices $\mathcal{A}^{(i)}$, $13 \le i \le 300$, is infinite. Here and below, we use the MATLAB command `cond` to compute the condition number of the frontal slices of $\mathcal{A}$.

Let $\mathcal{X}_{\text{true}} \in \mathbb{R}^{300\times 3\times 300}$ represent the blur- and noise-free image that we would like to determine from the available contaminated image that is represented by $\mathcal{B}$; we assume that $\mathcal{X}_{\text{true}}$ is not available. The blur-contaminated but noise-free image associated with $\mathcal{X}_{\text{true}}$ is generated by $\mathcal{B}_{\text{true}} = \mathcal{A} *_L \mathcal{X}_{\text{true}}$. This image will be contaminated by noise that is represented by the tensor $\mathcal{E} \in \mathbb{R}^{300\times 3\times 300}$ given by

$$\mathcal{E} := \widetilde{\delta}\frac{\mathcal{E}_0}{\|\mathcal{E}_0\|_F}\|\mathcal{B}_{\text{true}}\|_F, \tag{5.4}$$

where the entries of $\mathcal{E}_0$ are $N(0,1)$. Hence, the entries of $\mathcal{E}$ are normally distributed with zero mean and are scaled to have the specific noise level $\widetilde{\delta} > 0$. The available blurred and noisy image is represented by $\mathcal{B} = \mathcal{B}_{\text{true}} + \mathcal{E}$. This image is displayed in Figure 1 (right) using the `multi_squeeze` operator. We would like to determine an accurate approximation of $\mathcal{X}_{\text{true}}$ from $\mathcal{B}$.

Figure 1: True `peppers` image (left), and blurred and noisy `peppers` image (right) for $\tilde{\delta} = 10^{-3}$.

Figure 2: Reconstructed `peppers` images for $\tilde{\delta} = 10^{-3}$ by the G-LtAT$_p$ method (left) and the GG-LtGMRES method (right) after 17 iterations.

In all computed examples, the regularization parameter and the number of iterations (iter) required by each method are determined with the aid of the discrepancy principle. In this example, we determine the regularization parameter(s) by the bisection method over the interval $[10^{-4}, 10^{5}]$ with $\eta = 1.2$ in (1.5). The reconstructed images determined by the G-LtAT$_p$ and GG-LtGMRES methods are displayed in Figure 2. Here and below, the effectiveness of each chosen method is measured by the relative error

$$E_{\text{method}} = \frac{\|\mathcal{X}_{\text{method}} - \mathcal{X}_{\text{true}}\|_F}{\|\mathcal{X}_{\text{true}}\|_F},$$

where $\mathcal{X}_{\text{method}}$ denotes the computed approximate solution of (1.1) determined by a given method.

Throughout this section, the table entry '-' indicates that a chosen method uses three different numbers of t-Arnoldi/bidiagonalization steps or computes three different values of the regularization parameter, or no regularization parameter or no invertible linear transform is required. Table 2 shows the relative errors in the computed reconstructions, the CPU times required, as well as the regularization parameters and the number of iterations. The GG-LtGMRES method, which works with the whole data tensor $\mathcal{B}$ at a time, can be seen to yield the reconstruction of highest quality when $\widetilde{\delta} = 10^{-3}$ and to require the least CPU time for both noise levels. The Golub-Kahan-Tikhonov-type methods require the most CPU time for both noise levels and determine restorations of the worst quality for $\widetilde{\delta} = 10^{-3}$. The G-LtGKT$_p$ method, which works with the lateral slices of the data tensor independently, requires the most CPU time, and determines for both noise levels restorations of higher quality than the GG-LtGKT method, which works with all lateral slices simultaneously. The GG-LtGKT method requires more iterations than the GG-LtAT method for both noise levels.

Example 5.2. We consider the restoration of the `papav256` image shown in Figure 3 (left) using the regularization operator $\mathcal{L} = \mathcal{I}$. The solution methods of Example 5.1 are compared. We use the DSC matrix to define the linear transformation L. This matrix is not orthogonal; it is the sum of the DCT and the Discrete Sine Transform[1] (DST) matrices from MATLAB. The frontal slices

[1] https://www.mathworks.com/matlabcentral/fileexchange/26040-dct-and-dst-inverse-in-arbitrary-dimension

$\tilde{\delta}$	Method	iter	μ_{iter}	E_{method}	CPU time (secs)
10^{-3}	GG-LtAT	17	$1.57 \cdot 10^{0}$	$6.8580 \cdot 10^{-2}$	$4.55 \cdot 10^{2}$
	GG-LtGKT	51	$6.34 \cdot 10^{0}$	$7.0629 \cdot 10^{-2}$	$9.75 \cdot 10^{3}$
	GG-LtGMRES	17	-	$6.7091 \cdot 10^{-2}$	$2.29 \cdot 10^{2}$
	G-LtAT$_p$	-	-	$6.8706 \cdot 10^{-2}$	$6.46 \cdot 10^{2}$
	G-LtGKT$_p$	-	-	$7.0194 \cdot 10^{-2}$	$2.23 \cdot 10^{4}$
	G-LtGMRES$_p$	-	-	$6.9608 \cdot 10^{-2}$	$5.67 \cdot 10^{2}$
10^{-2}	GG-LtAT	4	$1.22 \cdot 10^{-1}$	$1.0451 \cdot 10^{-1}$	$2.78 \cdot 10^{1}$
	GG-LtGKT	9	$3.57 \cdot 10^{-3}$	$1.0232 \cdot 10^{-1}$	$3.14 \cdot 10^{2}$
	GG-LtGMRES	4	-	$1.0631 \cdot 10^{-1}$	$1.33 \cdot 10^{1}$
	G-LtAT$_p$	-	-	$1.0407 \cdot 10^{-1}$	$5.18 \cdot 10^{1}$
	G-LtGKT$_p$	-	-	$1.0184 \cdot 10^{-1}$	$6.81 \cdot 10^{2}$
	G-LtGMRES$_p$	-	-	$1.0635 \cdot 10^{-1}$	$3.82 \cdot 10^{1}$

