The tensor Golub-Kahan-Tikhonov method applied to the solution of ill-posed problems with a t-product structure

Lothar Reichel* and Ugochukwu O. Ugwu†

Department of Mathematical Sciences, Kent State University, OH 44242

Abstract

This paper discusses an application of partial tensor Golub-Kahan bidiagonalization to the solution of large-scale linear discrete ill-posed problems based on the t-product formalism for third order tensors proposed by [M. E. Kilmer and C. D. Martin, Factorization strategies for third order tensors, Linear Algebra Appl., 435 (2011), pp. 641-658]. The solution methods presented first reduce a given (large-scale) problem to a problem of small size by application of a few steps of tensor Golub-Kahan bidiagonalization and then regularize the reduced problem so obtained by Tikhonov's method. The regularization operator is a third order tensor, and the data may be represented by a matrix, i.e., a tensor slice, or by a general third order tensor. A regularization parameter is determined by the discrepancy principle. This results in fully automatic solution methods that neither require a user to choose the number of bidiagonalization steps nor the regularization parameter. The methods presented extend available methods for the solution of linear discrete ill-posed problems defined by a matrix operator to linear discrete ill-posed problems defined by a third order tensor operator. An interlacing property of singular tubes for third order tensors is shown and applied. Several algorithms are presented. Computed examples illustrate the advantage of the tensor t-product approach, in comparison with solution methods that are based on matricization of the tensor equation.

Key words: discrepancy principle, discrete ill-posed problem, tensor Golub-Kahan bidiagonalization, Tikhonov regularization, t-product

1 Introduction

This paper is concerned with the solution of linear discrete ill-posed problem of the form

$$\min_{\vec{\mathcal{X}}\in\mathbb{R}^{m\times 1\times n}} \|\mathcal{A}*\vec{\mathcal{X}}-\vec{\mathcal{B}}\|_F, \tag{1.1}$$

where $\mathcal{A}=[a_{ijk}]_{i,j,k=1}^{\ell,m,n}\in\mathbb{R}^{\ell\times m\times n}$ is a third order tensor, $\vec{\mathcal{X}}\in\mathbb{R}^{m\times 1\times n}$ and $\vec{\mathcal{B}}\in\mathbb{R}^{\ell\times 1\times n}$ are lateral slices of third order tensors and may be thought of as laterally oriented matrices. The operation $*$ denotes the tensor t-product introduced in the seminal work by Kilmer and Martin [22] and applied to image deblurring problems by Kilmer et al. [21, 22]. The t-product between $\mathcal{A}$ and $\vec{\mathcal{X}}$, which will be defined below, is computed by transforming both tensors into the Fourier domain along the third dimension, evaluating n matrix-vector products in the Fourier domain, and computing the inverse Fourier transform of the result. Finally, $\|\cdot\|_F$ denotes the tensor Frobenius norm, i.e.,

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

An extension of (1.1) to higher-order tensors and a suitably defined t-product is described by Martin et al. [27]. We will not consider this generalization in the present paper. However, we will allow $\vec{\mathcal{B}}$ and $\vec{\mathcal{X}}$ in (1.1) to be general third order tensors in Subsections 4.2 and 5.1.

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

† e-mail: uugwu@kent.edu

In the problem (1.1) considered in this paper, the tensor $\mathcal{A}$ is of ill-determined tubal rank, i.e., it is difficult to define the tubal rank of $\mathcal{A}$ in a meaningful way. Specifically, the Frobenius norm of the singular tubes of $\mathcal{A}$, which are the analogues of the singular values of a matrix, decay quite rapidly to zero with increasing index number, and there are many nonvanishing singular tubes of tiny Frobenius norm of different orders of magnitude. Then the minimization problem (1.1) is a linear discrete ill-posed problem.

In applications of interest to us, the tensor $\vec{\mathcal{B}}$ represents measured data that is contaminated by measurement error, which is represented by a tensor $\vec{\mathcal{E}} \in \mathbb{R}^{\ell\times 1\times n}$. Straightforward solution of (1.1) generally is not meaningful due to propagation and severe amplification of the error $\vec{\mathcal{E}}$ into the solution of (1.1). We reduce this difficulty by applying Tikhonov regularization, i.e., we replace the minimization problem (1.1) by the penalized least-squares problem

$$\min_{\vec{\mathcal{X}}\in\mathbb{R}^{m\times 1\times n}} \{\|\mathcal{A}*\vec{\mathcal{X}}-\vec{\mathcal{B}}\|_F^2+\mu^{-1}\|\mathcal{L}*\vec{\mathcal{X}}\|_F^2\}, \tag{1.2}$$

where $\mathcal{L}\in\mathbb{R}^{s\times m\times n}$ is a regularization operator and $\mu>0$ is a regularization parameter. Our reason for using μ^{-1} in (1.2) instead of μ will be commented on below.

Let $\mathcal{N}(\mathcal{M})$ denote the null space of the tensor $\mathcal{M}$ under $*$. We assume that $\mathcal{L}$ is such that

$$\mathcal{N}(\mathcal{A})\cap\mathcal{N}(\mathcal{L})=\{\vec{\mathcal{O}}\}, \tag{1.3}$$

where $\vec{\mathcal{O}}\in\mathbb{R}^{m\times 1\times n}$ is an $m\times n$ zero matrix oriented laterally; see below. Then (1.2) has a unique solution $\vec{\mathcal{X}}_\mu\in\mathbb{R}^{m\times 1\times n}$ for any $\mu>0$; cf. Theorem 3.1 below.

We refer to the solution scheme for (1.2) described in this paper as the t-product Golub-Kahan-Tikhonov (tGKT) regularization method. This method is based on first reducing $\mathcal{A}$ to a small bidiagonal tensor by carrying out a few, say $1\leq k\ll\min\{\ell,m\}$, steps of the t-product Golub-Kahan bidiagonalization (tGKB) process discussed by Kilmer et al. [21]. This process, generically, furnishes an orthonormal basis for a k-dimensional tensor Krylov (t-Krylov) subspace

$$\mathcal{K}_k(\mathcal{A}^T*\mathcal{A},\mathcal{A}^T*\vec{\mathcal{B}})=\text{span}\{\mathcal{A}^T*\vec{\mathcal{B}},(\mathcal{A}^T*\mathcal{A})*\mathcal{A}^T*\vec{\mathcal{B}},\ldots,(\mathcal{A}^T*\mathcal{A})^{k-1}*\mathcal{A}^T*\vec{\mathcal{B}}\}, \tag{1.4}$$

where the superscript ^{T} denotes transposition. Each step of the tGKB process requires the evaluation of two tensor-matrix products, one with $\mathcal{A}$ and one with $\mathcal{A}^T$. An approximate solution of (1.2) in the t-Krylov subspace (1.4) can be computed quite rapidly also for large tensors $\mathcal{A}$ and $\vec{\mathcal{B}}$. Our solution methods for (1.1) differ from the one described by Kilmer et al. [21] in four ways: i) we use the tGKB process, ii) we allow a general regularization operator $\mathcal{L}$, iii) the regularization parameter $\mu>0$ is determined with the aid of the discrepancy principle, and iv) we allow the special data tensor $\vec{\mathcal{B}}$ to be replaced by a general third order tensor $\mathcal{B}\in\mathbb{R}^{\ell\times p\times n}$, $p>1$.

In the remainder of this section, we provide some details on the topics iii) and iv), starting with the former. Assume that the data tensor $\vec{\mathcal{B}}$ is contaminated by an error $\vec{\mathcal{E}}\in\mathbb{R}^{\ell\times 1\times n}$, which we will refer to as "noise", i.e.,

$$\vec{\mathcal{B}}=\vec{\mathcal{B}}_{\text{true}}+\vec{\mathcal{E}}, \tag{1.5}$$

where $\vec{\mathcal{B}}_{\text{true}}\in\mathbb{R}^{\ell\times 1\times n}$ denotes the unknown error-free data tensor associated with $\vec{\mathcal{B}}$. Assume that the (unavailable) system of equations

$$\mathcal{A}*\vec{\mathcal{X}}=\vec{\mathcal{B}}_{\text{true}}$$

is consistent and let $\vec{\mathcal{X}}_{\text{true}}\in\mathbb{R}^{m\times 1\times n}$ denote the unique solution of minimal Frobenius norm. Let a bound

$$\|\vec{\mathcal{E}}\|_F\leq\delta \tag{1.6}$$

be known. The discrepancy principle prescribes that the regularization parameter $\mu>0$ be determined so that the solution $\vec{\mathcal{X}}_\mu$ of (1.2) satisfies

$$\|\mathcal{A}*\vec{\mathcal{X}}_\mu-\vec{\mathcal{B}}\|_F=\eta\delta, \tag{1.7}$$

where $\eta>1$ is a user-specified constant that is independent of δ. It can be shown that $\vec{\mathcal{X}}_\mu\rightarrow\vec{\mathcal{X}}_{\text{true}}$ as $\delta\searrow 0$. The discrepancy principle and its properties are discussed in a Hilbert space setting, e.g., by Engl et al. [13]. The idea behind the discrepancy principle can easily be extended to the tensor setting using the t-product formalism; cf. (3.24).

We remark that many other techniques for determining the regularization parameter are available; see, e.g., [14, 15, 23, 24, 29]. Some of these methods may be attractive to apply in the present setting when no bound (1.6) is known or can be estimated.

We also describe an alternative to the tGKT method for the solution of (1.2), which we will refer to as the global tGKT (G-tGKT) method. This method works with the data tensor slice $\vec{\mathcal{B}} \in \mathbb{R}^{m\times 1\times n}$ and is analogous to the T-global Golub-Kahan-Tikhonov regularization method recently described by El Guide et al. [11], which works with a general third order data tensor $\mathcal{B} \in \mathbb{R}^{m\times p\times n}$, $p > 1$. Differently from the tGKT method, both the G-tGKT method and the method discussed by El Guide et al. [11] involve matricization or "flattening" of the tensor $\mathcal{A}$. The G-tGKT method first reduces the tensor $\mathcal{A}$ in (1.2) to a small lower bidiagonal matrix by carrying out a few steps of the global t-product Golub-Kahan bidiagonalization (G-tGKB) process. This process differs from the tGKB process in the choice of normalization of $\vec{\mathcal{B}}$ and the resulting decompositions. In particular, the tGKB process produces a lower bidiagonal tensor whose entries are tubal scalars, while the G-tGKB process determines a lower bidiagonal matrix with scalar entries. Both the tGKB and G-tGKB processes furnish an orthonormal basis for a t-Krylov subspace (1.4) and work with lateral tensor slices.

As mentioned above, we also consider minimization problems analogous to (1.1), in which the data is represented by a general third order tensor $\mathcal{B} \in \mathbb{R}^{\ell\times p\times n}$ with $p > 1$. Thus, we would like to solve

$$\min_{\mathcal{X}\in\mathbb{R}^{m\times p\times n}} \{\|\mathcal{A} * \mathcal{X} - \mathcal{B}\|_F^2 + \mu^{-1}\|\mathcal{L} * \mathcal{X}\|_F^2\}. \tag{1.8}$$

Four solution methods for (1.8) are described. Three of them are based on applying the tGKT and G-tGKT methods to each lateral slice $\vec{\mathcal{B}}_j$ of $\mathcal{B}$, $j = 1, 2, \dots, p$, independently. The other method generalizes the T-global Golub-Kahan-Tikhonov regularization method recently described by El Guide et al. [11] to allow for $\mathcal{L} \neq \mathcal{I}$. The latter method works with the lateral slices of the data tensor $\mathcal{B}$ simultaneously and will be referred to as the generalized global tGKT (GG-tGKT) method.

The GG-tGKT and other solution methods are described and compared in Sections 3, 4, 5, and 6. Computed examples show the G-tGKT method to yield higher accuracy, but the GG-tGKT method to require less CPU time. The fact that the latter method requires less CPU time can be expected, since this method uses larger chunks of data simultaneously than the former methods. Both the G-tGKT and GG-tGKT methods belong to the GKT_BTF family of methods described by Beik et al. [1] and require additional product definitions to the t-product. They also involve flattening.

1.1 Related prior work and some applications

An advantage of the formulations (1.2) and (1.8) with the t-product is that they tend to avoid loss of information due to flattening of the tensor $\mathcal{A}$; see Kilmer et al. [21]. Third order tensors arise naturally in color image and video restoration problems, as well as in the solution of certain discretized partial differential equations [2]. For instance, in video restoration, the tensor dimensions may correspond to image height and width, and frame number. Many image restoration methods vectorize each image by stacking the matrix elements that represent the image. The matrix entries represent pixel values that encode gray scale values. Each frame represents an image and thus, a video can be represented as a long vector with each entry representing a pixel value in some frame. However, this representation destroys spatial correlations and structural complexities that the original video frames may posses. In particular, the video restoration methods considered by Bentbib et al. [3] vectorize consecutive video frames. The t-product approach avoids matricization and vectorization. As a result, it preserves multidimensional structures. Application areas where the t-product has proved useful include facial recognition [17], tomographic image reconstruction [31], video completion [32], and image classification [28].

Several matrix-based methods, that do not apply the t-product, for solving problems of the form (1.1) have recently been described in the literature. Beik et al. [1] introduced the Golub-Kahan-Tikhonov Based Tensor Format (GKT_BTF) method and applied it to determine the solution of large-scale ill-posed Sylvester and Stein equations. This method allows $\mathcal{A}$, $\vec{\mathcal{X}}$, and $\vec{\mathcal{B}}$ to be arbitrary compatible tensors. Related methods have been applied by El Guide et al. [10, 11] to the solution of tensor equations based on the Einstein and t-products.

Bentbib et al. [3, 4] described other matrix-based approaches to solve linear discrete ill-posed tensor equations that use the formalism of so-called global iterative methods. These methods, as well as the methods mentioned above involve matricization of the tensor equation (1.1), i.e., the equation is transformed to an equivalent equation that only involves matrices and vectors.

1.2 Organization

This paper is organized as follows. Section 2 introduces notation and preliminaries associated with the t-product, while Section 3 discusses the Tikhonov minimization problems (1.2) and (1.8). Sections 4 describes the tGKT method for the solution of the tensor equations (1.2) and (1.8). Specifically, Subsection 4.1 discusses the application of the tGKT method to the solution of (1.2), while Subsection 4.2 considers the application of this method to the solution of (1.8) by solving p problems of the form (1.2) with the lateral data slices $\vec{\mathcal{B}}_j$, $j = 1, 2, \ldots, p$, of $\mathcal{B}$ independently. We refer to the resulting method as the tGKT$_p$ method. A variant of this method that is based on using nested t-Krylov subspaces also is discussed.

Section 5 describes global-type tensor methods, which are referred to as the G-tGKT and GG-tGKT methods, for the solution of (1.2) and (1.8). The GG-tGKT method, discussed in Subsection 5.1, determines the solution of (1.8) by working with the lateral slices of $\mathcal{B}$ simultaneously, while the G-tGKT method is used to solve (1.2) in Subsection 5.2. The latter method also is applied p times to determine a solution of (1.8) by working with the lateral slices of $\mathcal{B}$ independently.

Section 6 presents numerical examples that illustrate the effectiveness of the proposed methods. A comparison between the classical Golub-Kahan-Tikhonov (GKT) regularization method, the tGKT method, and the G-tGKT method is presented. Applications to image restoration are considered. Moreover, the global GKT (G-GKT) method described by Bentbib et al. [4] is compared to the tGKT$_p$, G-tGKT$_p$, and GG-tGKT methods. Section 7 contains concluding remarks.

2 Notation and Preliminaries

This section reviews results by Kilmer et al. [21, 22] and uses notation defined there and by Kolda and Bader [26]. A tensor is a multidimensional array of scalars. In this paper, all tensors are real and of order three. The order of a tensor is the number of modes. Thus, we consider tensors of the form $\mathcal{A} = [a_{ijk}]_{i,j,k=1}^{\ell,m,n} \in \mathbb{R}^{\ell\times m\times n}$. Matrices and vectors are tensors of order two and one, respectively. We will use capital calligraphic script letters to denote third order tensors, capital letters to denote matrices, lower case letters to denote vectors, and bold face lower case letters for tubal scalars (tubes or tube fibers) defined below. A *slice* of a third order tensor $\mathcal{A}$ is a 2D section obtained by fixing any one of the indices. Using MATLAB notation, $\mathcal{A}(i,:,:)$, $\mathcal{A}(:,j,:)$, and $\mathcal{A}(:,:,k)$ denote the ith horizontal, jth lateral, and kth frontal slices of $\mathcal{A}$, respectively. The jth lateral slice also is denoted by $\vec{\mathcal{A}}_j$. It is a tensor and is referred to as a *tensor column*. Sometimes it is convenient to identify $\vec{\mathcal{A}}_j$ with a matrix. The kth frontal slice, $\mathcal{A}(:,:,k)$, is denoted by $\mathcal{A}^{(k)}$ and is a matrix. A *fiber* of a third order tensor $\mathcal{A}$ is a 1D section obtained by fixing any two of the indices. Thus, $\mathcal{A}(:,j,k)$, $\mathcal{A}(i,:,k)$, and $\mathcal{A}(i,j,:)$ denote mode-1, mode-2, and mode-3 fibers (tubes), respectively.

We will use the t-product introduced by Kilmer and Martin [22] for the multiplication of a pair of third order tensors $\mathcal{C}$ and $\vec{\mathcal{D}}$. The t-product proceeds by unfolding $\mathcal{C}$ into a block circulant matrix and $\vec{\mathcal{D}}$ into a vector, multiplying these together, and folding the result back into a third order tensor. The t-product is a natural extension of matrix multiplication for tensors of order three [22] and higher [27]. The t-product allows QR- and SVD-like factorizations.

Given a third order tensor $\mathcal{A} \in \mathbb{R}^{\ell\times m\times n}$ with $\ell \times m$ frontal slices $\mathcal{A}^{(i)}$, $i = 1, 2, \ldots, n$, the operator `unfold`$(\mathcal{A})$ returns an $\ell n \times m$ matrix with the frontal slices, whereas the `fold` operator folds back the unfolded tensor $\mathcal{A}$, i.e.,

$$\texttt{unfold}(\mathcal{A}) = \begin{bmatrix} \mathcal{A}^{(1)} \\ \mathcal{A}^{(2)} \\ \vdots \\ \mathcal{A}^{(n)} \end{bmatrix}, \qquad \texttt{fold}(\texttt{unfold}(\mathcal{A})) = \mathcal{A}.$$

The operator $\texttt{bcirc}(\mathcal{A})$ generates an $\ell n \times mn$ block-circulant matrix with $\texttt{unfold}(\mathcal{A})$ forming the first block column,

$$\texttt{bcirc}(\mathcal{A}) = \begin{bmatrix} \mathcal{A}^{(1)} & \mathcal{A}^{(n)} & \dots & \mathcal{A}^{(2)} \\ \mathcal{A}^{(2)} & \mathcal{A}^{(1)} & \dots & \mathcal{A}^{(3)} \\ \vdots & \vdots & \ddots & \vdots \\ \mathcal{A}^{(n)} & \mathcal{A}^{(n-1)} & \dots & \mathcal{A}^{(1)} \end{bmatrix}.$$

Definition 2.1. (t-product [22]) Let $\mathcal{A} \in \mathbb{R}^{\ell \times m \times n}$ and $\mathcal{B} \in \mathbb{R}^{m \times p \times n}$. Then the t-product $\mathcal{A} * \mathcal{B}$ is the tensor $\mathcal{C} \in \mathbb{R}^{\ell \times p \times n}$ defined by

$$\mathcal{C} = \texttt{fold}(\texttt{bcirc}(\mathcal{A}) \cdot \texttt{unfold}(\mathcal{B})). \tag{2.9}$$

Its (i,j)th tube is given by

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

When a third order tensor of size $\ell \times m \times n$ is viewed as an $\ell \times m$ matrix of tubes oriented along the third dimension, the t-product is analogous to matrix multiplication, except that multiplication between scalars is replaced by *circular convolution* between tubes. This allows us to define the tensor (2.9) as a t-linear operator on the set of laterally oriented matrices.

We may choose to evaluate the t-product $\mathcal{A} * \mathcal{B}$ according to the definition (2.9) if the tensors $\mathcal{A}$ and $\mathcal{B}$ are sparse. For general tensors, it is more efficient to use the fast Fourier transformation (FFT). Just as circulant matrices are diagonalizable by the discrete Fourier transform (DFT) matrix, block circulant matrices can be block diagonalized by this matrix combined with a Kronecker product. Suppose $\mathcal{A} \in \mathbb{R}^{\ell \times m \times n}$ and let F_n be an $n \times n$ unitary DFT matrix defined by

$$F_n = \frac{1}{\sqrt{n}} \begin{bmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & \omega & \omega^2 & \dots & \omega^{n-1} \\ 1 & \omega^2 & \omega^4 & \dots & \omega^{2(n-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega^{n-1} & \omega^{n-1} & \dots & \omega^{(n-1)(n-1)} \end{bmatrix}$$

with $\omega = e^{\frac{-2\pi i}{n}}$ and $i^2 = -1$. Then

$$\bar{A} := \texttt{blockdiag}(\widehat{A}^{(1)}, \widehat{A}^{(2)}, \dots, \widehat{A}^{(n)}) = (F_n \otimes I_\ell) \cdot \texttt{bcirc}(\mathcal{A}) \cdot (F_n^* \otimes I_m), \tag{2.10}$$

where $\otimes$ stands for Kronecker product, F_n^* denotes the conjugate transpose of F_n, and $\cdot$ is the standard matrix-matrix product. The matrix $\bar{A} \in \mathbb{R}^{\ell n \times mn}$ is block diagonal with the diagonal blocks $\widehat{A}^{(i)} \in \mathbb{R}^{\ell \times m}$, $i = 1, 2, \dots, n$. The matrices $\widehat{A}^{(i)}$ are the frontal slices of the tensor $\widehat{\mathcal{A}}$ obtained by applying the FFT along each tube of $\mathcal{A}$. Each matrix $\widehat{A}^{(i)}$ may be dense and have complex entries unless certain symmetry conditions hold; see [22] for further details. Throughout this paper, we often will denote objects that are obtained by taking the FFT along the third dimension with a widehat over the argument, i.e., $\widehat{\cdot}$. We remark that

$$\|\mathcal{A}\|_F = \frac{1}{\sqrt{n}} \|\bar{A}\|_F. \tag{2.11}$$

Let n be a power of 2. Then the right-hand side of (2.10) can be evaluated in $\mathcal{O}(\ell mn \log_2(n))$ arithmetic floating point operations (flops) by using the fast Fourier transform (FFT). Similar flop counts also hold when n is not a power of two. Using the DFT matrix, the t-product (2.9) can be evaluated as

$$\mathcal{A} * \mathcal{B} = \texttt{fold}\Big((F_n^* \otimes I_\ell)\big((F_n \otimes I_\ell) \cdot \texttt{bcirc}(\mathcal{A}) \cdot (F_n^* \otimes I_m)\big)(F_n \otimes I_m)\texttt{unfold}(\mathcal{B})\Big). \tag{2.12}$$

It is easily shown that by taking the FFT along the tubes of $\mathcal{B} \in \mathbb{R}^{m\times p\times n}$, we can compute $(F_n \otimes I_m)\texttt{unfold}(\mathcal{B})$ in $\mathcal{O}(pmn\log_2(n))$ flops; see Kilmer and Martin [22] for details.

The computations (2.12) can be easily implemented in MATLAB. Using MATLAB notation, let $\widehat{\mathcal{C}} := \texttt{fft}(\mathcal{C}, [\,], 3)$ be the tensor obtained by applying the FFT along the third dimension. Then the t-product (2.12) can be computed by taking the FFT along the tubes of $\mathcal{A}$ and $\mathcal{B}$ to get $\widehat{\mathcal{A}} = \texttt{fft}(\mathcal{A}, [\,], 3)$ and $\widehat{\mathcal{B}} = \texttt{fft}(\mathcal{B}, [\,], 3)$, followed by a matrix-matrix product of each pair of the frontal slices of $\widehat{\mathcal{A}}$ and $\widehat{\mathcal{B}}$,

$$\widehat{\mathcal{C}}(:,:,i) = \widehat{\mathcal{A}}(:,:,i) \cdot \widehat{\mathcal{B}}(:,:,i), \quad i = 1, 2, \ldots, n,$$

and then taking the inverse FFT along the third dimension to obtain $\mathcal{C} = \texttt{ifft}(\widehat{\mathcal{C}}, [\,], 3)$.

Express the tensor $\mathcal{A}$ in terms of its tensor columns, i.e.,

$$\mathcal{A} = [\vec{\mathcal{A}}_1, \vec{\mathcal{A}}_2, \ldots, \vec{\mathcal{A}}_m] \in \mathbb{R}^{\ell\times m\times n}, \qquad \vec{\mathcal{A}}_j \in \mathbb{R}^{\ell\times 1\times n}, \qquad j = 1, 2, \ldots, m.$$

Following Newman et al. [28], we define the range of the tensor $\mathcal{A}$ as the *t-linear span* of the lateral slices of $\mathcal{A}$,

$$\mathcal{R}(\mathcal{A}) = \{\vec{\mathcal{A}}_1 * \mathbf{x}_1 + \cdots + \vec{\mathcal{A}}_m * \mathbf{x}_m \mid \mathbf{x}_j \in \mathbb{R}^{1\times 1\times n}\}.$$

For more details on the range and null space of $\mathcal{A}$; see [21]. The action of a third order tensor on oriented matrices, using the t-product, defines a linear transformation. In particular, if $T(\vec{\mathcal{X}}) = \mathcal{A} * \vec{\mathcal{X}}$, then $T : \mathbb{R}^{m\times 1\times n} \to \mathbb{R}^{\ell\times 1\times n}$ is a t-linear operator with the property that

$$T(\vec{\mathcal{X}} * \mathbf{c} + \vec{\mathcal{Y}} * \mathbf{d}) = T(\vec{\mathcal{X}}) * \mathbf{c} + T(\vec{\mathcal{Y}}) * \mathbf{d}$$

for arbitrary tubal scalars $\mathbf{c}, \mathbf{d}$ and arbitrary $\vec{\mathcal{X}}, \vec{\mathcal{Y}} \in \mathbb{R}^{m\times 1\times n}$; see Braman [5] for a proof. If $\ell = m$, then T is invertible when $\mathcal{A}$ is invertible.

Let $\mathcal{A} \in \mathbb{R}^{\ell\times m\times n}$. Then the tensor transpose $\mathcal{A}^T \in \mathbb{R}^{m\times \ell\times n}$ is the tensor obtained by first transposing each one of the frontal slices of $\mathcal{A}$, and then reversing the order of the transposed frontal slices 2 through n; see [22]. The tensor transpose has similar properties as the matrix transpose. For instance, if $\mathcal{A}$ and $\mathcal{B}$ are two tensors such that $\mathcal{A} * \mathcal{B}$ and $\mathcal{B}^T * \mathcal{A}^T$ are defined, then $(\mathcal{A} * \mathcal{B})^T = \mathcal{B}^T * \mathcal{A}^T$.

The identity tensor $\mathcal{I} \in \mathbb{R}^{m\times m\times n}$ is a tensor, whose first frontal slice, $\mathcal{I}^{(1)}$, is the $m \times m$ identity matrix and all other frontal slices, $\mathcal{I}^{(i)}$, $i = 2, 3, \ldots, n$, are zero matrices; see [22].

The concept of orthogonality is well defined under the t-product formalism; see Kilmer and Martin [22]. A tensor $\mathcal{Q} \in \mathbb{R}^{m\times m\times n}$ is said to be orthogonal if $\mathcal{Q}^T * \mathcal{Q} = \mathcal{Q} * \mathcal{Q}^T = \mathcal{I}$. Analogously to the columns of an orthogonal matrix, the lateral slices of $\mathcal{Q}$ are orthonormal, i.e.,

$$\mathcal{Q}^T(:, i, :) * \mathcal{Q}(:, j, :) = \begin{cases} \mathbf{e}_1 & i = j, \\ \mathbf{0} & i \neq j, \end{cases}$$

where $\mathbf{e}_1 \in \mathbb{R}^{1\times 1\times n}$ is a tubal scalar with the $(1, 1, 1)$th entry equal to 1 and the remaining entries are zero. It is shown in [22] that if $\mathcal{Q}$ is an orthogonal tensor, then

$$\|\mathcal{Q} * \mathcal{A}\|_F = \|\mathcal{A}\|_F. \tag{2.13}$$

The notion of *partial orthogonality* is similar as in matrix theory. The tensor $\mathcal{Q} \in \mathbb{R}^{\ell\times m\times n}$ with $\ell > m$ is said to be partially orthogonal if $\mathcal{Q}^T * \mathcal{Q}$ is well defined and equal to the identity tensor $\mathcal{I} \in \mathbb{R}^{m\times m\times n}$; see [22].

A tensor $\mathcal{A} \in \mathbb{R}^{m\times m\times n}$ is said to have an inverse, denoted by $\mathcal{A}^{-1}$, provided that $\mathcal{A} * \mathcal{A}^{-1} = \mathcal{I}$ and $\mathcal{A}^{-1} * \mathcal{A} = \mathcal{I}$, whereas a tensor is said to be f-diagonal if each frontal slice of the tensor is a diagonal matrix; see [22].

The norm of a nonzero tensor column $\vec{\mathcal{X}} \in \mathbb{R}^{m\times 1\times n}$ is defined as

$$\|\vec{\mathcal{X}}\| := \frac{\|\vec{\mathcal{X}}^T * \vec{\mathcal{X}}\|_F}{\|\vec{\mathcal{X}}\|_F},$$

and $\|\vec{\mathcal{X}}\| = 0$ if $\vec{\mathcal{X}} = \vec{\mathcal{O}}$; see [21] for details.

The Frobenius norm of a tensor column $\vec{\mathcal{X}} \in \mathbb{R}^{m\times 1\times n}$ is given by

$$\|\vec{\mathcal{X}}\|_F = \sqrt{(\vec{\mathcal{X}}^T * \vec{\mathcal{X}})_{(:,:,1)}}; \tag{2.14}$$

see [21]. Thus, the square of the Frobenius norm of $\vec{\mathcal{X}}$ is the (1, 1, 1)th entry of the tubal scalar $\vec{\mathcal{X}}^T * \vec{\mathcal{X}}$. In other words, it is the first frontal face of the tube $\vec{\mathcal{X}}^T * \vec{\mathcal{X}} \in \mathbb{R}^{1\times 1\times n}$. Throughout this paper, the quantity $\|\cdot\|_2$ denotes the Euclidean vector norm. It is used in Algorithm 1 below. This algorithm takes a nonzero tensor $\vec{\mathcal{X}} \in \mathbb{R}^{m\times 1\times n}$ and returns a normalized tensor $\vec{\mathcal{V}} \in \mathbb{R}^{m\times 1\times n}$ and a tubal scalar $\mathbf{a} \in \mathbb{R}^{1\times 1\times n}$ such that

$$\vec{\mathcal{X}} = \vec{\mathcal{V}} * \mathbf{a} \quad \text{and} \quad \|\vec{\mathcal{V}}\| = 1.$$

Note that the tubal scalar $\mathbf{a}$ might not be invertible; see [21]. We mention that $\mathbf{a}$ is invertible if there is a tubal scalar $\mathbf{b}$ such that $\mathbf{a} * \mathbf{b} = \mathbf{b} * \mathbf{a} = \mathbf{e}_1$. The scalar $\mathbf{a}^{(j)}$ is the jth frontal slice (face) of the $1 \times 1 \times n$ tubal scalar $\mathbf{a}$, while $\vec{\mathcal{V}}^{(j)}$ is a vector of length m and the jth face of $\vec{\mathcal{V}} \in \mathbb{R}^{m\times 1\times n}$.

Algorithm 1: Normalization [20]

```
Input: $\vec{\mathcal{X}} \in \mathbb{R}^{m\times 1\times n} \neq \vec{\mathcal{O}}$
Output: $\vec{\mathcal{V}}$, $\mathbf{a}$ with $\vec{\mathcal{X}} = \vec{\mathcal{V}} * \mathbf{a}$ and $\|\vec{\mathcal{V}}\| = 1$
1  $\vec{\mathcal{V}} \leftarrow \texttt{fft}(\vec{\mathcal{X}}, [\,], 3)$
2  for $j = 1, 2, \dots, n$ do
3      $\mathbf{a}^{(j)} \leftarrow \|\vec{\mathcal{V}}^{(j)}\|_2$   ($\vec{\mathcal{V}}^{(j)}$ is a vector)
4      if $\mathbf{a}^{(j)} > \texttt{tol}$ then
5          $\vec{\mathcal{V}}^{(j)} \leftarrow \frac{1}{\mathbf{a}^{(j)}} \vec{\mathcal{V}}^{(j)}$
6      else
7          $\vec{\mathcal{V}}^{(j)} \leftarrow \texttt{randn}(m, 1)$;  $\mathbf{a}^{(j)} \leftarrow \|\vec{\mathcal{V}}^{(j)}\|_2$;  $\vec{\mathcal{V}}^{(j)} \leftarrow \frac{1}{\mathbf{a}^{(j)}} \vec{\mathcal{V}}^{(j)}$;  $\mathbf{a}^{(j)} \leftarrow 0$
8      end
9  end
10 $\vec{\mathcal{V}} \leftarrow \texttt{ifft}(\vec{\mathcal{V}}, [\,], 3)$;  $\mathbf{a} \leftarrow \texttt{ifft}(\mathbf{a}, [\,], 3)$
```

The t-product-based tensor QR (tQR) factorization is described by Kilmer et al. [21] and implemented by Algorithm 2. Let $\mathcal{A} \in \mathbb{R}^{\ell\times m\times n}$. Then its tQR factorization is given by

$$\mathcal{A} = \mathcal{Q} * \mathcal{R},$$

where the tensor $\mathcal{Q} \in \mathbb{R}^{\ell\times m\times n}$ is partially orthogonal and the tensor $\mathcal{R} \in \mathbb{R}^{m\times m\times n}$ is f-upper triangular (i.e., each face is upper triangular). Algorithm 2 computes a QR factorization of each diagonal block of a block diagonal matrix in the Fourier domain. Existence of the factorization is assured under suitable conditions; see [21].