Table 2: Results for Example 5.1.

of the blurring operator $\mathcal{A}$ are defined by (5.3) with

$$\mathcal{A}^{(i)} = \frac{1}{\sqrt{2\pi\sigma}} A_1(i,1)A_1, \quad i = 1, 2, \ldots, 256, \quad N = 256, \quad \sigma = 2.5, \quad \texttt{band} = 12.$$

Then $\texttt{cond}(\mathcal{A}^{(i)}) = 1.94 \cdot 10^{8}$ for $i = 1, 2, \ldots 12$, and the condition number of each frontal slice of $\mathcal{A}$ is infinite for $i \geq 13$. We use the discrepancy principle to determine the regularization parameter(s) by the bisection method over the interval $[10^{-2}, 10^{7}]$ with $\eta = 1.1$.

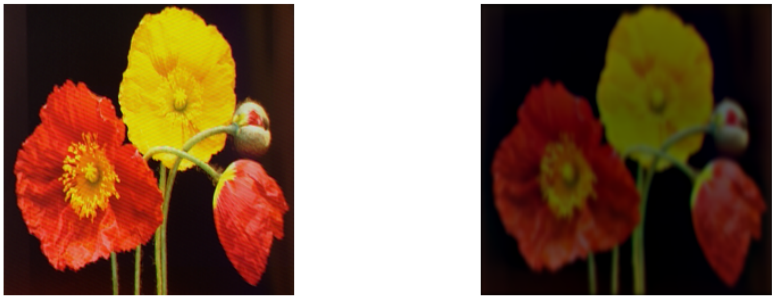

Figure 3: True `papav256` image (left), and blurred and noisy `papav256` image (right) for $\widetilde{\delta} = 10^{-3}$.

Figure 4: Reconstructed images by the GG-LtGKT method (left) and the G-LtGMRES$_p$ method (right) for $\widetilde{\delta} = 10^{-3}$.

The true `papav256` image is of size $256 \times 256 \times 3$ and is stored as a tensor $\mathcal{X}_{\text{true}} \in \mathbb{R}^{256\times3\times256}$ using the `multi_twist` operator. This image is blurred by the tensor $\mathcal{A}$ defined above. Let $\mathcal{B}_{\text{true}} \in \mathbb{R}^{256\times3\times256}$ store the blurred image so obtained. The blurred and noisy image is determined by

$\mathcal{B} = \mathcal{B}_{\text{true}} + \mathcal{E}$, where the "noise tensor" $\mathcal{E} \in \mathbb{R}^{256\times 3\times 256}$ is generated by (5.4). The image represented by $\mathcal{B}$ is shown in Figure 3 (right) using the `multi_squeeze` operator. The images reconstructed by the GG-LtGKT and G-LtGMRES$_p$ methods for $\widetilde{\delta} = 10^{-3}$ are displayed similarly in Figure 4.

$\widetilde{\delta}$	Method	iter	μ_{iter}	E_{method}	CPU time (secs)
10^{-3}	GG-LtAT	10	$1.56 \cdot 10^4$	$5.0166 \cdot 10^{-2}$	$1.34 \cdot 10^2$
	GG-LtGKT	45	$4.15 \cdot 10^4$	$5.0495 \cdot 10^{-2}$	$5.57 \cdot 10^3$
	GG-LtGMRES	10	-	$5.0179 \cdot 10^{-2}$	$5.11 \cdot 10^1$
	G-LtAT$_p$	-	-	$5.0124 \cdot 10^{-2}$	$1.60 \cdot 10^2$
	G-LtGKT$_p$	-	-	$5.0471 \cdot 10^{-2}$	$1.29 \cdot 10^4$
	G-LtGMRES$_p$	-	-	$5.0079 \cdot 10^{-2}$	$1.35 \cdot 10^2$
10^{-2}	GG-LtAT	4	$1.42 \cdot 10^3$	$7.3333 \cdot 10^{-2}$	$2.40 \cdot 10^1$
	GG-LtGKT	10	$2.27 \cdot 10^3$	$6.6785 \cdot 10^{-2}$	$3.62 \cdot 10^2$
	GG-LtGMRES	4	-	$7.3842 \cdot 10^{-2}$	$8.43 \cdot 10^0$
	G-LtAT$_p$	-	-	$7.2670 \cdot 10^{-2}$	$5.14 \cdot 10^1$
	G-LtGKT$_p$	-	-	$6.6788 \cdot 10^{-2}$	$6.79 \cdot 10^2$
	G-LtGMRES$_p$	-	-	$7.4270 \cdot 10^{-2}$	$2.40 \cdot 10^1$

Table 3: Results for Example 5.2.

Relative errors as well as CPU times are shown in Table 3. The GG-LtGKT method yields a restoration of the worst quality for $\widetilde{\delta} = 10^{-3}$, while the G-LtGKT$_p$ method requires the most CPU time for both noise levels. Moreover, the former method requires more iterations than the GG-LtAT method for both noise levels. The solution methods that work with all lateral slices of the data tensor $\mathcal{B}$ simultaneously, i.e., the GG-LtAT, GG-LtGKT, and GG-LtGMRES methods, can be seen to give restorations of worse quality for $\widetilde{\delta} = 10^{-3}$ than methods that work with the lateral slices of $\mathcal{B}$ independently, but the GG-LtGMRES method requires the least CPU time for both noise levels. We will comment more on the relative performance of the methods below.