Algorithm 2: tQR decomposition [21]

```
Input: $\mathcal{A} \in \mathbb{R}^{\ell\times m\times n}$, $\ell \geq m$
Output: $\mathcal{Q} \in \mathbb{R}^{\ell\times m\times n}$, $\mathcal{R} \in \mathbb{R}^{m\times m\times n}$ such that $\mathcal{A} = \mathcal{Q} * \mathcal{R}$
1  $\widehat{\mathcal{A}} \leftarrow \texttt{fft}(\mathcal{A}, [\,], 3)$
2  for $i = 1, 2, \dots, n$ do
3      Factor $\widehat{\mathcal{A}}(:, :, i) = QR$, where $Q$ is unitary
4      $\widehat{\mathcal{Q}}(:, :, i) \leftarrow Q$,   $\widehat{\mathcal{R}}(:, :, i) \leftarrow R$
5  end
6  $\mathcal{Q} \leftarrow \texttt{ifft}(\widehat{\mathcal{Q}}, [\,], 3)$,   $\mathcal{R} \leftarrow \texttt{ifft}(\widehat{\mathcal{R}}, [\,], 3)$
```

Let $\mathcal{A} \in \mathbb{R}^{\ell\times m\times n}$. Then the tensor singular value decomposition (tSVD), introduced by Kilmer and Martin [22], is given by

$$\mathcal{A} = \mathcal{U} * \mathcal{S} * \mathcal{V}^T, \tag{2.15}$$

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

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

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

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

The tubal rank of $\mathcal{A}$ is the number of nonzero singular tubes of $\mathcal{A}$; see Kilmer et al. [21].

We introduce additional notation used by El Guide et al. [11], which will be useful when discussing the G-tGKT and GG-tGKT methods in Section 5. Table 1 summarizes the main notations used in this paper. Let

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

where $\mathcal{C}_j \in \mathbb{R}^{m \times p \times n}$ and $\vec{\mathcal{C}}_j \in \mathbb{R}^{m \times 1 \times n}$, $j = 1, 2, \ldots, k$. Suppose that $y = [y_1, y_2, \ldots, y_k]^T \in \mathbb{R}^k$. Then El Guide et al. [11] defined the product $\circledast$ as

$$\mathbb{C}_k \circledast y = \sum_{j=1}^{k} y_j \mathcal{C}_j, \qquad \mathcal{C}_k \circledast y = \sum_{j=1}^{k} y_j \vec{\mathcal{C}}_j. \tag{2.16}$$

It can be shown that for orthogonal tensors $\mathbb{Q} \in \mathbb{R}^{m \times kp \times n}$ and $\mathcal{Q} \in \mathbb{R}^{m \times k \times n}$, one has

$$\|\mathbb{Q} \circledast y\|_F = \|y\|_2, \qquad \|\mathcal{Q} \circledast y\|_F = \|y\|_2; \tag{2.17}$$

see [11] for details.

Consider the tensors $\mathcal{C} = [c_{ijk}], \mathcal{D} = [d_{ijk}] \in \mathbb{R}^{m \times p \times n}$ with lateral slices $\vec{\mathcal{C}} = [c_{i1k}], \vec{\mathcal{D}} = [d_{i1k}] \in \mathbb{R}^{m \times 1 \times n}$. Introduce the inner products

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

Let

$$\begin{aligned} \mathbb{A} &:= [\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_m] \in \mathbb{R}^{\ell \times sm \times n}, \qquad \mathbb{B} := [\mathcal{B}_1, \mathcal{B}_2, \ldots, \mathcal{B}_p] \in \mathbb{R}^{\ell \times sp \times n}, \\ \mathcal{A} &:= [\vec{\mathcal{A}}_1, \vec{\mathcal{A}}_2, \ldots, \vec{\mathcal{A}}_m] \in \mathbb{R}^{\ell \times m \times n}, \qquad \mathcal{B} := [\vec{\mathcal{B}}_1, \vec{\mathcal{B}}_2, \ldots, \vec{\mathcal{B}}_p] \in \mathbb{R}^{\ell \times p \times n}, \end{aligned}$$

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 $\mathcal{B}_j \in \mathbb{R}^{\ell \times s \times n}$, $\vec{\mathcal{B}}_j \in \mathbb{R}^{\ell \times 1 \times n}$ $j = 1, 2, \ldots, p$. Following El Guide et al. [11], we define the T-diamond products $\mathbb{A}^T \lozenge \mathbb{B}$ and $\mathcal{A}^T \lozenge \mathcal{B}$. They yield $m \times p$ matrices given by

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

We conclude this section with the definition of some tensor operators that are convenient to apply in the sequel. The `squeeze` and `twist` operators, defined by Kilmer et al. [21], allow us to easily move between a lateral slice $\vec{\mathcal{X}}$ of a third order tensor and an associated matrix X. Specifically, the `squeeze` operator applied to $\vec{\mathcal{X}}$ is identical to the MATLAB squeeze function

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

whereas $\vec{\mathcal{X}} = \mathtt{twist}(X)$. Moreover, $\mathtt{twist}(\mathtt{squeeze}(\vec{\mathcal{X}})) = \vec{\mathcal{X}}$.

We also define the operators `multi_squeeze` and `multi_twist` that enable us to squeeze and twist a general third order tensor. Let $\mathcal{C} \in \mathbb{R}^{m \times p \times n}$, $p > 1$. Then the tensor $\mathcal{D} = \mathtt{multi\_squeeze}(\mathcal{C}) \in \mathbb{R}^{m \times n \times p}$ is obtained by squeezing each of the lateral slices of $\mathcal{C}$ and stacking them as frontal slices. The operation $\mathcal{C} = \mathtt{multi\_twist}(\mathcal{D})$ twists the frontal slices $\mathcal{D}^{(i)}$, $i = 1, 2, \ldots, p$, of $\mathcal{D}$ and stacks them as lateral slices of $\mathcal{C}$. We have $\mathtt{multi\_twist}(\mathtt{multi\_squeeze}(\mathcal{C})) = \mathcal{C}$.

Notation	Description	Used in section
$\mathcal{A}$	tensor	1-6
$\vec{\mathcal{A}}$	lateral slice	1-6
A	matrix	2-6
$A(:,j)$	jth column of A	5
$A(j,:)$	jth row of A	4
$\widehat{\mathcal{A}}$	FFT of $\mathcal{A}$ along the third mode	2-4
$\mathcal{A}^{(i)}$ or $A^{(i)}$	ith frontal slice of $\mathcal{A}$	2-4, 6
$\vec{\mathcal{A}}_j$	jth lateral slice of $\mathcal{A}$	1-6
$\mathcal{A}(j,:,:)$	jth horizontal slice of $\mathcal{A}$	4
$\mathbb{A}$	tensor	5.1
$\mathcal{A}_j$	jth column of $\mathbb{A}$	5.1
$\mathbf{a}$	tube	4
$(\mathbf{a})_{(:,:,1)}$	first face of $\mathbf{a}$	4
a	vector with jth entry a_j	4-5
$\mathcal{I}$	identity tensor	3
I	identity matrix	4-5
$\vec{e}_1$	first lateral slice of $\mathcal{I}$	4
e_1	first column of I	4-5
$*$	t-product	1-6
$\mathbb{A} \circledast a$	$\mathbb{A} \circledast a = \sum_j a_j \mathcal{A}_j$	5.1
$\mathcal{A} \circledast a$	$\mathcal{A} \circledast a = \sum_j a_j \vec{\mathcal{A}}_j$	5.2
$\mathbb{A} \circledast A$	$\mathbb{A} \circledast A = [\mathbb{A} \circledast A(:,1), \ldots, \mathbb{A} \circledast A(:,\text{end})]$	5.1
$\mathcal{A} \circledast A$	$\mathcal{A} \circledast A = [\mathcal{A} \circledast A(:,1), \ldots, \mathcal{A} \circledast A(:,\text{end})]$	5.2
$\langle \mathcal{A}, \mathcal{B} \rangle$	$\langle \mathcal{A}, \mathcal{B} \rangle = \sum_{ijk} a_{ijk} b_{ijk}$	5.1
$\langle \vec{\mathcal{A}}, \vec{\mathcal{B}} \rangle$	$\langle \vec{\mathcal{A}}, \vec{\mathcal{B}} \rangle = \sum_{ik} a_{i1k} b_{i1k}$	5.2
$\mathbb{A}^T \Diamond \mathbb{B}$	$[\mathbb{A}^T \Diamond \mathbb{B}]_{ij} = \langle \mathcal{A}_i,\ \mathcal{B}_j \rangle$	5.1
$\mathcal{A}^T \Diamond \mathcal{B}$	$[\mathcal{A}^T \Diamond \mathcal{B}]_{ij} = \langle \vec{\mathcal{A}}_i,\ \vec{\mathcal{B}}_j \rangle$	5.2
$\lVert \mathcal{A} \rVert_F$	$\lVert \mathcal{A} \rVert_F = \sqrt{\sum_{ijk} a_{ijk}^2}$	1-5
$\lVert \vec{\mathcal{A}} \rVert_F$	$\lVert \vec{\mathcal{A}} \rVert_F = \sqrt{(\vec{\mathcal{A}}^T * \vec{\mathcal{A}})_{(:,:,1)}}$	4

Table 1: Summary of notation.

3 Tensor Tikhonov Regularization

This section considers the Tikhonov minimization problem (1.2) and its solution. The discussion generalizes easily to the problem (1.8). We will i) show that the penalized least squares problem (1.2) has a unique solution $\vec{\mathcal{X}}_\mu$ for any $\mu > 0$, ii) describe how a suitable value of the regularization parameter μ and the associated solution $\vec{\mathcal{X}}_\mu$ can be computed by using the discrepancy principle, and iii) discuss the structure of the regularization operator $\mathcal{L}$ used in the computed examples of Section 6. We start with the minimization problem (1.2).

Theorem 3.1. Let $\mu > 0$ be the regularization parameter and assume that (1.3) holds. Then the minimization problem (1.2) has the unique solution

$$\vec{\mathcal{X}}_\mu = (\mathcal{A}^T * \mathcal{A} + \mu^{-1} \mathcal{L}^T * \mathcal{L})^{-1} * \mathcal{A}^T * \vec{\mathcal{B}}. \tag{3.18}$$

Proof: The functional

$$\mathcal{J}_\mu(\vec{\mathcal{X}}) := \lVert \mathcal{A} * \vec{\mathcal{X}} - \vec{\mathcal{B}} \rVert_F^2 + \mu^{-1} \lVert \mathcal{L} * \vec{\mathcal{X}} \rVert_F^2$$

can be written as

$$\mathcal{J}_\mu(\vec{\mathcal{X}}) = \left\lVert \begin{bmatrix} \mathcal{A} \\ \mu^{-1/2} \mathcal{L} \end{bmatrix} * \vec{\mathcal{X}} - \begin{bmatrix} \vec{\mathcal{B}} \\ \vec{\mathcal{O}} \end{bmatrix} \right\rVert_F^2,$$

where

$$\begin{bmatrix} \mathcal{A} \\ \mu^{-1/2} \mathcal{L} \end{bmatrix} \in \mathbb{R}^{(m+s) \times m \times n}, \quad \begin{bmatrix} \vec{\mathcal{B}} \\ \mathcal{O} \end{bmatrix} \in \mathbb{R}^{(m+s) \times 1 \times n}, \tag{3.19}$$

and $\vec{\mathcal{O}} \in \mathbb{R}^{s\times 1\times n}$ denotes a laterally oriented $s\times n$ zero matrix. It follows from (1.3) that the tensor on the left of (3.19) is of tubal rank m for any $\mu > 0$. Therefore the tensor $\mathcal{A}^T * \mathcal{A} + \mu^{-1}\mathcal{L}^T * \mathcal{L}$ is of tubal rank m. The normal equations associated with the minimization problem (1.2) are given by

$$(\mathcal{A}^T * \mathcal{A} + \mu^{-1}\mathcal{L}^T * \mathcal{L}) * \vec{\mathcal{X}} = \mathcal{A}^T * \vec{\mathcal{B}}. \tag{3.20}$$

Their unique solution can be expressed as (3.18). $\square$

Kilmer et al. [21] and Martin et al. [27] considered a similar formulation of (3.20) when $\mathcal{L} = \mathcal{I}$. Then the solution (3.18) simplifies to

$$\vec{\mathcal{X}}_\mu = (\mathcal{A}^T * \mathcal{A} + \mu^{-1}\mathcal{I})^{-1} * \mathcal{A}^T * \vec{\mathcal{B}}. \tag{3.21}$$

Using this expression for $\vec{\mathcal{X}}_\mu$, define the function

$$\phi(\mu) := \|\mathcal{A} * \vec{\mathcal{X}}_\mu - \vec{\mathcal{B}}\|_F^2. \tag{3.22}$$

Then equation (1.7) (for $\mathcal{L} = \mathcal{I}$) can be written as

$$\phi(\mu) = \eta^2\delta^2. \tag{3.23}$$

A zero-finder, such as bisection, Newton's method, or one of the methods discussed in [6, 30] can be used to solve (3.23) for $\mu_{\rm discr} = \mu > 0$. We assume here and below that $0 < \eta\delta < \|\vec{\mathcal{B}}\|_F$. Then (3.23) has a unique bounded solution $\mu_{\rm discr} > 0$, and $\vec{\mathcal{X}}_{\mu_{\rm discr}}$ satisfies the discrepancy principle (1.7) (when $\mathcal{L} = \mathcal{I}$).

With the notation of Section 2, we can write (3.23) in matrix-vector form,

$$\|\mathcal{A} * \vec{\mathcal{X}}_\mu - \vec{\mathcal{B}}\|_F^2 = \|\texttt{bcirc}(\mathcal{A})\texttt{unfold}(\vec{\mathcal{X}}_\mu) - \texttt{unfold}(\vec{\mathcal{B}})\|_2^2 = \|\texttt{bcirc}(\mathcal{A})x_\mu - b\|_2^2 = \eta^2\delta^2, \tag{3.24}$$

where $x_\mu = \texttt{unfold}(\vec{\mathcal{X}}_\mu)$ and $b = \texttt{unfold}(\vec{\mathcal{B}})$. Thus, the application of the discrepancy principle to the solution of (3.23) is equivalent to its application to the matrix-vector form (3.24). The use of the discrepancy principle for discrete ill-posed problems in matrix-vector form has been thoroughly discussed in the literature; see, e.g., [3, 7, 15, 18]. We use the form (1.7) because it is not necessary to form a $\texttt{bcirc}(\mathcal{A})$ explicitly; see [21].

The following results describe some properties of the function ϕ. We remark that while the solution (3.21) is meaningful for $\mu > 0$ only, we may define $\phi(\mu)$ for $\mu \geq 0$ by continuity. This is done in the propositions below.

Proposition 3.1. Assume that $\mathcal{A}^T * \vec{\mathcal{B}} \neq \vec{\mathcal{O}}$ and let $\phi(\mu)$ be given by (3.22) with $\vec{\mathcal{X}}_\mu$ defined by (3.21). Then

$$\phi(\mu) = \left(\vec{\mathcal{B}}^T * (\mu\mathcal{A} * \mathcal{A}^T + \mathcal{I})^{-2} * \vec{\mathcal{B}}\right)_{(:,:,1)}, \qquad \mu \geq 0, \tag{3.25}$$

and $\phi(0) = \|\vec{\mathcal{B}}\|_F^2$.

Proof: Let $\mu > 0$ and substitute (3.21) into (3.22). Using the identity

$$\mathcal{I} - \mathcal{A} * (\mathcal{A}^T * \mathcal{A} + \mu^{-1}\mathcal{I})^{-1} * \mathcal{A}^T = (\mu\mathcal{A} * \mathcal{A}^T + \mathcal{I})^{-1} \tag{3.26}$$

gives

$$\phi(\mu) = \|(\mu\mathcal{A} * \mathcal{A}^T + \mathcal{I})^{-1} * \vec{\mathcal{B}}\|_F^2. \tag{3.27}$$

By continuity, eq. (3.27) also holds for $\mu = 0$, i.e., $\phi(0) = \|\vec{\mathcal{B}}\|_F^2$. The result now follows by using (2.14). $\square$

Define the function

$$f(\mu) := (\mu\mathcal{A} * \mathcal{A}^T + \mathcal{I})^{-2}.$$

Transformation of $f(\mu)$ to the Fourier domain, and differentiating the resulting expression face-wise gives

$$\frac{d}{d\mu}\widehat{f}^{(i)}(\mu) = -2\big(\mu\widehat{\mathcal{A}}^{(i)}\big(\widehat{\mathcal{A}}^{(i)}\big)^T + I\big)^{-3}\widehat{\mathcal{A}}^{(i)}(\widehat{\mathcal{A}}^{(i)})^T, \quad i = 1, 2, \ldots, n. \tag{3.28}$$

Folding the resulting tensor function associated with (3.28) and transforming back to the spatial domain gives

$$f'(\mu) = -2\big((\mu\mathcal{A} * \mathcal{A}^T + \mathcal{I})^{-3} * \mathcal{A} * \mathcal{A}^T\big). \tag{3.29}$$

Hence by (3.29), the derivative of (3.25) can be written as

$$\phi'(\mu) = -2\big((\vec{\mathcal{B}}^T * (\mu\mathcal{A} * \mathcal{A}^T + \mathcal{I})^{-3} * \mathcal{A} * \mathcal{A}^T * \vec{\mathcal{B}}\big)_{(:,:,1)}. \tag{3.30}$$

Using the identity

$$(\mu\mathcal{A} * \mathcal{A}^T + \mathcal{I})^{-1} * \mathcal{A} * \mathcal{A}^T = \mathcal{A} * \mathcal{A}^T * (\mu\mathcal{A} * \mathcal{A}^T + \mathcal{I})^{-1}, \tag{3.31}$$

we can simplify (3.30) to

$$\phi'(\mu) = -2\big(\vec{\mathcal{Z}}_1^T * \vec{\mathcal{Z}}_2\big)_{(:,:,1)},$$

where

$$\vec{\mathcal{Z}}_1 := (\mu\mathcal{A} * \mathcal{A}^T + \mathcal{I})^{-1} * \vec{\mathcal{B}}, \quad \vec{\mathcal{Z}}_2 := (\mu\mathcal{A} * \mathcal{A}^T + \mathcal{I})^{-1} * \mathcal{A} * \mathcal{A}^T * \vec{\mathcal{Z}}_1. \tag{3.32}$$

We are now in a position to show that the function $\phi(\mu)$ is decreasing and convex.

Proposition 3.2. The function $\phi(\mu)$ defined by (3.22) for $\mu > 0$ satisfies

$$\phi'(\mu) < 0, \qquad \phi''(\mu) > 0.$$

Proof: The representation (3.30) and an identity analogous to (3.31) give

$$\phi'(\mu) = -2\big((\vec{\mathcal{B}}^T * \mathcal{A} * (\mu\mathcal{A} * \mathcal{A}^T + \mathcal{I})^{-3} * \mathcal{A}^T * \vec{\mathcal{B}}\big)_{(:,:,1)}. \tag{3.33}$$

Substituting the tSVD (2.15) into (3.33), and letting $\vec{\mathcal{W}} = \mathcal{U}^T * \mathcal{A}^T * \vec{\mathcal{B}}$ and $\mathcal{D} = \mathcal{S} * \mathcal{S}^T$ yield

$$\phi'(\mu) = -2\big(\vec{\mathcal{W}}^T * (\mu\mathcal{D} + \mathcal{I})^{-3} * \vec{\mathcal{W}}\big)_{(:,:,1)} < 0.$$

Generically, the inequality is strict. Thus, $\phi(\mu)$ is decreasing. Also,

$$\phi''(\mu) = 6\big(\vec{\mathcal{B}}^T * (\mu\mathcal{A} * \mathcal{A}^T + \mathcal{I})^{-4} * \mathcal{A} * \mathcal{A}^T * \vec{\mathcal{B}}\big)_{(:,:,1)},$$

which gives

$$\phi''(\mu) = 6\big(\vec{\mathcal{W}}^T * (\mu\mathcal{D} + \mathcal{I})^{-4} * \vec{\mathcal{W}}\big)_{(:,:,1)} > 0,$$

where, generically, the inequality is strict. The proposition follows. $\square$

We now describe a sufficiently stable way for our applications of interest to compute the solution $\vec{\mathcal{X}}_\mu$ of the least-squares problem (1.2) when $\mu > 0$. Transformation of (3.18) to the Fourier domain gives

$$\widehat{x}_\mu^{(i)} = \Big((\widehat{\mathcal{A}}^{(i)})^T\widehat{\mathcal{A}}^{(i)} + \mu^{-1}(\widehat{\mathcal{L}}^{(i)})^T\widehat{\mathcal{L}}^{(i)}\Big)^{-1}(\widehat{\mathcal{A}}^{(i)})^T\widehat{b}^{(i)}, \quad i = 1, 2, \ldots, n.$$

where the vectors $\widehat{x}_\mu^{(1)}, \widehat{x}_\mu^{(2)}, \ldots, \widehat{x}_\mu^{(n)}$ and $\widehat{b}^{(1)}, \widehat{b}^{(2)}, \ldots, \widehat{b}^{(n)}$ are the faces of the lateral slices $\vec{\mathcal{X}}$ and $\vec{\mathcal{B}}$ in the Fourier domain, respectively. The vectors $\widehat{x}_\mu^{(i)}$, $i = 1, 2\ldots, n$, exist and are uniquely determined, because (1.3) is equivalent to

$$\mathcal{N}(\widehat{\mathcal{A}}^{(i)}) \cap \mathcal{N}(\widehat{\mathcal{L}}^{(i)}) = \{0\}, \quad i = 1, 2, \ldots, n,$$

where $\mathcal{N}(M)$ denotes the null space of the matrix M, and 0 is the zero vector. The vectors $\widehat{x}_\mu^{(i)}$ are computed as the solutions of the least squares problems

$$\min_{\widehat{x}_\mu^{(i)} \in \mathbb{R}^m} \left\| \begin{bmatrix} \widehat{\mathcal{A}}^{(i)} \\ \mu^{-1/2}\widehat{\mathcal{L}}^{(i)} \end{bmatrix} \widehat{x}_\mu^{(i)} - \begin{bmatrix} \widehat{b}^{(i)} \\ 0 \end{bmatrix} \right\|_F, \quad i = 1, 2, \ldots, n. \tag{3.34}$$

This is advantageous compared to computing $\widehat{x}_\mu^{(i)}$ as the solution of

$$\Big((\widehat{\mathcal{A}}^{(i)})^T\widehat{\mathcal{A}}^{(i)} + \mu^{-1}(\widehat{\mathcal{L}}^{(i)})^T\widehat{\mathcal{L}}^{(i)}\Big)\,\widehat{x}_\mu^{(i)} = (\widehat{\mathcal{A}}^{(i)})^T\widehat{b}^{(i)},$$

because the condition number of the system matrix on the left-hand side above is the square of the condition number of the system matrix in (3.34). The n least squares problem (3.34) in the spatial domain are equivalent to the minimization problem

$$\min_{\vec{\mathcal{X}}\in\mathbb{R}^{m\times 1\times n}} \left\| \begin{bmatrix} \mathcal{A} \\ \mu^{-1/2}\mathcal{L} \end{bmatrix} * \vec{\mathcal{X}} - \begin{bmatrix} \vec{\mathcal{B}} \\ \vec{\mathcal{O}} \end{bmatrix} \right\|_F . \tag{3.35}$$

Algorithm 3 implements the solution of (3.35) by solving (3.34) at step 4 of the algorithm.

Algorithm 3: Solution of a generic tensor least squares problem

```
   Input: $\mathcal{C}\in\mathbb{R}^{\ell\times m\times n}$, where its Fourier transform has nonsingular frontal slices;
          $\vec{\mathcal{D}}\in\mathbb{R}^{\ell\times 1\times n}$, $\vec{\mathcal{D}}\neq\vec{\mathcal{O}}$
   Output: The solution $\vec{\mathcal{Y}}\in\mathbb{R}^{m\times 1\times n}$ of $\min_{\vec{\mathcal{Y}}\in\mathbb{R}^{m\times 1\times n}}\|\mathcal{C}*\vec{\mathcal{Y}}-\vec{\mathcal{D}}\|_F$
1  $\mathcal{C}\leftarrow$ fft($\mathcal{C}$, [ ], 3)
2  $\vec{\mathcal{D}}\leftarrow$ fft($\vec{\mathcal{D}}$, [ ], 3)
3  for $i=1$ to $n$ do
4      $\vec{\mathcal{Y}}(:,:,i)=\mathcal{C}(:,:,i)\backslash\vec{\mathcal{D}}(:,:,i)$, where $\backslash$ denotes MATLAB's backslash operator
5  end
6  $\vec{\mathcal{Y}}\leftarrow$ ifft($\vec{\mathcal{Y}}$, [ ], 3)
```

We conclude this section with the definition of the regularization operators $\mathcal{L}$ that we will use in Section 6. We will use two regularization operators $\mathcal{L}_1\in\mathbb{R}^{(m-2)\times m\times n}$ and $\mathcal{L}_2\in\mathbb{R}^{(m-1)\times m\times n}$. The tensor $\mathcal{L}_1$ has a tridiagonal matrix as its first frontal slice,

$$\mathcal{L}_1^{(1)} = \frac{1}{4}\begin{bmatrix} -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \end{bmatrix} \in\mathbb{R}^{(m-2)\times m}, \tag{3.36}$$

and the remaining frontal slices $\mathcal{L}_1^{(i)}\in\mathbb{R}^{(m-2)\times m}$, $i=2,3,\dots,n$, are zero matrices. The first frontal slice of the regularization operator $\mathcal{L}_2\in\mathbb{R}^{(m-1)\times m\times n}$ is the bidiagonal matrix

$$\mathcal{L}_2^{(1)} = \frac{1}{2}\begin{bmatrix} 1 & -1 & & & \\ & 1 & -1 & & \\ & & \ddots & \ddots & \\ & & & 1 & -1 \end{bmatrix} \in\mathbb{R}^{(m-1)\times m}, \tag{3.37}$$

and the remaining frontal slices $\mathcal{L}_2^{(i)}\in\mathbb{R}^{(m-1)\times m}$, $i=2,3,\dots,n$, are zero matrices. Other regularization operators of interest can be defined similarly.

When solving linear discrete ill-posed matrix equations, e.g. (3.34), the penalty term in Tikhonov regularization is often chosen to be a matrix that is a discretized derivative operator; see, e.g., [8, 9, 15, 18]. This kind of matrix penalty term extends naturally to the tensor setting, because of how the computation of the t-product is carried out, i.e., by using MATLAB's `fft` operator along the third dimension to transform to the Fourier domain. Thus, we define the tensor regularization operator $\mathcal{L}$ so that each frontal slice of $\mathcal{L}=\mathcal{L}_1$ or $\mathcal{L}=\mathcal{L}_2$ in the Fourier domain corresponds to the matrix regularization operators $\mathcal{L}_1^{(1)}$ and $\mathcal{L}_2^{(2)}$, respectively. These matrices are finite difference operators in one space dimension. To solve the minimization problem (1.2), we compute in the Fourier domain the solution of n least squares problem of the form (3.34), where the first or second difference operators $\mathcal{L}_2^{(1)}$ or $\mathcal{L}_1^{(1)}$ are used.

4 The t-product Golub-Kahan-Tikhonov (tGKT) Method

This section discusses the computation of an approximate solution of the tensor Tikhonov regularization problem (1.2) with the aid of partial t-product Golub-Kahan bidiagonalization (tGKB), which has been introduced by Kilmer et al. [21]. We describe how partial tGKB can be used with the discrepancy principle and a general regularization tensor in Tikhonov regularization.

4.1 The tGKT method for the approximate solution of (1.2)

This subsection develops the tGKT method for the approximate solution of least-squares problems of the form (1.2). The discussion of this subsection will be generalized to least-squares problems of the form (1.8) in Subsection 4.2. Methods described in the present subsection will be used to illustrate the potential superiority of tensorizing as opposed to vectorizing or matricizing ill-posed problem for tensors.

The partial tGKB method, introduced by Kilmer et al. [21], can be applied to reduce a large-scale problem (1.1) to a problem of small size. This method is described by Algorithm 4, which reduces the tensor $\mathcal{A}$ to a small bidiagonal tensor. We will assume that the number of steps k of the tGKB process is small enough to avoid break down, i.e., k is chosen small enough so that the tubal scalars $\mathbf{c}_i$ and $\mathbf{z}_{i+1}$ for $i = 1, 2, \ldots, k$, determined by Algorithm 4, are invertible. This means that the transformed tubal scalars $\widehat{\mathbf{c}}_i$ of $\mathbf{c}_i$ and $\widehat{\mathbf{z}}_{i+1}$ of $\mathbf{z}_{i+1}$ do not have zero Fourier coefficients.

Algorithm 4: Partial tensor Golub-Kahan bidiagonalization

```
  Input: $\mathcal{A} \in \mathbb{R}^{\ell\times m\times n}$, $\vec{\mathcal{B}} \in \mathbb{R}^{\ell\times 1\times n}$ such that $\mathcal{A}^T * \vec{\mathcal{B}} \neq \vec{\mathcal{O}}$
1 $\vec{\mathcal{W}}_0 \leftarrow \vec{\mathcal{O}}$
2 $[\vec{\mathcal{Q}}_1, \mathbf{z}_1] \leftarrow$ Normalize($\vec{\mathcal{B}}$) with $\mathbf{z}_1$ invertible
3 for $i = 1, 2, \ldots, k$ do
4     $\begin{cases} \vec{\mathcal{W}}_i \leftarrow \mathcal{A}^T * \vec{\mathcal{Q}}_i - \vec{\mathcal{W}}_{i-1} * \mathbf{z}_i \text{ (no reorthogonalization)} \\ \vec{\mathcal{W}}_i \leftarrow \mathcal{A}^T * \vec{\mathcal{Q}}_i - \vec{\mathcal{W}}_{i-1} * \mathbf{z}_i,\ \vec{\mathcal{W}}_i \leftarrow \vec{\mathcal{W}}_i - \sum_{j=1}^{i-1} \vec{\mathcal{W}}_j * (\vec{\mathcal{W}}_j^T * \vec{\mathcal{W}}_i) \text{ (with reorthogonalization)} \end{cases}$
5     $[\vec{\mathcal{W}}_i, \mathbf{c}_i] \leftarrow$ Normalize($\vec{\mathcal{W}}_i$)
6     $\begin{cases} \vec{\mathcal{Q}}_{i+1} \leftarrow \mathcal{A} * \vec{\mathcal{W}}_i - \vec{\mathcal{Q}}_i * \mathbf{c}_i \text{ (no reorthogonalization)} \\ \vec{\mathcal{Q}}_{i+1} \leftarrow \mathcal{A} * \vec{\mathcal{W}}_i - \vec{\mathcal{Q}}_i * \mathbf{c}_i, \vec{\mathcal{Q}}_{i+1} \leftarrow \vec{\mathcal{Q}}_{i+1} - \sum_{j=1}^{i} \vec{\mathcal{Q}}_j * (\vec{\mathcal{Q}}_j^T * \vec{\mathcal{Q}}_{i+1}) \text{ (with reorthogonalization)} \end{cases}$
7     $[\vec{\mathcal{Q}}_{i+1}, \mathbf{z}_{i+1}] \leftarrow$ Normalize($\vec{\mathcal{Q}}_{i+1}$)
8 end
```

Algorithm 4 produces the partial tGKB decompositions

$$\mathcal{A} * \mathcal{W}_k = \mathcal{Q}_{k+1} * \bar{\mathcal{P}}_k, \qquad \mathcal{A}^T * \mathcal{Q}_k = \mathcal{W}_k * \mathcal{P}_k^T, \tag{4.38}$$

where

$$\bar{\mathcal{P}}_k = \begin{bmatrix} \mathbf{c}_1 & & & & \\ \mathbf{z}_2 & \mathbf{c}_2 & & & \\ & \mathbf{z}_3 & \mathbf{c}_3 & & \\ & & \ddots & \ddots & \\ & & & \mathbf{z}_k & \mathbf{c}_k \\ & & & & \mathbf{z}_{k+1} \end{bmatrix} \in \mathbb{R}^{(k+1)\times k\times n}$$

is a lower bidiagonal tensor and $\mathcal{P}_k$ represents the leading $k \times k \times n$ subtensor of $\bar{\mathcal{P}}_k$. The tensor columns $\vec{\mathcal{Q}}_i \in \mathbb{R}^{\ell\times 1\times n}$, $i = 1, 2, \ldots, k{+}1$, and $\vec{\mathcal{W}}_i \in \mathbb{R}^{m\times 1\times n}$, $i = 1, 2, \ldots, k$, generated by Algorithm 4 are orthonormal tensor bases for the t-Krylov subspaces $\mathcal{K}_{k+1}(\mathcal{A}*\mathcal{A}^T, \vec{\mathcal{B}})$ and $\mathcal{K}_k(\mathcal{A}^T*\mathcal{A}, \mathcal{A}^T*\vec{\mathcal{B}})$, respectively. They are the lateral slices of the tensors $\mathcal{Q}_{k+1} \in \mathbb{R}^{\ell\times(k+1)\times n}$ and $\mathcal{W}_k \in \mathbb{R}^{m\times k\times n}$, respectively. We assume k to be chosen small enough so that the decompositions (4.38) with the stated properties exist. These decompositions will be applied to determine an approximate solution of the Tikhonov minimization problem (1.2). Reorthogonalization will be applied in Algorithm 7 below.