Example 5.3. This example considers the restoration of the sixth frame of the `Xylophone video` from MATLAB with the regularization tensor $\mathcal{L}_2$. The defining matrix for L is the unnormalized DFT matrix from MATLAB. The regularization parameter is determined by the discrepancy principle and computed by the bisection method over the interval $[10^{-5}, 10^6]$ with $\eta = 1.2$ in (1.5).

Figure 5: True sixth frame (left), and blurred and noisy sixth frame (right) for $\widetilde{\delta} = 10^{-3}$.

The true sixth frame, which is of size $240\times 240\times 3$, is shown in Figure 5 (left). This frame is stored as the tensor $\mathcal{X}_{\text{true}} \in \mathbb{R}^{240\times 3\times 240}$ using the `multi_twist` operator and blurred by $\mathcal{A} \in \mathbb{R}^{240\times 240\times 240}$, which is generated by (5.3) and

$$\texttt{zz} = [\texttt{z(1)}\ \texttt{fliplr(z(end} - \texttt{length(z)} + 2 : \texttt{end))}], \quad A_3 = \texttt{toeplitz(z, zz)},$$

$$\mathcal{A}^{(i)} = \frac{1}{2\pi\sigma^2} A_3(i,1) A_1, \quad i = 1, 2, \ldots, 240, \quad N = 240, \quad \sigma = 3, \quad \texttt{band} = 12,$$

where A_3 is a circulant matrix with $\texttt{cond}(\mathcal{A}^{(i)}) = 1.41 \cdot 10^6$, $i = 1, 2, \ldots, 12$. The remaining frontal slices have infinite condition number. The blurred and noise-contaminated sixth frame is obtained

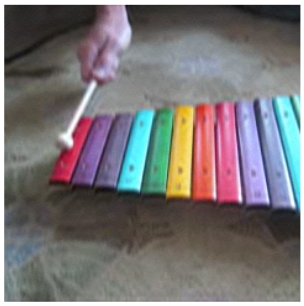
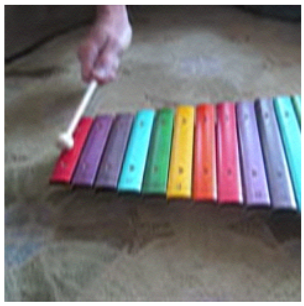

Figure 6: Reconstructed images by the G-LtGKT$_p$ (left) and G-LtGMRES$_p$ (right) methods for $\widetilde{\delta} = 10^{-3}$.

by $\mathcal{B} = \mathcal{A} *_L \mathcal{X}_{\text{true}} + \mathcal{E} \in \mathbb{R}^{240\times 3\times 240}$ and is shown in Figure 5 (right) using the `multi_squeeze` operator, where the "noise tensor" $\mathcal{E} \in \mathbb{R}^{240\times 3\times 240}$ is generated by (5.4).

$\widetilde{\delta}$	Method	iter	μ_{iter}	E_{method}	CPU time (secs)
10^{-3}	GG-LtAT	25	$3.78 \cdot 10^{3}$	$5.1773 \cdot 10^{-2}$	$9.46 \cdot 10^{2}$
	GG-LtGKT	29	$5.14 \cdot 10^{2}$	$4.8805 \cdot 10^{-2}$	$3.39 \cdot 10^{3}$
	GG-LtGMRES	25	-	$5.2055 \cdot 10^{-2}$	$4.93 \cdot 10^{2}$
	G-LtAT$_p$	-	-	$5.1766 \cdot 10^{-2}$	$1.37 \cdot 10^{3}$
	G-LtGKT$_p$	-	-	$4.8806 \cdot 10^{-2}$	$8.15 \cdot 10^{3}$
	G-LtGMRES$_p$	-	-	$5.2060 \cdot 10^{-2}$	$1.17 \cdot 10^{3}$
10^{-2}	GG-LtAT	9	$1.11 \cdot 10^{2}$	$9.8577 \cdot 10^{-2}$	$1.15 \cdot 10^{2}$
	GG-LtGKT	7	$4.17 \cdot 10^{0}$	$7.3944 \cdot 10^{-2}$	$2.14 \cdot 10^{2}$
	GG-LtGMRES	9	-	$1.0711 \cdot 10^{-1}$	$6.08 \cdot 10^{1}$
	G-LtAT$_p$	-	-	$9.8867 \cdot 10^{-2}$	$1.72 \cdot 10^{2}$
	G-LtGKT$_p$	-	-	$7.3932 \cdot 10^{-2}$	$5.20 \cdot 10^{2}$
	G-LtGMRES$_p$	-	-	$1.0495 \cdot 10^{-1}$	$1.42 \cdot 10^{2}$

Table 4: Results for Example 5.3.

The relative errors and CPU times for each method are displayed in Table 4 for two noise levels. Restored images obtained with the G-LtGKT$_p$ and G-LtGMRES$_p$ methods are shown in Figure 6 for $\widetilde{\delta} = 10^{-3}$ using the `multi_squeeze` operator. The GG-LtGKT and G-LtGKT$_p$ methods yield the best restorations and of almost the same quality, but the latter method requires the most CPU time for both noise levels. The G-LtGMRES$_p$ method determines restorations of the worst quality for $\widetilde{\delta} = 10^{-3}$, and the GG-LtGMRES method is the fastest for both noise levels.