Let $\vec{\mathcal{X}} = \mathcal{W}_k * \vec{\mathcal{Y}}$ and substitute the decomposition on the left-hand side of (4.38) into (1.2). This yields

$$\min_{\vec{\mathcal{Y}}\in\mathbb{R}^{k\times 1\times n}} \{\|\bar{\mathcal{P}}_k * \vec{\mathcal{Y}} - \mathcal{Q}_{k+1}^T * \vec{\mathcal{B}}\|_F^2 + \mu^{-1}\|\mathcal{L} * \mathcal{W}_k * \vec{\mathcal{Y}}\|_F^2\}. \tag{4.39}$$

Using the fact that $\vec{\mathcal{B}} = \vec{\mathcal{Q}}_1 * \mathbf{z}_1$ (cf. Algorithm 4), we obtain

$$\mathcal{Q}_{k+1}^T * \vec{\mathcal{B}} = \vec{e}_1 * \mathbf{z}_1 \in \mathbb{R}^{(k+1)\times 1\times n}, \tag{4.40}$$

where $\vec{e}_1 \in \mathbb{R}^{m\times 1\times n}$ is such that the $(1,1,1)$th entry of $\vec{e}_1$ equals 1 and the remaining entries vanish; the entry 1 appear only in the first frontal slice of $\vec{e}_1$. Substituting (4.40) into (4.39) gives

$$\min_{\vec{\mathcal{Y}}\in\mathbb{R}^{k\times 1\times n}} \{\|\bar{\mathcal{P}}_k * \vec{\mathcal{Y}} - \vec{e}_1 * \mathbf{z}_1\|_F^2 + \mu^{-1}\|\mathcal{L} * \mathcal{W}_k * \vec{\mathcal{Y}}\|_F^2\}. \tag{4.41}$$

Our approach of handling these regularization operators, which is analogous to the approach described in [18] for matrix problems, also can be applied to many other regularization operators. We use Algorithm 2 to compute the tQR decomposition

$$\mathcal{L} * \mathcal{W}_k = \mathcal{Q}_{\mathcal{L},k} * \mathcal{R}_{\mathcal{L},k}, \tag{4.42}$$

where $\mathcal{Q}_{\mathcal{L},k} \in \mathbb{R}^{s\times k\times n}$ has k orthonormal tensor columns and $\mathcal{R}_{\mathcal{L},k} \in \mathbb{R}^{k\times k\times n}$ is f-upper triangular. The factorization (4.42) can be evaluated recursively by updating the tQR decomposition of $\mathcal{L} * \mathcal{W}_{k-1}$. In view of (2.13), the minimization problem (4.41) simplifies to

$$\min_{\vec{\mathcal{Y}}\in\mathbb{R}^{k\times 1\times n}} \{\|\bar{\mathcal{P}}_k * \vec{\mathcal{Y}} - \vec{e}_1 * \mathbf{z}_1\|_F^2 + \mu^{-1}\|\mathcal{R}_{\mathcal{L},k} * \vec{\mathcal{Y}}\|_F^2\}. \tag{4.43}$$

For the regularization operators (3.36) and (3.37), as well as for many other regularization operators $\mathcal{L}$, the tensor $\mathcal{R}_{\mathcal{L},k}$ is invertible and not very ill-conditioned. In this situation, we may form

$$\vec{\mathcal{Z}} = \mathcal{R}_{\mathcal{L},k} * \vec{\mathcal{Y}}, \qquad \widetilde{\mathcal{P}}_k = \bar{\mathcal{P}}_k * \mathcal{R}_{\mathcal{L},k}^{-1}. \tag{4.44}$$

Substituting the above expressions into (4.43) yields

$$\min_{\vec{\mathcal{Z}}\in\mathbb{R}^{k\times 1\times n}} \left\{\|\widetilde{\mathcal{P}}_k * \vec{\mathcal{Z}} - \vec{e}_1 * \mathbf{z}_1\|_F^2 + \mu^{-1}\|\vec{\mathcal{Z}}\|_F^2\right\}.$$

This minimization problem can be solved fairly stably by computing the solution of

$$\min_{\vec{\mathcal{Z}}\in\mathbb{R}^{k\times 1\times n}} \left\| \begin{bmatrix} \widetilde{\mathcal{P}}_k \\ \mu^{-1/2}\mathcal{I} \end{bmatrix} * \vec{\mathcal{Z}} - \begin{bmatrix} \vec{e}_1 * \mathbf{z}_1 \\ \vec{\mathcal{O}} \end{bmatrix} \right\|_F, \tag{4.45}$$

using Algorithm 3 presented in Section 3. The solution can be expressed as

$$\vec{\mathcal{Z}}_{\mu,k} = (\widetilde{\mathcal{P}}_k^T * \widetilde{\mathcal{P}}_k + \mu^{-1}\mathcal{I})^{-1} * \widetilde{\mathcal{P}}_k^T * \vec{e}_1 * \mathbf{z}_1, \tag{4.46}$$

and the associated approximate solution of (1.2) is given by

$$\vec{\mathcal{X}}_{\mu,k} = \mathcal{W}_k * \mathcal{R}_{\mathcal{L},k}^{-1} * (\widetilde{\mathcal{P}}_k^T * \widetilde{\mathcal{P}}_k + \mu^{-1}\mathcal{I})^{-1} * \widetilde{\mathcal{P}}_k^T * \vec{e}_1 * \mathbf{z}_1.$$

We use the discrepancy principle (1.7) to determine the regularization parameter and the required number of steps of the tGKB algorithm as follows. Define the function

$$\phi_k(\mu) := \|\widetilde{\mathcal{P}}_k * \vec{\mathcal{Z}}_{\mu,k} - \vec{e}_1 * \mathbf{z}_1\|_F^2, \tag{4.47}$$

which is analogous to (3.22). Substituting (4.46) into (4.47), and using the identity (3.26) with $\mathcal{A}$ replaced by $\widetilde{\mathcal{P}}_k$, as well as (2.14), we obtain

$$\phi_k(\mu) = \left((\vec{e}_1 * \mathbf{z}_1)^T * (\mu\widetilde{\mathcal{P}}_k * \widetilde{\mathcal{P}}_k^T + \mathcal{I})^{-2} * \vec{e}_1 * \mathbf{z}_1\right)_{(:,:,1)}. \tag{4.48}$$

The following proposition shows that we can apply the discrepancy principle (1.7) to the reduced problem to determine $\mu > 0$, i.e., we require μ to be such that

$$\|\widetilde{\mathcal{P}}_k * \vec{\mathcal{Z}}_{\mu,k} - \vec{e}_1 * \mathbf{z}_1\|_F = \eta\delta.$$

Proposition 4.1. Let $\phi_k(\mu)$ be defined by (4.47), and assume that $\mu = \mu_k$ solves $\phi_k(\mu) = \eta^2\delta^2$ and that $\vec{\mathcal{Z}}_{\mu,k}$ solves

$$(\widetilde{\mathcal{P}}_k^T * \widetilde{\mathcal{P}}_k + \mu^{-1}\mathcal{I}) * \vec{\mathcal{Z}} = \widetilde{\mathcal{P}}_k^T * \vec{e}_1 * \mathbf{z}_1.$$

Let $\vec{\mathcal{Y}}_{\mu,k}$ and $\vec{\mathcal{Z}}_{\mu,k}$ be related by (4.44). Then the associated approximate solution $\vec{\mathcal{X}}_{\mu,k} = \mathcal{W}_k * \vec{\mathcal{Y}}_{\mu,k}$ of (1.1) satisfies

$$\|\mathcal{A} * \vec{\mathcal{X}}_{\mu,k} - \vec{\mathcal{B}}\|_F^2 = \left((\vec{e}_1 * \mathbf{z}_1)^T * (\mu\widetilde{\mathcal{P}}_k * \widetilde{\mathcal{P}}_k^T + \mathcal{I})^{-2} * \vec{e}_1 * \mathbf{z}_1\right)_{(:,:,1)}.$$

Proof: Substituting $\vec{\mathcal{X}}_{\mu,k} = \mathcal{W}_k * \vec{\mathcal{Y}}_{\mu,k}$ into (1.7) and using the left-hand side decomposition of (4.38), as well as (4.40), shows that

$$\|\mathcal{A} * \vec{\mathcal{X}}_{\mu,k} - \mathcal{B}\|_F^2 = \|\mathcal{Q}_{k+1} * \bar{\mathcal{P}}_k * \vec{\mathcal{Y}}_{\mu,k} - \vec{\mathcal{B}}\|_F^2 = \|\bar{\mathcal{P}}_k * \vec{\mathcal{Y}}_{\mu,k} - \vec{e}_1 * \mathbf{z}_1\|_F^2 = \|\widetilde{\mathcal{P}}_k * \vec{\mathcal{Z}}_{\mu,k} - \vec{e}_1 * \mathbf{z}_1\|_F^2. \quad \square$$

We show below that the function $\phi_k(\mu)$ is decreasing and convex with $\phi_k(0) = \|\vec{e}_1 * \mathbf{z}_1\|_F^2$. These results are analogous to those for the function (3.22). We also will discuss the dependence of $\phi_k(\mu)$ on k for fixed $\mu \geq 0$.

Proposition 4.2. The function $\phi_k(\mu)$ defined by (4.48) for $\mu > 0$ satisfies, for fixed k,

$$\phi_k'(\mu) < 0, \qquad \phi_k''(\mu) > 0.$$

Proof: The result can be shown similarly as Proposition 3.2. $\square$

Since $\phi_k(\mu)$ is decreasing and convex, it is easy to implement Newton's method for the solution of

$$\phi_k(\mu) - \eta^2\delta^2 = 0. \tag{4.49}$$

Newton's method converges monotonically and quadratically to the solution μ_k of (4.49) for any initial approximate solution $0 < \mu_0 < \mu_k$. In particular, we may use $\mu_0 = 0$ when ϕ_k and ϕ_k' are suitably defined at $\mu = 0$. Newton's method does not have to be safeguarded. This is implemented by Algorithm 5; the details follow similarly as the discussion (3.30)-(3.32). We note that when the regularization parameter $\mu > 0$ in (1.2) is replaced by $1/\mu$, the analogues of the functions ϕ and ϕ_k are not guaranteed to be convex. Then Newton's method has to be safeguarded.

Algorithm 5: Solution of (4.49) by Newton's method

```
   Input: $\widetilde{\mathcal{P}}_k$, $\vec{e}_1$, $\mathbf{z}_1$, $\delta$, $\eta$, $\mu_0 = 0$, $\texttt{iter}_{\texttt{max}} = 30$, $\texttt{eps} = 1$, $i = 0$
   Output: $\mu_{i+1} > 0$
1  while eps $> 10^{-6}$ and $i < \texttt{iter}_{\texttt{max}}$ do
2      Solve
           $\min_{\vec{\mathcal{Z}}_1 \in \mathbb{R}^{(k+1)\times 1 \times n}} \left\| \begin{bmatrix} \mu_i^{1/2}\widetilde{\mathcal{P}}_k^T \\ \mathcal{I} \end{bmatrix} * \vec{\mathcal{Z}}_1 - \begin{bmatrix} \vec{\mathcal{O}} \\ \vec{e}_1 * \mathbf{z}_1 \end{bmatrix} \right\|_F$
        for $\vec{\mathcal{Z}}_1$ by using Algorithm 3
3      Compute $\phi_k(\mu_i) \leftarrow (\vec{\mathcal{Z}}_1^T * \vec{\mathcal{Z}}_1)_{(:,:,1)}$
4      Solve
           $\min_{\vec{\mathcal{Z}}_2 \in \mathbb{R}^{(k+1)\times 1 \times n}} \left\| \begin{bmatrix} \mu_i^{1/2}\widetilde{\mathcal{P}}_k^T \\ \mathcal{I} \end{bmatrix} * \vec{\mathcal{Z}}_2 - \begin{bmatrix} \vec{\mathcal{O}} \\ \widetilde{\mathcal{P}}_k * \widetilde{\mathcal{P}}_k^T * \vec{\mathcal{Z}}_1 \end{bmatrix} \right\|_F$
        for $\vec{\mathcal{Z}}_2$ by using Algorithm 3
5      Compute $\phi_k'(\mu_i) \leftarrow -2(\vec{\mathcal{Z}}_1^T * \vec{\mathcal{Z}}_2)_{(:,:,1)}$
6
           $\mu_{i+1} \leftarrow \mu_i - \frac{\phi_k(\mu_i) - \delta^2\eta^2}{\phi_k'(\mu_i)}$,   eps $\leftarrow |\mu_{i+1} - \mu_i|$,   $\mu_i \leftarrow \mu_{i+1}$,   $i \leftarrow i + 1$
7  end
```

We refer to solution method for (4.39) described above as the tGKT method. It is implemented by Algorithm 6 with $p = 1$ described below.

The question whether equation (4.49) can be satisfied for a finite $\mu > 0$ deserves some attention. To shed light on this, we first consider the matrix analogue of (4.48), which is expressed as

$$\psi_k(\mu) = \beta_1^2 e_1^T (\mu \widetilde{P}_k \widetilde{P}_k^T + I)^{-2} e_1, \tag{4.50}$$

where $e_1 = [1, 0, \ldots, 0]^T$, $\beta_1 > 0$, and $\widetilde{P}_k \in \mathbb{R}^{(k+1)\times k}$ is a lower bidiagonal matrix with positive nontrivial entries. This matrix will be encountered again in Subsection 5.1.

Proposition 4.3. Let $\psi_k(\mu)$ be given by (4.50). Then

$$\lim_{\mu\to\infty} \psi_k(\mu) = \gamma\beta_1^2,$$

where $\gamma > 0$ is the square of the $(1,1)$ entry of the $(k+1)$st left singular vector of $\widetilde{P}_k$.

Proof: Introduce the singular value decomposition

$$\widetilde{P}_k = USV^T,$$

where the matrices $U \in \mathbb{R}^{(k+1)\times(k+1)}$ and $V \in \mathbb{R}^{k\times k}$ are orthogonal, and $S = \text{diag}[\sigma_1, \sigma_2, \ldots, \sigma_k] \in \mathbb{R}^{(k+1)\times k}$ with $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_k \geq 0$. Substitution into (4.50) gives

$$\psi_k(\mu) = \beta_1^2 e_1^T U(\mu SS^T + I)^{-2} U^T e_1,$$

where $SS^T = \text{diag}[\sigma_1^2, \sigma_2^2, \ldots, \sigma_k^2, 0]$. Thus,

$$\psi_k(\mu) = \beta_1^2 e_1^T U \text{diag}\left[\frac{1}{1+\mu\sigma_1^2}, \frac{1}{1+\mu\sigma_2^2}, \ldots, \frac{1}{1+\mu\sigma_k^2}, 1\right] U^T e_1$$

and

$$\lim_{\mu\to\infty} \psi_k(\mu) = \beta_1^2 U(1,:)\text{diag}[0,0,\ldots,0,1]U(1,:)^T.$$

This shows the proposition. $\square$

We return to the tensor problem. The following result is a tensor analogue of Proposition 4.3.

Proposition 4.4. Let $\phi_k(\mu)$ be given by (4.48). Then

$$\lim_{\mu\to\infty} \phi_k(\mu) = \left(\mathbf{z}_1^T * \mathcal{U}(1,:,:) * \mathcal{D} * \mathcal{U}(1,:,:)^T * \mathbf{z}_1\right)_{(:,:,1)}, \tag{4.51}$$

where $\mathcal{D} \in \mathbb{R}^{(k+1)\times(k+1)\times n}$ is a tensor, whose first frontal slice $\mathcal{D}^{(1)}$ has entry 1 in the $(k+1, k+1)$st position, and the remaining frontal slices $\mathcal{D}^{(i)}$, $i = 2, 3, \ldots, n$, are zero matrices. The tensor $\mathcal{U} \in \mathbb{R}^{(k+1)\times(k+1)\times n}$ is the left singular tensor of $\widetilde{\mathcal{P}}_k$.

Proof: Let

$$\mathcal{J}_k(\mu) = (\vec{e}_1 * \mathbf{z}_1)^T * (\mu\widetilde{\mathcal{P}}_k * \widetilde{\mathcal{P}}_k^T + \mathcal{I})^{-2} * \vec{e}_1 * \mathbf{z}_1 \tag{4.52}$$

and introduce the tSVD

$$\widetilde{\mathcal{P}}_k = \mathcal{U} * \mathcal{S} * \mathcal{V}^T,$$

where $\mathcal{U} \in \mathbb{R}^{(k+1)\times(k+1)\times n}$ and $\mathcal{V} \in \mathbb{R}^{k\times k\times n}$ are orthogonal tensors and

$$\mathcal{S} = \text{diag}[\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_k] \in \mathbb{R}^{(k+1)\times k\times n},$$

where $\mathbf{s}_j \in \mathbb{R}^{1\times 1\times n}$, $1 \leq j \leq k$, are singular tubes. Substituting this decomposition into (4.52) gives

$$\mathcal{J}_k(\mu) = (\vec{e}_1 * \mathbf{z}_1)^T * \mathcal{U} * (\mu\mathcal{S} * \mathcal{S}^T + \mathcal{I})^{-2} * \mathcal{U}^T * \vec{e}_1 * \mathbf{z}_1.$$

Transforming the above expression to the Fourier domain, we obtain

$$\widehat{J}_k^{(i)}(\mu) = (\widehat{\mathbf{z}}_1^{(i)})^2 e_1^T \widehat{U}^{(i)} \left(\mu\widehat{S}^{(i)}(\widehat{S}^{(i)})^T + I\right)^{-2} (\widehat{U}^{(i)})^T e_1, \quad i = 1, 2, \ldots, n,$$

where

$$\widehat{S}^{(i)}(\widehat{S}^{(i)})^T = \text{diag}\left[\left(\widehat{\sigma}_1^{(i)}\right)^2, \left(\widehat{\sigma}_2^{(i)}\right)^2, \ldots, \left(\widehat{\sigma}_k^{(i)}\right)^2, 0\right].$$

Moreover, $e_1 \in \mathbb{R}^{k+1}$ is the first face of $\vec{e}_1$, $U^{(i)} \in \mathbb{R}^{(k+1)\times(k+1)}$ and $S^{(i)} \in \mathbb{R}^{(k+1)\times k}$ are the frontal slices of $\mathcal{U}$ and $\mathcal{S}$, respectively. Hence, by Proposition 4.3, we have

$$\lim_{\mu\to\infty} \widehat{J}_k^{(i)}(\mu) = \widehat{\mathbf{g}}^{(i)}(\widehat{\mathbf{z}}_1^{(i)})^2, \quad i = 1, 2, \ldots, n, \tag{4.53}$$

where $\widehat{\mathbf{g}}^{(i)} = \widehat{U}^{(i)}(1,:)\text{diag}[0,0,\ldots,0,1]\widehat{U}^{(i)}(1,:)^T$ is the square of the $(1, k+1)$st entry of $\widehat{U}^{(i)}$.