Example 5.4. We consider the restoration of the gray-scale analogue of the sixth frame of the `Xylophone video` in Example 5.3. The regularization tensors $\mathcal{L}_1$ and $\mathcal{L}_2$ are used. The quality of the restorations determined by the G-LtAT, G-LtGKT, and G-LtGMRES methods for the solution of (1.8) when the defining matrices for the transform L are the DFT, DCT, and DSC matrices are compared to those determined by the generalized Arnoldi-Tikhonov (GAT) and generalized Golub-Kahan-Tikhonov (G-GKT) regularization methods for the approximate solution of the minimization problem

$$\min_{x\in\mathbb{R}^{240^2}} \left\{ \|(A_1 \otimes A_1)x - b\|_2^2 + \mu^{-1}\|\widehat{L}x\|_2^2 \right\}, \tag{5.5}$$

with the regularization matrices $\widehat{L} = \mathcal{L}_1^{(1)} \in \mathbb{R}^{(240^2-2)\times 240^2}$ defined by (5.1) or $\widehat{L} = \mathcal{L}_2^{(1)} \in \mathbb{R}^{(240^2-1)\times 240^2}$ given by (5.2). The vectors $x = \texttt{vec}(\vec{\mathcal{X}})$ and $b = \texttt{vec}(\vec{\mathcal{B}})$ are defined by stacking the faces of the tensor columns $\vec{\mathcal{X}}$ and $\vec{\mathcal{B}} \in \mathbb{R}^{240\times 1\times 240}$ in order, respectively. The solution of (5.5) for $\mu = \infty$ is determined by the standard GMRES method. The G-GKT and GAT methods are implemented similarly as described in Sections 3 and 4, respectively.

The blurring operator $\mathcal{A} \in \mathbb{R}^{240\times 240\times 240}$ is defined by using (5.3) with frontal slices

$$\mathcal{A}^{(i)} = \frac{1}{2\pi\sigma^2} A_1(i,1)A_1, \quad i = 1, 2, \ldots, 240, \quad N = 240, \quad \sigma = 2.5, \quad \texttt{band} = 12,$$

where $\texttt{cond}(\mathcal{A}^{(i)}) = 1.35 \cdot 10^7$ for $i = 1, 2, \ldots, 12$. The condition number of the remaining frontal slices is infinite. The regularization parameter μ is determined by the bisection method over the interval $[10^{-5}, 10^7]$ using the discrepancy principle with $\eta = 1.01$.

The true sixth frame of the `Xylophone video`, shown in Figure 7, is stored as the tensor column $\vec{\mathcal{X}}_{\text{true}} \in \mathbb{R}^{240\times 1\times 240}$ and blurred by $\mathcal{A}$. Blurred and noisy images corresponding to different $*_L$ tensor products are shown in Figures 7 and 8 using the `squeeze` operator. These images are generated according to $\vec{\mathcal{B}} = \mathcal{A} *_L \vec{\mathcal{X}}_{\text{true}} + \vec{\mathcal{E}}$, where the noise tensor $\vec{\mathcal{E}}$ is determined analogously to (5.4). The blurred and noisy image for (5.5) is generated by $b = (A_1 \otimes A_1)\texttt{vec}(\vec{\mathcal{X}}_{\text{true}}) + \texttt{vec}(\vec{\mathcal{E}})$ and is shown in Figure 7 (middle) by using the MATLAB `reshape` operator.

The intensity of the blur is different for (5.5), and also for each tensor product $*_L$. Specifically, the blur- and noise-contaminated image produced for (5.5), and also for (1.8) when the tensor product $*_L$ is defined by the DCT matrix, is more blurred than when the DFT and DSC matrices are used to define the $*_L$ product. The reconstructed video frames determined by the G-LtGKT method for $\mathcal{L} = \mathcal{L}_2$ and shown in Figure 9 using the `squeeze` operator correspond to the $*_L$ products defined by the DFT, DCT, and DSC matrices, and the noise level $\widetilde{\delta} = 10^{-3}$. The reconstructed video frame by the G-GKT method is displayed on the right-hand side of Figure 8 using the MATLAB command `reshape`.

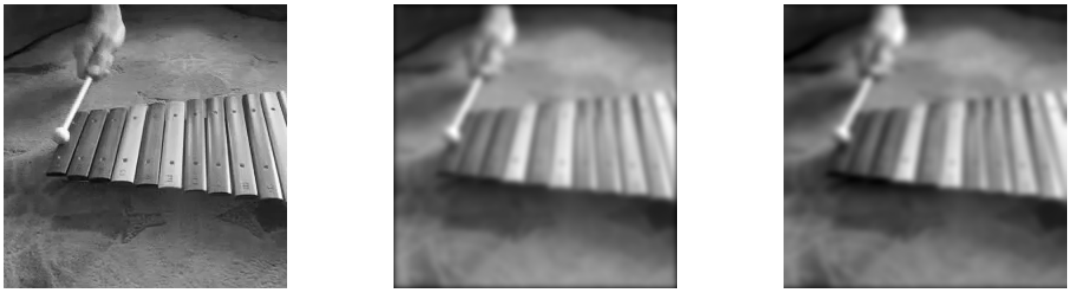

Figure 7: True sixth `Xylophone video` frame (left), blurred and noisy sixth frames generated for (5.5) (middle), and when L is defined by DCT matrix (right) for $\widetilde{\delta} = 10^{-3}$.

Figure 8: Blurred and noisy sixth frames generated when L is defined by DFT matrix (left), DSC matrix (middle), and reconstructed video frame by the G-GKT method after 69 iterations for $\widetilde{\delta} = 10^{-3}$.

Table 5 shows the number of iterations required by each method to satisfy the discrepancy principle, the computed regularization parameters, as well as the relative errors and the CPU times. The table shows that the performance of the methods depends on the invertible linear transform used. In particular, the G-LtGKT method yields restoration of the highest quality for both noise levels when the defining matrices for L are the DFT or DSC matrices, and always gives near-best restorations. This method requires the most CPU time for both noise levels, all invertible

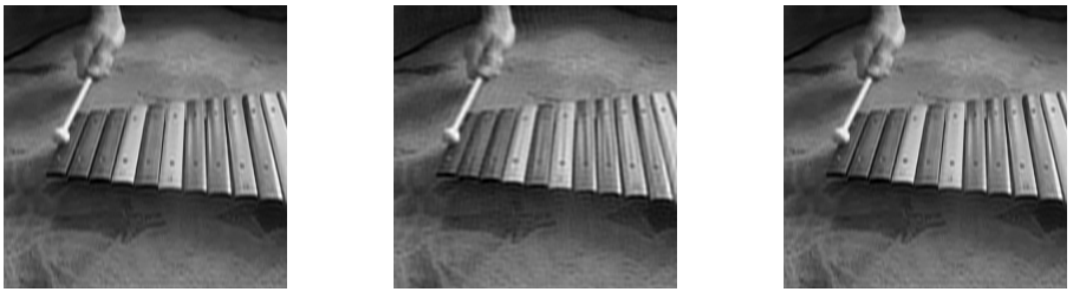

Figure 9: Reconstructed sixth frame by the G-LtGKT method for $\widetilde{\delta} = 10^{-3}$ when the defining matrix for L is DFT matrix (left) after 35 iterations, the DCT matrix (middle) after 116 iterations, and the DSC matrix (right) after 53 iterations.

linear transforms and regularization operators considered. Moreover, for $\widetilde{\delta} = 10^{-3}$ and for any of the invertible linear transforms considered, the G-LtGKT method with $\mathcal{L} = \mathcal{L}_2$ yields restorations of the higher quality than when $\mathcal{L} = \mathcal{L}_1$ is used. The relative performance is reversed for $\widetilde{\delta} = 10^{-2}$. The GMRES method is the fastest for both noise levels but gives the worst quality restoration for $\widetilde{\delta} = 10^{-3}$. Independently of the invertible linear transform L, the regularization operator, and the noise level, the G-LtGKT method is the slowest and yields restorations of the best or near-best quality, while the GMRES method is the fastest and yields restorations of worst or near-worst quality. In general, the "classical" approaches, e.g., the GAT, G-GKT and GMRES methods, yield the worst quality restorations. Their performance may depend on the intensity of the blur.

Example 5.5. This example discusses the restoration of the `flower` image[2] with the regularization tensor $\mathcal{L}_1$. We compare the quality of restorations determined by the GG-LtAT, G-LtAT$_p$, GG-LtGMRES, and G-LtGMRES$_p$ methods with the transform L defined by the DFT, DCT, or DSC matrices. Examples 5.1-5.4 show the GG-LtGKT and G-LtGKT$_p$ methods to be slow, and therefore are not considered here. We use the normalized DFT matrix for L in this example. Numerical experiments suggest that we can save some computing time by using this matrix instead of the unnormalized DFT matrix of Examples 5.3 and 5.4. The blurring operator $\mathcal{A} \in \mathbb{R}^{300\times300\times300}$ is generated similarly as in Example 5.2 with $\sigma = 3$ and `band` $= 12$. The condition number of the first 12 frontal slices $\mathcal{A}^{(i)}$ is $7.58 \cdot 10^8$; the remaining frontal slices have infinite condition numbers. Using the discrepancy principle (1.5) with $\eta = 1.2$, we determine the regularization parameter(s) by the bisection method over the interval $[10^{-2}, 10^5]$. The true `flower` image, shown in Figure 10, is stored as the tensor $\mathcal{X}_{\text{true}} \in \mathbb{R}^{300\times3\times300}$ using the `multi_twist` operator and is blurred by $\mathcal{A}$ similarly as described above.

Figure 10: True `flower` image.

The blurred and noisy images $\mathcal{B} \in \mathbb{R}^{300\times3\times300}$ associated with the DFT, DCT, and DSC matrices are shown in Figure 11 for $\widetilde{\delta} = 10^{-3}$. These images are generated by $\mathcal{B} = \mathcal{A} *_L \mathcal{X}_{\text{true}} + \mathcal{E}$, where $\mathcal{E} \in \mathbb{R}^{300\times3\times300}$ is a "noise tensor" determined by (5.4). The images restored by the G-LtAT$_p$