Transforming the resulting limiting tensor associated with (4.53) back to the tensor domain gives

$$\lim_{\mu\to\infty} \mathcal{J}_k(\mu) = \mathbf{z}_1^T * \mathbf{g} * \mathbf{z}_1,$$

where $\mathbf{g} = \mathcal{U}(1,:,:) * \mathcal{D} * \mathcal{U}(1,:,:)^T$. Hence,

$$\lim_{\mu\to\infty} \phi_k(\mu) = \Big(\lim_{\mu\to\infty} \mathcal{J}_k(\mu)\Big)_{(:,:,1)} = \Big(\mathbf{z}_1^T * \mathbf{g} * \mathbf{z}_1\Big)_{(:,:,1)}. \quad \square$$

It follows from Propositions 4.2 and 4.4 that the right-hand side of (4.51) is the infimum of $\phi_k(\mu)$ for $\mu > 0$. The right-hand side typically decreases quite rapidly as k increases, because making k larger increases the dimension of the subspace over which the least-squares problem (4.39) is minimized. Therefore, generally only a fairly small number of steps of Algorithm 6 are required in order to satisfy (4.49) for some $0 < \mu < \infty$.

We conclude this subsection with a result on the interlacing of the Frobenius norm of the singular tubes. An extension of Cauchy's interlacing theorem to third order tensors with the t-product has recently been described independently in [25]. The proof presented there is different from the one below.

Theorem 4.1. Given tensors $\mathcal{A} \in \mathbb{R}^{\ell\times m\times n}$ and $\mathcal{B} \in \mathbb{R}^{\ell\times s\times n}$ with $\ell \geq m$ and $s = m-1$. Suppose that $\mathcal{B}$ is obtained by deleting one or more columns of $\mathcal{A}$. Then the Frobenius norm of the singular tubes $\mathbf{b}_k$, $k = 1,2,\dots,s$, of $\mathcal{B}$ interlace the Frobenius norm of the singular tubes $\mathbf{a}_j$, $j = 1,2,\dots,m$, of $\mathcal{A}$. That is, let the singular tubes of $\mathcal{A}$ be ordered so that

$$\|\mathbf{a}_1\|_F \geq \|\mathbf{a}_2\|_F \geq \ldots \geq \|\mathbf{a}_m\|_F,$$

and let the singular tubes of $\mathcal{B}$ be ordered so that

$$\|\mathbf{b}_1\|_F \geq \|\mathbf{b}_2\|_F \geq \ldots \geq \|\mathbf{b}_s\|_F.$$

Then

$$\|\mathbf{a}_1\|_F \geq \|\mathbf{b}_1\|_F \geq \|\mathbf{a}_2\|_F \geq \ldots \geq \|\mathbf{a}_{m-1}\|_F \geq \|\mathbf{b}_s\|_F \geq \|\mathbf{a}_m\|_F.$$

Proof: Introduce the tSVD

$$\mathcal{A} = \mathcal{U}_1 * \mathcal{D} * \mathcal{V}_1^T,$$

where $\mathcal{D} = \mathrm{diag}[\mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_m] \in \mathbb{R}^{\ell\times m\times n}$ is f-diagonal, and $\mathcal{U}_1 \in \mathbb{R}^{\ell\times\ell\times n}$ and $\mathcal{V}_1 \in \mathbb{R}^{m\times m\times n}$ are orthogonal tensors. In the Fourier domain, the above decomposition is expressed as

$$\widehat{A}^{(i)} = \widehat{U}_1^{(i)} \widehat{D}^{(i)} (\widehat{V}_1^{(i)})^T, \quad i = 1,2,\dots,n,$$

where $\widehat{D}^{(i)} = \mathrm{diag}[\widehat{\alpha}_1^{(i)}, \widehat{\alpha}_2^{(i)}, \dots, \widehat{\alpha}_m^{(i)}]$ with $\widehat{\alpha}_1^{(i)} \geq \widehat{\alpha}_2^{(i)} \geq \ldots \geq \widehat{\alpha}_m^{(i)} \geq 0$.

Similarly, the tSVD of $\mathcal{B}$ in the Fourier domain is given by

$$\widehat{B}^{(i)} = \widehat{U}_2^{(i)} \widehat{C}^{(i)} (\widehat{V}_2^{(i)})^T, \quad i = 1,2,\dots,n,$$

where $\widehat{C}^{(i)} = \mathrm{diag}[\widehat{\beta}_1^{(i)}, \widehat{\beta}_2^{(i)}, \dots, \widehat{\beta}_s^{(i)}]$ with $\widehat{\beta}_1^{(i)} \geq \widehat{\beta}_2^{(i)} \geq \ldots \geq \widehat{\beta}_s^{(i)} \geq 0$. Therefore, by the interlacing property of singular values, the singular values of $\widehat{C}^{(i)}$ interlace the singular values of $\widehat{D}^{(i)}$, i.e.,

$$\widehat{\alpha}_1^{(i)} \geq \widehat{\beta}_1^{(i)} \geq \widehat{\alpha}_2^{(i)} \geq \ldots \geq \widehat{\alpha}_{m-1}^{(i)} \geq \widehat{\beta}_s^{(i)} \geq \widehat{\alpha}_m^{(i)} \geq 0.$$

Put each singular value into the corresponding tube, i.e., let

$$\widehat{\alpha}_j^{(i)} \quad \text{and} \quad \widehat{\beta}_k^{(i)} \quad i = 1,2,\dots,n, \quad j = 1,2,\dots,m, \quad k = 1,2,\dots,s,$$

be the faces of the tubal scalars $\widehat{\mathbf{a}}_j$ and $\widehat{\mathbf{b}}_k$, respectively. Then

$$\|\widehat{\mathbf{a}}_1\|_F \geq \|\widehat{\mathbf{b}}_1\|_F \geq \|\widehat{\mathbf{a}}_2\|_F \geq \ldots \geq \|\widehat{\mathbf{a}}_{m-1}\|_F \geq \|\widehat{\mathbf{b}}_s\|_F \geq \|\widehat{\mathbf{a}}_m\|_F.$$

The proof now follows from (2.11), i.e.,

$$\frac{1}{\sqrt{n}}\|\widehat{\mathbf{a}}_j\|_F = \|\mathbf{a}_j\|_F, \quad \frac{1}{\sqrt{n}}\|\widehat{\mathbf{b}}_k\|_F = \|\mathbf{b}_k\|_F, \quad j = 1,2,\dots,m, \quad k = 1,2,\dots,s. \quad \square$$

The following corollary is an interesting special case of Theorem 4.1.

Corollary 4.1. The Frobenius norm of the singular tubes of $\mathcal{T}_{k-1} := \widetilde{\mathcal{P}}_{k-1} * \widetilde{\mathcal{P}}_{k-1}^T$ interlace the Frobenius norm of the singular tubes of $\mathcal{T}_k := \widetilde{\mathcal{P}}_k * \widetilde{\mathcal{P}}_k^T$.

Proof: The result follows from Theorem 4.1. $\square$

4.2 The tGKT method for the approximate solution of (1.8)

This subsection generalizes the solution method of Subsection 4.1 to the solution of least squares problems of the form (1.8). The methods of this section can be applied to color image and video restoration. These problems are solved by several matrix-based methods in recent papers by Beik et al. [1], Bentbib et al. [3], and El Guide et al. [10, 11].

We will describe two algorithms for the approximate solution of (1.8). They both consider (1.8) as p separate Tikhonov minimization problems

$$\min_{\vec{\mathcal{X}}_j \in \mathbb{R}^{m\times 1\times n}} \{\|\mathcal{A} * \vec{\mathcal{X}}_j - \vec{\mathcal{B}}_j\|_F^2 + \frac{1}{\mu}\|\mathcal{L} * \vec{\mathcal{X}}_j\|_F^2\}, \quad j = 1, 2, \ldots, p, \tag{4.54}$$

where $\vec{\mathcal{B}}_1, \vec{\mathcal{B}}_2, \ldots, \vec{\mathcal{B}}_p$ are tensor columns of the data tensor $\mathcal{B}$ in (1.8). Both algorithms are based on the tGKB process and the tGKT method described in Subsection 4.1.

Let $\vec{\mathcal{B}}_{j,\text{true}}$ denote the unknown error-free tensor associated with the available error-contaminated tensor $\vec{\mathcal{B}}_j$ and assume that bounds for the norm of the errors

$$\mathcal{E}_j := \vec{\mathcal{B}}_j - \vec{\mathcal{B}}_{j,\text{true}}, \quad j = 1, 2, \ldots, p,$$

are available or can be estimated, i.e.,

$$\|\vec{\mathcal{E}}_j\|_F \leq \delta_j, \quad j = 1, 2, \ldots, p;$$

cf. (1.5) and (1.6).

The first algorithm we describe, Algorithm 6, solves each one of the p least-squares problems (4.54) independently. We refer to this approach of solving (4.54) as the tGKT$_p$ method.

Algorithm 6: The tGKT$_p$ method for the approximate solution of (1.8) by solving the problems (4.54) independently

```
   Input: $\mathcal{A}, \vec{\mathcal{B}}_1, \vec{\mathcal{B}}_2, \ldots, \vec{\mathcal{B}}_p, \delta_1, \delta_2, \ldots, \delta_p, \mathcal{L}, \eta > 1, k_{\text{init}} = 2$
1  for $j = 1, 2, \ldots, p$ do
2      $k \leftarrow k_{\text{init}}$, $[\vec{\mathcal{Q}}_1, \mathbf{z}_1] \leftarrow$ Normalize($\vec{\mathcal{B}}_j$)
3      Compute $\mathcal{W}_k$, $\mathcal{Q}_{k+1}$ and $\bar{\mathcal{P}}_k$ by Algorithm 4
4      Construct $\mathcal{R}_{\mathcal{L},k}$ by computing the tQR of $\mathcal{L} * \mathcal{W}_k$ using Algorithm 2
5      Compute $\widetilde{\mathcal{P}}_k \leftarrow \bar{\mathcal{P}}_k * \mathcal{R}_{\mathcal{L},k}^{-1}$, and $\vec{e}_1 \leftarrow \mathcal{I}(:,1,:)$
6      Solve the minimization problem
           $$\min_{\vec{\mathcal{Z}} \in \mathbb{R}^{k\times 1\times n}} \|\widetilde{\mathcal{P}}_k * \vec{\mathcal{Z}} - \vec{e}_1 * \mathbf{z}_1\|_F$$
        for $\vec{\mathcal{Z}}_k$ by using Algorithm 3
7      while $\|\widetilde{\mathcal{P}}_k * \vec{\mathcal{Z}}_k - \vec{e}_1 * \mathbf{z}_1\|_F \geq \eta\delta_j$ do
8          $k \leftarrow k + 1$
9          Go to step 3
10     end
11     Determine the regularization parameter by the discrepancy principle, i.e., use
        Algorithm 5 to compute the zero $\mu_k > 0$ of
           $$\xi_k(\mu_k) := \|\widetilde{\mathcal{P}}_k * \vec{\mathcal{Z}}_{j,\mu_k} - \vec{e}_1 * \mathbf{z}_1\|_F^2 - \eta^2\delta_j^2$$
        and the associated solution $\vec{\mathcal{Z}}_{j,\mu_k}$ of
           $$\min_{\vec{\mathcal{Z}} \in \mathbb{R}^{k\times 1\times n}} \left\| \begin{bmatrix} \widetilde{\mathcal{P}}_k \\ \mu_k^{-1/2}\mathcal{I} \end{bmatrix} * \vec{\mathcal{Z}} - \begin{bmatrix} \vec{e}_1 * \mathbf{z}_1 \\ \vec{\mathcal{O}} \end{bmatrix} \right\|_F$$
        by using Algorithm 3
12     Compute $\vec{\mathcal{Y}}_{j,\mu_k} \leftarrow \mathcal{R}_{\mathcal{L},k}^{-1} * \vec{\mathcal{Z}}_{j,\mu_k}$,  $\vec{\mathcal{X}}_{j,\mu_k} \leftarrow \mathcal{W}_k * \vec{\mathcal{Y}}_{j,\mu_k}$
13 end
```

Our second algorithm of this subsection generates a t-Krylov subspace $\mathcal{K}_k(\mathcal{A}^T * \mathcal{A}, \mathcal{A}^T * \vec{\mathcal{B}}_1)$ of dimension k large enough to contain sufficiently accurate approximate solutions of all the p least-squares problems (4.54). Thus, we first solve the least-squares problem (4.54) for $p = 1$ by Algorithm 6. Then we seek to solve the least-squares problem (4.54) for $p = 2$ by using the same t-Krylov subspace $\mathcal{K}_k(\mathcal{A}^T * \mathcal{A}, \mathcal{A}^T * \vec{\mathcal{B}}_1)$. If the discrepancy principle cannot be satisfied, then the dimension k of the t-Krylov subspace is increased until the discrepancy principle can be satisfied. Having solved the least-squares problem for $p = 2$, we proceed similarly to solve the remaining problems (4.54) for $j = 3, 4, \ldots, p$. The details are described in Algorithm 7. The tGKB process is implemented with reorthogonalization when applied in Algorithm 7 to yield the expressions $\mathcal{Q}_{k+1}^T * \vec{\mathcal{B}}_j$ with sufficient accuracy. When the required number of bidiagonalization steps k is large, it may be beneficial to restart Algorithm 7 with a new tensor $\vec{\mathcal{B}}_j$. This approach of using nested t-Krylov subspaces is referred to as the `nested_`tGKT$_p$ method.

Algorithm 7: The `nested_`tGKT$_p$ method for the approximate solution of (1.8) by solving the problems (4.54) using nested t-Krylov subspaces

```
   Input: $\mathcal{A}, \vec{\mathcal{B}}_1, \vec{\mathcal{B}}_2, \ldots, \vec{\mathcal{B}}_p$, $\delta_1, \delta_2, \ldots, \delta_p, \mathcal{L}, \eta > 1$, $k_{\text{init}} = 2$
 1 $k \leftarrow k_{\text{init}}$, $[\vec{\mathcal{Q}}_1, \mathbf{z_1}] \leftarrow \texttt{Normalize}(\vec{\mathcal{B}}_1)$ by Algorithm 1
 2 Compute $\mathcal{W}_k, \mathcal{Q}_{k+1}$ and $\bar{\mathcal{P}}_k$ by Algorithm 4 with reorthogonalization of the tensor columns
     of $\mathcal{W}_k$ and $\mathcal{Q}_{k+1}$
 3 Construct $\mathcal{R}_{\mathcal{L},k}$ by computing the tQR of $\mathcal{L} * \mathcal{W}_k$ using Algorithm 2
 4 Compute $\widetilde{\mathcal{P}}_k \leftarrow \bar{\mathcal{P}}_k * \mathcal{R}_{\mathcal{L},k}^{-1}$
 5 Solve the minimization problem
       $$\min_{\vec{\mathcal{Z}} \in \mathbb{R}^{k \times 1 \times n}} \|\widetilde{\mathcal{P}}_k * \vec{\mathcal{Z}} - \mathcal{Q}_{k+1}^T * \vec{\mathcal{B}}_1\|_F$$
     for $\vec{\mathcal{Z}}_k$ by using Algorithm 3
 6 while $\|\widetilde{\mathcal{P}}_k * \vec{\mathcal{Z}}_k - \mathcal{Q}_{k+1}^T * \vec{\mathcal{B}}_1\|_F \geq \eta\delta_1$ do
 7     $k \leftarrow k + 1$
 8     Go to step 2
 9 end
10 Determine the regularization parameter by the discrepancy principle, i.e., use an analogue
     of Algorithm 5 to compute the zero $\mu_k > 0$ of
       $$\xi_k(\mu_k) := \|\widetilde{\mathcal{P}}_k * \vec{\mathcal{Z}}_{1,\mu_k} - \mathcal{Q}_{k+1}^T * \vec{\mathcal{B}}_1\|_F^2 - \eta^2\delta_1^2$$
     and the associated solution $\vec{\mathcal{Z}}_{1,\mu_k}$ of
       $$\min_{\vec{\mathcal{Z}}_1 \in \mathbb{R}^{k \times 1 \times n}} \left\| \begin{bmatrix} \widetilde{\mathcal{P}}_k \\ \mu_k^{-1/2}\mathcal{I} \end{bmatrix} * \vec{\mathcal{Z}}_1 - \begin{bmatrix} \mathcal{Q}_{k+1}^T * \vec{\mathcal{B}}_1 \\ \vec{\mathcal{O}} \end{bmatrix} \right\|_F$$
     by using Algorithm 3
11 Compute $\vec{\mathcal{Y}}_{1,\mu_k} \leftarrow \mathcal{R}_{\mathcal{L},k}^{-1} * \vec{\mathcal{Z}}_{1,\mu_k}$, $\vec{\mathcal{X}}_{1,\mu_k} \leftarrow \mathcal{W}_k * \vec{\mathcal{Y}}_{1,\mu_k}$
12 for $j = 2, 3, \ldots, p$ do
13     $[\vec{\mathcal{Q}}_1, \mathbf{z_1}] \leftarrow \texttt{Normalize}(\vec{\mathcal{B}}_j)$
14     while $\|\widetilde{\mathcal{P}}_k * \vec{\mathcal{Z}}_k - \mathcal{Q}_{k+1}^T * \vec{\mathcal{B}}_j\|_F \geq \eta\delta_j$ do
15         $k \leftarrow k + 1$
16         Repeat steps 2-5 with the present tensors $\widetilde{\mathcal{P}}_k$, $\mathcal{Q}_{k+1}^T$, and $\vec{\mathcal{B}}_j$
17     end
18     Repeat step 10 with the present $\delta_j$, and tensors $\widetilde{\mathcal{P}}_k$, $\mathcal{Q}_{k+1}^T$, and $\vec{\mathcal{B}}_j$ to compute $\vec{\mathcal{Z}}_{j,\mu_k}$
19     Compute $\vec{\mathcal{Y}}_{j,\mu_k} \leftarrow \mathcal{R}_{\mathcal{L},k}^{-1} * \vec{\mathcal{Z}}_{j,\mu_k}$, $\vec{\mathcal{X}}_{j,\mu_k} \leftarrow \mathcal{W}_k * \vec{\mathcal{Y}}_{j,\mu_k}$
20 end
```

5 The GG-tGKT and G-tGKT Methods for the Approximate Solution of (1.8) and (1.2)

We describe the GG-tGKT and G-tGKT methods. Both methods involve flattening, and their performance will be illustrated in computed examples in Section 6.