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

$\tilde{\delta}$	$\mathcal{L}$	Transform	Method	iter	μ_{iter}	E_{method}	CPU time (secs)
10^{-3}	$\mathcal{L}_1$	DFT	G-LtAT	25	$3.66 \cdot 10^3$	$4.6321 \cdot 10^{-2}$	$4.30 \cdot 10^2$
			G-LtGKT	35	$1.83 \cdot 10^2$	$4.1062 \cdot 10^{-2}$	$3.75 \cdot 10^3$
		DCT	G-LtAT	30	$8.49 \cdot 10^2$	$6.5542 \cdot 10^{-2}$	$3.14 \cdot 10^2$
			G-LtGKT	116	$1.75 \cdot 10^3$	$6.5530 \cdot 10^{-2}$	$1.78 \cdot 10^4$
		DSC	G-LtAT	11	$2.25 \cdot 10^3$	$4.0409 \cdot 10^{-2}$	$4.78 \cdot 10^1$
			G-LtGKT	53	$5.38 \cdot 10^3$	$3.9429 \cdot 10^{-2}$	$3.59 \cdot 10^3$
	$\mathcal{L}_2$	DFT	G-LtAT	25	$5.18 \cdot 10^3$	$4.6322 \cdot 10^{-2}$	$4.49 \cdot 10^2$
			G-LtGKT	35	$1.16 \cdot 10^3$	$4.1036 \cdot 10^{-2}$	$3.94 \cdot 10^3$
		DCT	G-LtAT	30	$3.58 \cdot 10^3$	$6.5510 \cdot 10^{-2}$	$3.27 \cdot 10^2$
			G-LtGKT	116	$8.16 \cdot 10^3$	$6.5515 \cdot 10^{-2}$	$1.78 \cdot 10^4$
		DSC	G-LtAT	11	$3.57 \cdot 10^3$	$4.0418 \cdot 10^{-2}$	$4.92 \cdot 10^1$
			G-LtGKT	53	$3.29 \cdot 10^3$	$3.9409 \cdot 10^{-2}$	$3.74 \cdot 10^3$
	$\mathcal{L}_1^{(1)}$	-	GAT	12	$5.48 \cdot 10^4$	$7.9158 \cdot 10^{-2}$	$6.82 \cdot 10^1$
			G-GKT	69	$5.86 \cdot 10^3$	$7.2786 \cdot 10^{-2}$	$2.60 \cdot 10^2$
	$\mathcal{L}_2^{(1)}$	-	AT	12	$1.88 \cdot 10^4$	$7.9162 \cdot 10^{-2}$	$6.47 \cdot 10^1$
			GKT	69	$4.84 \cdot 10^3$	$7.2778 \cdot 10^{-2}$	$2.39 \cdot 10^2$
		DFT	G-LtGMRES	25	-	$4.7058 \cdot 10^{-2}$	$3.89 \cdot 10^2$
		DCT	G-LtGMRES	30	-	$6.5131 \cdot 10^{-2}$	$3.03 \cdot 10^2$
		DSC	G-LtGMRES	11	-	$4.0871 \cdot 10^{-2}$	$4.22 \cdot 10^1$
		-	GMRES	12	-	$8.1534 \cdot 10^{-2}$	$4.17 \cdot 10^0$
10^{-2}	$\mathcal{L}_1$	DFT	G-LtAT	9	$1.55 \cdot 10^3$	$1.0185 \cdot 10^{-1}$	$6.32 \cdot 10^1$
			G-LtGKT	16	$1.09 \cdot 10^0$	$5.9297 \cdot 10^{-2}$	$3.04 \cdot 10^2$
		DCT	G-LtAT	6	$5.06 \cdot 10^2$	$1.1994 \cdot 10^{-1}$	$1.59 \cdot 10^1$
			G-LtGKT	16	$5.38 \cdot 10^0$	$9.6204 \cdot 10^{-2}$	$3.63 \cdot 10^2$
		DSC	G-LtAT	5	$1.69 \cdot 10^2$	$6.8818 \cdot 10^{-2}$	$1.21 \cdot 10^1$
			G-LtGKT	14	$2.17 \cdot 10^0$	$5.6424 \cdot 10^{-2}$	$2.78 \cdot 10^2$
	$\mathcal{L}_2$	DFT	G-LtAT	9	$2.06 \cdot 10^3$	$1.0185 \cdot 10^{-1}$	$7.04 \cdot 10^1$
			G-LtGKT	9	$1.13 \cdot 10^1$	$5.9110 \cdot 10^{-2}$	$3.06 \cdot 10^2$
		DCT	G-LtAT	6	$6.81 \cdot 10^2$	$1.1995 \cdot 10^{-1}$	$1.65 \cdot 10^1$
			G-LtGKT	16	$4.33 \cdot 10^1$	$9.6128 \cdot 10^{-2}$	$3.60 \cdot 10^2$
		DSC	G-LtAT	5	$2.33 \cdot 10^2$	$6.8827 \cdot 10^{-2}$	$1.50 \cdot 10^1$
			G-LtGKT	14	$2.32 \cdot 10^1$	$5.6317 \cdot 10^{-2}$	$2.79 \cdot 10^2$
	$\mathcal{L}_1^{(1)}$	-	GAT	5	$2.06 \cdot 10^3$	$1.1193 \cdot 10^{-1}$	$6.97 \cdot 10^0$
			G-GKT	13	$1.12 \cdot 10^2$	$9.2457 \cdot 10^{-2}$	$8.93 \cdot 10^0$
	$\mathcal{L}_2^{(1)}$	-	GAT	5	$6.91 \cdot 10^2$	$1.1193 \cdot 10^{-1}$	$6.54 \cdot 10^0$
			G-GKT	13	$9.78 \cdot 10^1$	$9.2424 \cdot 10^{-2}$	$8.95 \cdot 10^0$
		DFT	G-LtGMRES	9	-	$1.0287 \cdot 10^{-1}$	$5.47 \cdot 10^1$
		DCT	G-LtGMRES	6	-	$1.3644 \cdot 10^{-1}$	$1.31 \cdot 10^1$
		DSC	G-LtGMRES	5	-	$7.3642 \cdot 10^{-2}$	$9.32 \cdot 10^0$
		-	GMRES	5	-	$1.2654 \cdot 10^{-1}$	$0.81 \cdot 10^0$

Table 5: Results for Example 5.4.

method, that correspond to the above transforms, are displayed in Figure 12 for $\widetilde{\delta} = 10^{-3}$ using the `multi_squeeze` operator.

Table 6 shows the relative errors and CPU times for each method. Among the GMRES-type methods, the G-LtGMRES$_p$ method, which works with the lateral slices of $\mathcal{B}$ independently, is seen to yield restorations of near-best quality for both noise levels and for all transforms con-

Figure 11: Blurred and noisy `flower` image generated when L is defined by the DFT matrix (left), the DCT matrix (middle), and the DSC matrix (right) for $\widetilde{\delta} = 10^{-3}$.

Figure 12: Reconstructed images by the G-LtAT$_p$ method for $\widetilde{\delta} = 10^{-3}$ when the defining matrix for L is the DFT matrix (left) after 29 iterations, the DCT matrix (middle) after 16 iterations, and the DSC matrix (right) after 10 iterations.

sidered. Moreover, the GG-LtGMRES method is faster than the G-LtGMRES$_p$, GG-LtAT, and G-LtAT$_p$ methods. A similar behavior also can be noted for the Arnoldi-Tikhonov-type methods. These observations are consistent with our findings in [27, 28]. The G-LtAT$_p$ method, which works independently with the lateral slices of the data tensor, is seen to yield restorations of near-best quality, but is the slowest method for both noise levels and for all transforms considered. The GG-LtGMRES method yields restorations of near-worst quality.

6 Conclusions

This paper discusses several transform-based methods for solving linear discrete ill-posed tensor problems and extends available global tensor Krylov subspace methods defined by a t-product to global tensor Krylov subspace methods defined by an invertible linear transform tensor product $*_L$ introduced by Kernfeld et al. [16]. The latter tensor product and its performance in tensor Krylov subspace iterative methods has not received much attention in the literature.

Both orthogonal and non-orthogonal invertible linear transform matrices are considered. The blurring effects and the performance of the described methods depend on the transformation used.

We found Golub-Kahan-type bidiagonalization methods, i.e., the G-LtGKT, GG-LtGKT, and G-LtGKT methods, in general to be slow. Their performance is sensitive to the noise levels used. Independently of the regularization operator, the G-LtGKT$_p$ and GG-LtGKT methods yield the worst quality restorations for 0.1% noise when the DCT and DSC matrices are used to define the $*_L$ product. Moreover, they are the best methods when the DFT is used to define $*_L$ product. Additionally, irrespective of the choice of regularization operator and invertible linear transform, both methods give restorations of the best quality for 1% noise. The GG-LtGKT method, which works with the lateral slices of the data tensor simultaneously, yields worse or nearly worse quality restorations for both noise levels than the G-LtGKT$_p$ method, which works with the lateral slices independently. The latter method is slower than the GG-LtGKT method.

$\tilde{\delta}$	Transform	Method	iter	μ_{iter}	E_{method}	CPU time (secs)
10^{-3}	DFT	GG-LtAT	29	$1.33 \cdot 10^{1}$	$6.5252 \cdot 10^{-2}$	$3.09 \cdot 10^{3}$
		GG-LtGMRES	29	-	$6.6161 \cdot 10^{-2}$	$1.55 \cdot 10^{3}$
		G-LtAT$_p$	-	-	$6.5239 \cdot 10^{-2}$	$4.25 \cdot 10^{3}$
		G-LtGMRES$_p$	-	-	$6.5836 \cdot 10^{-2}$	$3.69 \cdot 10^{3}$
	DCT	GG-LtAT	16	$2.58 \cdot 10^{2}$	$8.8341 \cdot 10^{-2}$	$4.13 \cdot 10^{2}$
		GG-LtGMRES	16	-	$9.0044 \cdot 10^{-2}$	$2.02 \cdot 10^{2}$
		G-LtAT$_p$	-	-	$8.4814 \cdot 10^{-2}$	$5.40 \cdot 10^{2}$
		G-LtGMRES$_p$	-	-	$8.6098 \cdot 10^{-2}$	$4.74 \cdot 10^{2}$
	DSC	GG-LtAT	10	$8.74 \cdot 10^{0}$	$5.8140 \cdot 10^{-2}$	$1.62 \cdot 10^{2}$
		GG-LtGMRES	10	-	$5.7754 \cdot 10^{-2}$	$8.00 \cdot 10^{1}$
		G-LtAT$_p$	-	-	$5.8071 \cdot 10^{-2}$	$2.88 \cdot 10^{2}$
		G-LtGMRES$_p$	-	-	$5.7581 \cdot 10^{-2}$	$2.40 \cdot 10^{2}$
10^{-2}	DFT	GG-LtAT	8	$4.10 \cdot 10^{-1}$	$1.0306 \cdot 10^{-1}$	$2.22 \cdot 10^{2}$
		GG-LtGMRES	8	-	$1.1034 \cdot 10^{-1}$	$1.09 \cdot 10^{2}$
		G-LtAT$_p$	-	-	$1.0324 \cdot 10^{-1}$	$3.25 \cdot 10^{2}$
		G-LtGMRES$_p$	-	-	$1.0653 \cdot 10^{-1}$	$2.55 \cdot 10^{2}$
	DCT	GG-LtAT	4	$1.30 \cdot 10^{-1}$	$1.0527 \cdot 10^{-1}$	$2.95 \cdot 10^{1}$
		GG-LtGMRES	4	-	$1.0669 \cdot 10^{-1}$	$1.37 \cdot 10^{1}$
		G-LtAT$_p$	-	-	$1.0511 \cdot 10^{-1}$	$5.29 \cdot 10^{1}$
		G-LtGMRES$_p$	-	-	$1.0670 \cdot 10^{-1}$	$3.87 \cdot 10^{1}$
	DSC	GG-LtAT	4	$2.55 \cdot 10^{-1}$	$8.3032 \cdot 10^{-2}$	$2.70 \cdot 10^{1}$
		GG-LtGMRES	4	-	$8.3178 \cdot 10^{-2}$	$1.32 \cdot 10^{1}$
		G-LtAT$_p$	-	-	$8.2905 \cdot 10^{-2}$	$5.07 \cdot 10^{1}$
		G-LtGMRES$_p$	-	-	$8.3190 \cdot 10^{-2}$	$3.75 \cdot 10^{1}$

Table 6: Results for Example 5.5.

The G-LtGKT method is seen to give restorations of higher quality than the G-LtAT and G-LtGMRES methods. Its performance depends on the invertible linear transform, regularization operator, and noise level used. It is often the case that the quality of the restoration improves with Tikhonov regularization. However, this behavior is different when the DCT matrix is used to define the $*_L$ product, since for 0.1% noise and independently of the choice of regularization operator for the G-LtAT method, the G-LtGMRES method gives higher quality restoration.

The performances of the G-LtGMRES, G-LtGMRES$_p$, and GG-LtGMRES methods are almost independent of the transform used and the noise levels. The G-LtGMRES and GG-LtGMRES methods are the fastest but often yield restorations of the worst or near-worst quality.