5.1 The GG-tGKT method for the approximate solution of (1.8)

This section generalizes and modifies the T-global Golub-Kahan-Tikhonov regularization method, recently described by El Guide et al. [11] for the approximate solution of (1.8) with $\mathcal{L} = \mathcal{I}$, to allow a general regularization operator $\mathcal{L} \neq \mathcal{I}$. This generalization requires Algorithm 9 below. We refer to our generalization of the method by El Guide et al. [11] as the generalized global tGKT (GG-tGKT) method. Differently from El Guide et al. [11], we use the discrepancy principle to determine the regularization parameter.

The GG-tGKT method is based on first reducing $\mathcal{A}$ to a small bidiagonal matrix by the application of a few, $1 \leq k \ll \min\{\ell, m\}$, steps of the generalized global t-product Golub-Kahan bidiagonalization (GG-tGKB) process. This process is implemented by Algorithm 8. It uses the Golub-Kahan Bidiagonalization Based Tensor Format (GKB_BTF) recently described by Beik et al. [1]. A version of the GG-tGKB algorithm with different initializations has been presented in [11]. We assume that the number of steps, k, is small enough to avoid breakdown. Then applying k steps of the partial GG-tGKB process to $\mathcal{A}$ yields the decompositions

$$\mathcal{A} * \mathbb{W}_k = \mathbb{Q}_{k+1} \circledast \bar{P}_k, \;\; \mathcal{A}^T * \mathbb{Q}_k = \mathbb{W}_k \circledast P_k^T, \tag{5.55}$$

where

$$\mathbb{W}_k := [\mathcal{W}_1, \ldots, \mathcal{W}_k] \in \mathbb{R}^{m \times kp \times n}, \;\; \mathbb{Q}_{k+1} := [\mathcal{Q}_1, \ldots, \mathcal{Q}_{k+1}] \in \mathbb{R}^{\ell \times (k+1)p \times n},$$

and

$$\begin{aligned} \mathcal{A} * \mathbb{W}_k &:= [\mathcal{A} * \mathcal{W}_1, \mathcal{A} * \mathcal{W}_2, \ldots, \mathcal{A} * \mathcal{W}_k] \in \mathbb{R}^{\ell \times kp \times n}, \\ \mathbb{Q}_{k+1} \circledast \bar{P}_k &:= [\mathbb{Q}_{k+1} \circledast \bar{P}_k(:,1), \mathbb{Q}_{k+1} \circledast \bar{P}_k(:,2), \ldots, \mathbb{Q}_{k+1} \circledast \bar{P}_k(:,k)] \in \mathbb{R}^{\ell \times kp \times n}. \end{aligned} \tag{5.56}$$

The tensors $\mathcal{A}^T * \mathbb{Q}_k$ and $\mathbb{W}_k \circledast P_k^T$ are defined similarly as (5.56). Details of the computations are described by Algorithm 8. The tensors $\mathcal{Q}_j \in \mathbb{R}^{\ell \times p \times n}$ and $\mathcal{W}_j \in \mathbb{R}^{m \times p \times n}$, $j = 1, 2, \ldots, k$, generated by the algorithm form orthonormal tensor bases for the t-Krylov subspaces $\mathcal{K}_k(\mathcal{A} * \mathcal{A}^T, \mathcal{B})$ and $\mathcal{K}_k(\mathcal{A}^T * \mathcal{A}, \mathcal{A}^T * \mathcal{B})$, respectively. The lower bidiagonal matrix $\bar{P}_k$ in (5.56) is given by

$$\bar{P}_k = \begin{bmatrix} \alpha_1 & & & & \\ \beta_2 & \alpha_2 & & & \\ & \beta_3 & \alpha_3 & & \\ & & \ddots & \ddots & \\ & & & \beta_k & \alpha_k \\ & & & & \beta_{k+1} \end{bmatrix} \in \mathbb{R}^{(k+1) \times k}, \tag{5.57}$$

and P_k is the leading $k \times k$ submatrix of $\bar{P}_k$. The relation

$$\mathcal{B} = \mathbb{Q}_{k+1} \circledast e_1 \beta_1, \quad e_1 = [1, 0, \ldots, 0]^T \tag{5.58}$$

is easily deduced from Algorithm 8.

We compute an approximate solution of (1.8) analogously as described in Subsection 4.1. Thus, letting $\mathcal{X} = \mathbb{W}_k \circledast y$ and using the left-hand side of (5.55), and (5.58), the minimization problem (1.8) reduces to

$$\min_{y \in \mathbb{R}^k} \{\|\mathbb{Q}_{k+1} \circledast \bar{P}_k \circledast y - \mathbb{Q}_{k+1} \circledast e_1 \beta_1\|_F^2 + \mu^{-1} \|\mathcal{L} * \mathbb{W}_k \circledast y\|_F^2\}. \tag{5.59}$$

Algorithm 8: Partial generalized global tGKB (GG-tGKB)

```
    Input: $\mathcal{A} \in \mathbb{R}^{\ell\times m\times n}$, $\mathcal{B} \in \mathbb{R}^{\ell\times p\times n}$ such that $\mathcal{A}^T * \mathcal{B} \neq \mathcal{O}$
1   $\beta_1 \leftarrow \|\mathcal{B}\|_F$, $\mathcal{Q}_1 \leftarrow \frac{1}{\beta_1}\mathcal{B}$, $\mathcal{W}_0 \leftarrow \mathcal{O}$
2   for $j = 1, 2, \ldots, k$ do
3       $\mathcal{W} \leftarrow \mathcal{A}^T * \mathcal{Q}_j - \beta_j \mathcal{W}_{j-1}$
4       $\alpha_j \leftarrow \|\mathcal{W}\|_F$. If $\alpha_j = 0$ stop else $\mathcal{W}_j \leftarrow \mathcal{W}/\alpha_j$
5       $\mathcal{Q} \leftarrow \mathcal{A} * \mathcal{W}_j - \alpha_j \mathcal{Q}_j$
6       $\beta_{j+1} \leftarrow \|\mathcal{Q}\|_F$. If $\beta_{j+1} = 0$ stop else $\mathcal{Q}_{j+1} \leftarrow \mathcal{Q}/\beta_{j+1}$
7   end
```

Algorithm 9: Generalized global tensor QR (GG-tQR) decomposition

```
    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: $\mathbb{Q} := [\mathcal{Q}_1, \mathcal{Q}_2, \ldots, \mathcal{Q}_k] \in \mathbb{R}^{\ell\times km\times n}$, $R = (r_{ij}) \in \mathbb{R}^{k\times k}$ such that $\mathcal{A} = \mathbb{Q} \circledast R$ and
            $\mathbb{Q}^T \lozenge \mathbb{Q} = I_k$
1   Set $r_{11} \leftarrow \langle \mathcal{A}_1, \mathcal{A}_1 \rangle^{1/2}$, $\mathcal{Q}_1 \leftarrow \frac{1}{r_{11}}\mathcal{A}_1$
2   for $j = 1, 2, \ldots, k$ do
3       $\mathcal{W} \leftarrow \mathcal{A}_j$
4       for $i = 1, 2, \ldots, j-1$ do
5           $r_{ij} \leftarrow \langle \mathcal{Q}_i, \mathcal{W} \rangle$
6           $\mathcal{W} \leftarrow \mathcal{W} - r_{ij}\mathcal{Q}_i$
7       end
8       $r_{jj} \leftarrow \langle \mathcal{W}, \mathcal{W} \rangle^{1/2}$
9       $\mathcal{Q}_j \leftarrow \mathcal{W}/r_{jj}$
10  end
```

Using the $\circledast$ product (2.16) introduced by El Guide et al. [11], we define a generalized global tQR (GG-tQR) factorization, which is applied in Algorithm 10 to compute the decomposition

$$\mathcal{L} * \mathbb{W}_k = \mathbb{Q}_{\mathcal{L},k} \circledast R_{\mathcal{L},k} \in \mathbb{R}^{s\times kp\times n}, \tag{5.60}$$

where $R_{\mathcal{L},k} \in \mathbb{R}^{k\times k}$ is an upper triangular matrix and $\mathbb{Q}_{\mathcal{L},k} \in \mathbb{R}^{s\times kp\times n}$ has k orthogonal tensor columns. Analogues of Algorithm 9 that apply the n-mode and Einstein products have recently been described by El Ichi et al. [12].

Substituting (5.60) into (5.59), and using the left-hand side of (2.17), gives

$$\min_{y\in\mathbb{R}^k} \{\|\bar{P}_k y - e_1\beta_1\|_2^2 + \mu^{-1}\|R_{\mathcal{L},k} y\|_2^2\}. \tag{5.61}$$

Typically, the matrix $R_{\mathcal{L},k}$ is nonsingular and not very ill-conditioned. Then we can express (5.61) as a Tikhonov minimization problem in standard form,

$$\min_{z\in\mathbb{R}^k} \{\|\widetilde{P}_k z - e_1\beta_1\|_2^2 + \mu^{-1}\|z\|_2^2\}, \tag{5.62}$$

where

$$z := R_{\mathcal{L},k} y, \quad \widetilde{P}_k := \bar{P}_k R_{\mathcal{L},k}^{-1}. \tag{5.63}$$

The minimization problem (5.62) is analogous to (4.45). Its solution can be computed fairly stably by solving

$$\min_{z\in\mathbb{R}^k} \left\| \begin{bmatrix} \widetilde{P}_k \\ \mu^{-1/2} I \end{bmatrix} z - \begin{bmatrix} e_1\beta_1 \\ 0 \end{bmatrix} \right\|_2. \tag{5.64}$$

Denote the solution by $z_{\mu,k}$. Then the computed approximate solution of (1.8) is given by

$$\vec{\mathcal{X}}_{\mu,k} = \mathbb{W}_k \circledast R_{\mathcal{L},k}^{-1} z_{\mu,k}.$$

We determine the regularization parameter μ by the discrepancy principle based on the Frobenius norm. This assumes knowledge of a bound

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

for the error tensor $\mathcal{E}$ in $\mathcal{B}$. Thus, we choose $\mu > 0$ so that the solution $z_{\mu,k}$ of (5.64) satisfies

$$\|\widetilde{P}_k z_{\mu,k} - e_1\beta_1\|_2 = \eta\delta. \tag{5.65}$$

Define the function

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

where $z_{\mu,k}$ solves (5.64). Manipulations similar to those applied in Subsection 4.1 show that $\psi_k(\mu)$ can be expressed as (4.50). Proposition 4.3 shows that the discrepancy principle (5.65) can be satisfied by choosing k large enough. It is readily verified by following the proof of Proposition 3.2 that the function $\mu \mapsto \psi_k(\mu)$ is decreasing and convex with $\psi_k(0) = \beta_1^2$. Therefore, through a similar reasoning as in Subsection 4.1, it may be convenient to solve

$$\psi_k(\mu) - \eta^2\delta^2 = 0$$

by Newton's method, i.e., by an analogue of Algorithm 5 with initial approximate solution $\mu_0 = 0$.

We turn to the matrix analogue of Proposition 4.1.

Proposition 5.1. Let μ_k solve $\psi_k(\mu) = \eta^2\delta^2$ and suppose that $z_{\mu,k}$ is the solution of (5.62) with $\mu = \mu_k$. Let $y_{\mu,k}$ and $z_{\mu,k}$ be related by (5.63). Then the associated approximate solution $\mathcal{X}_{\mu,k} = \mathbb{W}_k \circledast y_{\mu,k}$ of (1.8) satisfies

$$\|\mathcal{A} * \mathcal{X}_{\mu,k} - \mathcal{B}\|_F^2 = \beta_1^2 e_1^T (\mu \widetilde{P}_k \widetilde{P}_k^T + I)^{-2} e_1. \tag{5.66}$$

Proof: Substituting $\mathcal{X}_{\mu,k} = \mathbb{W}_k \circledast y_{\mu,k}$ into left-hand side of (5.66), using the left-hand side of (5.55) and (5.58), as well as left-hand side of (2.17) gives

$$\|\mathcal{A} * \mathcal{X}_{\mu,k} - \mathcal{B}\|_F^2 = \|\mathbb{Q}_{k+1} \circledast (\bar{P}_k \circledast y_{\mu,k} - e_1\beta_1)\|_F^2 = \|\bar{P}_k y_{\mu,k} - e_1\beta_1\|_2^2 = \|\widetilde{P}_k z_{\mu,k} - e_1\beta_1\|_2^2. \quad \square$$

We refer to the solution method described above as the GG-tGKT method. This method is implemented by Algorithm 10. It works with the lateral slices $\vec{\mathcal{B}}_j$, $j = 1, 2, \ldots, p$, of the data tensor $\mathcal{B}$ simultaneously.

Algorithm 10: The GG-tGKT method for computing an approximate solution of (1.8)

```
   Input: $\mathcal{A}$, $\mathcal{B}$, $\delta, \mathcal{L}$, $\eta > 1$, $k_{\text{init}} = 2$
1  $k \leftarrow k_{\text{init}}$, $\beta_1 \leftarrow \|\mathcal{B}\|_F$, $\mathcal{Q}_1 \leftarrow \frac{1}{\beta_1}\mathcal{B}$
2  Compute $\mathbb{W}_k$, $\mathbb{Q}_{k+1}$, and $\bar{P}_k$ by Algorithm 8
3  Construct $R_{\mathcal{L},k}$ by computing the GG-tQR of $\mathcal{L} * \mathbb{W}_k$ using Algorithm 9
4  Compute $\widetilde{P}_k \leftarrow \bar{P}_k R_{\mathcal{L},k}^{-1}$
5  Let $z_k \in \mathbb{R}^k$ solve the minimization problem
       $$\min_{z \in \mathbb{R}^k} \|\widetilde{P}_k z - e_1\beta_1\|_2$$
   while $\|\widetilde{P}_k z_k - e_1\beta_1\|_2 \geq \eta\delta$ do
6      $k \leftarrow k + 1$
7      Go to step 2
8  end
9  Determine the regularization parameter by the discrepancy principle, i.e., use an analogue
     of Algorithm 5 to compute the zero $\mu_k > 0$ of
       $$\varphi_k(\mu) := \|\widetilde{P}_k z_{\mu,k} - e_1\beta_1\|_2^2 - \eta^2\delta^2$$
     and the associated solution $z_{\mu,k}$ of
       $$\min_{z \in \mathbb{R}^k} \left\| \begin{bmatrix} \widetilde{P}_k \\ \mu_k^{-1/2} I \end{bmatrix} z - \begin{bmatrix} e_1\beta_1 \\ 0 \end{bmatrix} \right\|_2$$
10 Compute $y_{\mu,k} \leftarrow R_{\mathcal{L},k}^{-1} z_{\mu,k}$,  $\mathcal{X}_{\mu,k} \leftarrow \mathbb{W}_k \circledast y_{\mu,k}$
```

5.2 The G-tGKT method for the approximate solution of (1.2) and (1.8)

This subsection described the global t-product Golub-Kahan-Tikhonov (G-tGKT) method for the approximate solution of (1.2) and (1.8). An alternative to the GG-tGKT method described by Algorithm 10 is to work with each lateral slice $\vec{\mathcal{B}}_j$, $j = 1, 2, \ldots, p$, of the tensor $\mathcal{B}$ independently. Thus, one solves the p Tikhonov minimization problems (4.54) separately. Such a solution method is implemented by Algorithm 13, and is referred to as the G-tGKT$_p$ method.

The G-tGKT method for the approximate solution of (1.2) first reduces $\mathcal{A}$ in (1.2) to a small bidiagonal matrix by carrying out a few, say k, steps of the global t-product Golub-Kahan bidiagonalization (G-tGKB) process, which is described by Algorithm 11. It is readily implemented by taking $p = 1$ in Algorithm 8. We choose k small enough to avoid breakdown. Algorithm 11 yields the partial G-tGKB decompositions

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

where

$$\mathcal{W}_k := [\vec{\mathcal{W}}_1, \ldots, \vec{\mathcal{W}}_k] \in \mathbb{R}^{m \times k \times n}, \;\; \mathcal{Q}_{k+1} := [\vec{\mathcal{Q}}_1, \ldots, \vec{\mathcal{Q}}_{k+1}] \in \mathbb{R}^{\ell \times (k+1) \times n}.$$

The expressions $\mathcal{A} * \mathcal{W}_k$, $\mathcal{Q}_{k+1} \circledast \bar{B}_k$, $\mathcal{A}^T * \mathcal{Q}_k$, and $\mathcal{W}_k \circledast B_k^T$ are defined similarly to (5.56). The matrix $\bar{B}_k \in \mathbb{R}^{(k+1) \times k}$ has a form analogous to (5.57), and B_k is the leading $k \times k$ submatrix of $\bar{B}_k$. The tensors $\vec{\mathcal{Q}}_j \in \mathbb{R}^{\ell \times 1 \times n}$, for $j = 1, 2, \ldots, k+1$, and $\vec{\mathcal{W}}_j \in \mathbb{R}^{m \times 1 \times n}$, for $j = 1, 2, \ldots, k$, generated by Algorithm 11 form orthonormal tensor bases for the t-Krylov subspaces $\mathcal{K}_{k+1}(\mathcal{A} * \mathcal{A}^T, \vec{\mathcal{B}})$ and $\mathcal{K}_k(\mathcal{A}^T * \mathcal{A}, \mathcal{A}^T * \vec{\mathcal{B}})$, respectively.