Among the Arnoldi-type methods applied to color and video image processing, i.e., G-LtAT$_p$, GG-LtAT, G-LtGMRES$_p$, and GG-LtGMRES methods, the G-LtAT$_p$ method, which works with the lateral slices of the data tensor independently, is the best or near-best method when the DCT and DFT matrices are used to define the $*_L$ product. Similarly, the G-LtGMRES$_p$ method gives the best or near-best quality restoration when the DSC matrix is used to define the $*_L$ product.

The performance of the Golub-Kahan bidiagonalization-type and GMRES-type methods leads us to recommend the use of Tikhonov regularization methods together with Arnoldi-type methods for the reduction of a large problem to a problem of fairly small dimension. Though, we have to add that Arnoldi-type reduction methods may perform poorly for pronounced motion blur. The latter has been illustrated for matrix problems in [5].

Acknowledgment

The authors would like to thank the referees for comments that lead to an improved presentation. Work by LR was supported in part by NSF grant DMS-1720259.

References

[1] M. Baburaj and S. N. George, DCT based weighted adaptive multi-linear data completion and denoising, Neurocomputing, 318 (2018), pp. 120–136.

[2] F. P. A. Beik, K. Jbilou, M. Najafi-Kalyani, and L. Reichel, Golub-Kahan bidiagonalization for ill-conditioned tensor equations with applications, Numer. Algorithms, 84 (2020), pp. 1535–1563.

[3] F. P. A. Beik, M. Najafi-Kalyani, and L. Reichel, Iterative Tikhonov regularization of tensor equations based on the Arnoldi process and some of its generalizations, Appl. Numer. Math., 151 (2020), pp. 425–447.

[4] D. Calvetti, B. Lewis, and L. Reichel, On the regularizing properties of the GMRES method, Numer. Math., 91 (2002), pp. 605–625.

[5] M. Donatelli, D. Martin, and L. Reichel, Arnoldi methods for image deblurring with anti-reflective boundary conditions, Appl. Math. Comput., 253 (2015), pp. 135–150.

[6] M. El Guide, A. El Ichi, K. Jbilou, and F. P. A Beik, Tensor GMRES and Golub-Kahan bidiagonalization methods via the Einstein product with applications to image and video processing, https://arxiv.org/pdf/2006.07133.pdf

[7] M. El Guide, A. El Ichi, K. Jbilou, and R. Sadaka, Tensor Krylov subspace methods via the T-product for color image processing, June 2020. https://arxiv.org/pdf/2006.07133.pdf

[8] H. W. Engl, M. Hanke, and A. Neubauer, Regularization of Inverse Problems, Kluwer, Dordrecht, 1996.

[9] C. Fenu, L. Reichel, and G. Rodriguez, GCV for Tikhonov regularization via global Golub-Kahan decomposition, Numer. Linear Algebra Appl., 23 (2016), pp. 467–484.

[10] G. H. Golub, M. Heath, and G. Wahba, Generalized cross-validation as a method for choosing a good ridge parameter, Technometrics, 21 (1979), pp. 215–223.

[11] J. Han, p-order tensor products with invertible linear transforms, May 2020. https://arxiv.org/abs/2004.11477

[12] P. C. Hansen, Regularization tools version 4.0 for MATLAB 7.3. Numer. Algorithms, 46 (2007), pp. 189–194.

[13] P. C. Hansen, Analysis of discrete ill-posed problems by means of the L-curve, SIAM Rev., 34 (1992), pp. 561–580.

[14] P. C. Hansen, J. G. Nagy, and D. P. OLeary, Deblurring Images: Matrices, Spectra, and Filtering. Society for Industrial and Applied Mathematics, 2006.

[15] G. Huang, L. Reichel, and F. Yin, On the choice of subspace for large-scale Tikhonov regularization problems in general form, Numer. Algorithms, 81 (2019), pp. 33–55.

[16] E. Kernfeld, M. Kilmer, and S. Aeron, Tensor-tensor products with invertible linear transforms, Linear Algebra Appl., 485 (2015), pp. 545–570.

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

[18] M. E. Kilmer, L. Horesh, H. Avron, and E. Newman, Tensor-tensor products for optimal representation and compression, 2020. https://arxiv.org/abs/2001.00046

[19] M. E. Kilmer and C. D. Martin, Factorization strategies for third-order tensors, Linear Algebra Appl., 434 (2011), pp. 641–658.

[20] S. Kindermann and K. Raik, A simplified L-curve method as error estimator, Electron. Trans. Numer. Anal., 53 (2020), pp. 217–238.

[21] T. G. Kolda and B. W. Bader, Tensor decompositions and applications, SIAM Rev., 51 (2009), pp. 455–500.

[22] C. Lu, X. Peng, and Y. Wei, Low-rank tensor completion with a new tensor nuclear norm induced by invertible linear transforms, in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2019, pp. 5996–6004.

[23] A. Neubauer, Augmented GMRES-type versus CGNE methods for the solution of linear ill-posed problems, Electron. Trans. Numer. Anal., 51 (2019), pp. 412–431.

[24] A. Neuman, L. Reichel, and H. Sadok, Implementations of range restricted iterative methods for linear discrete ill-posed problems, Linear Algebra Appl., 436 (2012), pp. 3974–3990.

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

[26] L. Reichel and G. Rodriguez, Old and new parameter choice rules for discrete ill-posed problems, Numer. Algorithms, 63 (2013), pp. 65–87.

[27] L. Reichel and U. O. Ugwu, Tensor Golub-Kahan-Tikhonov methods applied to the solution of ill-posed problem with a t-product structure, 2020, submitted for publication.

[28] L. Reichel and U. O. Ugwu, Tensor Arnoldi-Tikhonov and GMRES-type methods for ill-posed problem with t-product structure, 2020, submitted for publication.