Algorithm 11: Partial global tGKB (G-tGKB)

```
   Input: $\mathcal{A} \in \mathbb{R}^{\ell \times m \times n}$, $\vec{\mathcal{B}} \in \mathbb{R}^{\ell \times 1 \times n}$ such that $\mathcal{A}^T * \vec{\mathcal{B}} \neq \vec{\mathcal{O}}$
1  $\beta_1 \leftarrow \|\vec{\mathcal{B}}\|_F, \vec{\mathcal{Q}}_1 \leftarrow \frac{1}{\beta_1}\vec{\mathcal{B}}$, $\vec{\mathcal{W}}_0 \leftarrow \vec{\mathcal{O}}$.
2  for $j = 1, 2, \ldots, k$ do
3      $\vec{\mathcal{W}} \leftarrow \mathcal{A}^T * \vec{\mathcal{Q}}_j - \beta_j \vec{\mathcal{W}}_{j-1}$
4      $\alpha_j \leftarrow \|\vec{\mathcal{W}}\|_F$. If $\alpha_j = 0$, stop else
5      $\vec{\mathcal{W}}_j \leftarrow \vec{\mathcal{W}}/\alpha_j$
6      $\vec{\mathcal{Q}} \leftarrow \mathcal{A} * \vec{\mathcal{W}}_j - \alpha_j \vec{\mathcal{Q}}_j$
7      $\beta_{j+1} \leftarrow \|\vec{\mathcal{Q}}\|_F$. If $\beta_{j+1} = 0$, stop else
8      $\vec{\mathcal{Q}}_{j+1} \leftarrow \vec{\mathcal{Q}}/\beta_{j+1}$
9  end
```

Algorithm 12: Global tensor QR (G-tQR) decomposition

```
   Input: $\mathcal{A} := [\vec{\mathcal{A}}_1, \vec{\mathcal{A}}_2, \ldots, \vec{\mathcal{A}}_k] \in \mathbb{R}^{\ell \times k \times n}$, $\vec{\mathcal{A}}_j \in \mathbb{R}^{\ell \times 1 \times n}$, $j = 1, 2, \ldots, k$, $\ell \geq k$
   Output: $\mathcal{Q} := [\vec{\mathcal{Q}}_1, \vec{\mathcal{Q}}_2, \ldots, \vec{\mathcal{Q}}_k] \in \mathbb{R}^{\ell \times k \times n}$, $\vec{\mathcal{Q}}_j \in \mathbb{R}^{\ell \times 1 \times n}$, $\bar{R} = (r_{ij}) \in \mathbb{R}^{k \times k}$ such that
           $\mathcal{A} = \mathcal{Q} \circledast \bar{R}$, and $\mathcal{Q}^T \lozenge \mathcal{Q} = I_k$
1  $r_{11} \leftarrow \langle \vec{\mathcal{A}}_1, \vec{\mathcal{A}}_1 \rangle^{1/2}$, $\vec{\mathcal{Q}}_1 \leftarrow \frac{1}{r_{11}} \vec{\mathcal{A}}_1$
2  for $j = 1, 2, \ldots, k$ do
3      $\vec{\mathcal{W}} \leftarrow \vec{\mathcal{A}}_j$
4      for $i = 1, 2, \ldots, j-1$ do
5          $r_{ij} \leftarrow \langle \vec{\mathcal{Q}}_i, \vec{\mathcal{W}} \rangle$
6          $\vec{\mathcal{W}} \leftarrow \vec{\mathcal{W}} - r_{ij} \vec{\mathcal{Q}}_i$
7      end
8      $r_{jj} \leftarrow \langle \vec{\mathcal{W}}, \vec{\mathcal{W}} \rangle^{1/2}$
9      $\vec{\mathcal{Q}}_j \leftarrow \vec{\mathcal{W}}/r_{jj}$
10 end
```

Let $\vec{\mathcal{X}} = \mathcal{W}_k \circledast y$. Then following a similar approach as in Section 4, we reduce (1.2) to

$$\min_{y \in \mathbb{R}^k} \{ \|\mathcal{Q}_{k+1} \circledast \bar{B}_k \circledast y - \mathcal{Q}_{k+1} \circledast e_1 \beta_1\|_F^2 + \mu^{-1} \|\mathcal{L} * \mathcal{W}_k \circledast y\|_F^2 \}. \tag{5.67}$$

Compute the global tQR (G-tQR) factorization of $\mathcal{L} * \mathcal{W}_k$ by using Algorithm 12 to obtain

$$\mathcal{L} * \mathcal{W}_k = \mathcal{Q}_{\mathcal{L},k} \circledast \bar{R}_{\mathcal{L},k}, \tag{5.68}$$

where $\mathcal{Q}_{\mathcal{L},k} \in \mathbb{R}^{s\times k\times n}$ has k orthonormal tensor columns and the matrix $\bar{R}_{\mathcal{L},k} \in \mathbb{R}^{k\times k}$ is upper triangular. Note that the G-tQR algorithm is readily obtained by taking $m = 1$ in Algorithm 9, in which case $\ell \geq k$.

Substitute (5.68) into (5.67), use the right-hand side of (2.17), and define

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

where we assume the matrix $\bar{R}_{\mathcal{L},k}$ to be invertible and not very ill-conditioned. We obtain the Tikhonov minimization problem in standard form

$$\min_{z\in\mathbb{R}^k} \{\|\breve{P}_k z - e_1\beta_1\|_2^2 + \mu^{-1}\|z\|_2^2\}. \tag{5.69}$$

This problem can be solved similarly as (5.62), whose solution has been described previously. We refer to this approach of solving (5.69), and thereby of computing an approximate solution of (1.2), as the G-tGKT method. It is implemented by Algorithm 13 with $p = 1$. Algorithm 13 with $p > 1$ is used to solve (1.8).

Algorithm 13: The G-tGKT$_p$ method for the approximate solution of (1.8).

```
   Input: $\mathcal{A}$, $\vec{\mathcal{B}}_1, \vec{\mathcal{B}}_2, \ldots, \vec{\mathcal{B}}_p$, $\mathcal{L}$, $\delta_1, \delta_2, \ldots, \delta_p$, $\eta > 1$, $k_{\text{init}} = 2$
1  for $j = 1, 2, \ldots, p$ do
2      $k \leftarrow k_{\text{init}}$, $\beta_1 \leftarrow \|\vec{\mathcal{B}}_j\|_F$, $\vec{\mathcal{Q}}_1 \leftarrow \frac{1}{\beta_1}\vec{\mathcal{B}}_j$
3      Compute $\mathcal{W}_k$, $\mathcal{Q}_{k+1}$ and $\bar{B}_k$ by Algorithm 11
4      Construct $\bar{R}_{\mathcal{L},k}$ by computing the G-tQR factorization of $\mathcal{L} * \vec{\mathcal{W}}_k$ using Algorithm 12
5      Compute $\breve{P}_k \leftarrow \bar{B}_k \bar{R}_{\mathcal{L},k}^{-1}$
6      Solve the minimization problem
           $$\min_{z\in\mathbb{R}^k} \|\breve{P}_k z - e_1\beta_1\|_2$$
        for $z_k$
7      while $\|\breve{P}_k z_k - e_1\beta_1\|_2 \geq \eta\delta_j$ do
8          $k \leftarrow k + 1$
9          Go to step 3
10     end
11     Determine the regularization parameter by the discrepancy principle, i.e., proceed
        similarly as in Algorithm 5 to compute the zero $\mu_k > 0$ of
           $$\varphi_k(\mu_k) := \|\breve{P}_k z_{j,\mu_k} - e_1\beta_1\|_2^2 - \eta^2\delta_j^2$$
        and the associated solution $z_{j,\mu_k}$ of
           $$\min_{z\in\mathbb{R}^k} \left\| \begin{bmatrix} \breve{P}_k \\ \mu_k^{-1/2} I \end{bmatrix} z - \begin{bmatrix} e_1\beta_1 \\ 0 \end{bmatrix} \right\|_2$$
12     Compute $y_{j,\mu_k} \leftarrow \bar{R}_{\mathcal{L},k}^{-1} z_{j,\mu_k}$, $\vec{\mathcal{X}}_{j,\mu_k} \leftarrow \vec{\mathcal{W}}_k \circledast y_{j,\mu_k}$
13 end
```

6 Numerical Examples

This section illustrates the performance of the methods described when applied to the solution of linear discrete ill-posed problems. Unless otherwise stated, $\mathcal{A} \in \mathbb{R}^{256\times 256\times 256}$ in all computed examples. All computations are carried out in MATLAB 2019b on a Lenovo computer running Windows 10 with Intel Core i3 and 4 GB RAM. The discrepancy principle is used in all computed examples to determine the regularization parameter and the number of bidiagonalization steps. The condition numbers of the frontal slices of $\mathcal{A}$ are computed using the MATLAB function `cond`. We set $\texttt{tol} = 10^{-12}$ in Algorithm 1.

Let $\vec{\mathcal{X}}_{\text{method}}$ be the computed solution of the minimization problem (1.1) by a chosen method. To compare the quality of the computed solution, we evaluate the relative error

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

The relative error for problems whose data is given by a three-way tensor $\mathcal{B}$ is determined analogously.

We generate a "noise" tensor $\mathcal{E}$ that simulates the error in the data tensor $\mathcal{B} = \mathcal{B}_{\text{true}} + \mathcal{E}$ by determining the lateral slices $\vec{\mathcal{E}}_j$, $j = 1, 2, \ldots, p$, of $\mathcal{E}$. Their entries are normally distributed with zero mean and are scaled to correspond to a specific noise level $\widetilde{\delta}$. Thus,

$$\vec{\mathcal{E}}_j := \widetilde{\delta} \frac{\vec{\mathcal{E}}_{0,j}}{\|\vec{\mathcal{E}}_{0,j}\|_F} \|\vec{\mathcal{B}}_{\text{true},j}\|_F, \quad j = 1, 2, \ldots, p, \tag{6.70}$$

where the entries of $\vec{\mathcal{E}}_{0,j}$ are distributed according to $N(0, 1)$. For problems of the form (1.1), we have $p = 1$. Throughout this section, a table entry "-" indicates that the solution method carries out several different numbers of bidiagonalization steps or computes several different values of the regularization parameter.

Example 6.1. This example implements the tGKT and G-tGKT methods with the regularization tensor $\mathcal{L}_2 \in \mathbb{R}^{255\times256\times256}$ described in Subsection 4.1. Let the matrix $A_1 = \texttt{baart}(256)$ be generated by the function `baart` from Hansen's Regularization Tools [16] and define $A_2 = \texttt{gallery}('\texttt{prolate}', 256, \alpha)$ in MATLAB. We take $\alpha = 0.46$. Then A_2 is a symmetric positive definite ill-conditioned Toeplitz matrix. To generate $\mathcal{A}$, we let

$$\mathcal{A}^{(i)} = A_1(i, 1)A_2, \quad i = 1, 2, \ldots, 256.$$

The exact data tensor $\vec{\mathcal{B}}_{\text{true}} \in \mathbb{R}^{256\times1\times256}$ is given by $\vec{\mathcal{B}}_{\text{true}} = \mathcal{A} * \vec{\mathcal{X}}_{\text{true}}$, where the exact solution $\vec{\mathcal{X}}_{\text{true}} \in \mathbb{R}^{256\times1\times256}$ has all entries equal to unity. The noise-contaminated data tensor is generated

Noise level	Method	k	μ_k	Relative error	CPU time (secs)
10^{-3}	tGKT	4	3.80e-02	2.15e-03	1.56e+01
	G-tGKT	4	3.92e-02	2.32e-03	1.41e+01
10^{-2}	tGKT	2	7.19e-02	9.97e-03	4.47e+00
	G-tGKT	2	7.11e-02	1.00e-02	3.65e+00

Table 2: Results for Example 6.1.

by $\vec{\mathcal{B}} = \vec{\mathcal{B}}_{\text{true}} + \vec{\mathcal{E}}$ with the noise tensor $\vec{\mathcal{E}} \in \mathbb{R}^{256\times1\times256}$ defined as described above. The condition numbers of $\mathcal{A}^{(i)}$ satisfy $\texttt{cond}(\mathcal{A}^{(i)}) \geq 10^{16}$ for $i \geq 1$. Thus, every slice is numerically singular. We take $\eta = 1.1$ in Algorithm 5 and determine the regularization parameter by Newton's method.

Table 2 shows the computed regularization parameter values and relative errors for different noise levels, as well as the number of iterations required to satisfy the discrepancy principle by each method. Also timings are displayed. The tGKT method can be seen to be slightly slower but more accurate than the G-tGKT method.

Example 6.2. This example compares the tGKT$_p$, `nested_`tGKT$_p$, G-tGKT$_p$, and GG-tGKT method when applied with the regularization tensor $\mathcal{L}_2$ of Example 6.1. The tensor $\mathcal{A}$ is the same as in Example 6.1. The exact solution $\mathcal{X}_{\text{true}} \in \mathbb{R}^{256\times3\times256}$ is a tensor with all entries equal to one.

The noise-contaminated right-hand side $\mathcal{B} \in \mathbb{R}^{256\times3\times256}$ is generated by $\mathcal{B} = \mathcal{A}*\mathcal{X}_{\text{true}}+\mathcal{E}$, where the noise tensor $\mathcal{E} \in \mathbb{R}^{256\times3\times256}$ is determined according to (6.70). The regularization parameter or parameters are determined similarly as in Example 6.1. The computed regularization parameters

Noise level	Method	k	μ_k	Relative error	CPU time (secs)
10^{-3}	tGKT$_p$	-	-	2.15e-03	4.50e+01
	`nested_`tGKT$_p$	4	-	2.30e-03	1.99e+01
	G-tGKT$_p$	-	-	2.33e-03	4.22e+01
	GG-tGKT	4	3.91e-02	2.33e-03	1.97e+01
10^{-2}	tGKT$_p$	-	-	9.91e-03	1.17e+01
	`nested_`tGKT$_p$	2	-	1.28e-02	6.59e+00
	G-tGKT$_p$	-	-	9.97e-03	1.13e+01
	GG-tGKT	2	7.10e-02	9.97e-03	4.67e+00

Table 3: Results for Example 6.2.

and relative errors for different noise levels, as well as the number of iterations required to satisfy the discrepancy principle by each method, and timings are shown in Table 3.

Table 3 shows the GG-tGKT and `nested_`tGKT$_p$ methods to be the fastest for both noise levels, but the tGKT$_p$ and `nested_`tGKT$_p$ methods that do not involve flattening to yield approximate solutions of higher accuracy for $\delta = 10^{-3}$. In general, the tGKT$_p$ method yields the most accurate approximations of $\mathcal{X}_{\text{true}}$ and requires the most CPU time for both noise levels.

The remainder of this section discusses image and video restoration problems. We use the bisection method to determine the regularization parameter over a chosen interval. The blurring tensors $\mathcal{A}$ are constructed similarly as described by Kernfeld et al. [19] by using the function `blur` from [16].

Example 6.3. (Medical imaging) This example considers the restoration of an `MRI image` from MATLAB. We let $\mathcal{L} = \mathcal{L}_1 \in \mathbb{R}^{254\times256\times256}$ and generate the frontal slices $\mathcal{A}^{(i)}$ of $\mathcal{A}$ by using a modified form of the function `blur` with $N = 256$, $\sigma = 4$ and `band` $= 7$, i.e.,

$$\texttt{z} = [\texttt{exp}(-([\texttt{0} : \texttt{band} - \texttt{1}].^{2})/(2\sigma^2)), \texttt{zeros}(\texttt{1}, \texttt{N} - \texttt{band})], \tag{6.71}$$

$$A = \frac{1}{\sigma\sqrt{2\pi}}\texttt{toeplitz}([\texttt{z(1) fliplr(z(2 : end))}], \texttt{z}), \quad \mathcal{A}^{(i)} = A(i,1)A, \quad i = 1, 2, \ldots, 256.$$

This modification yields a nonsymmetric block circulant matrix with circulant blocks (BCCB). We fold the first block column of the matrix $A \otimes A$ to obtain $\mathcal{A}$. The condition numbers of $\mathcal{A}^{(i)}$ are $\texttt{cond}(\mathcal{A}^{(1)}) = \texttt{cond}(\mathcal{A}^{(251)}) = \cdots = \texttt{cond}(\mathcal{A}^{(256)}) \approx 13$. The condition numbers of the remaining slices are infinite. We determine the regularization parameter by the bisection method over the interval $[10^1, 10^9]$ using the discrepancy principle with $\eta = 1.01$.

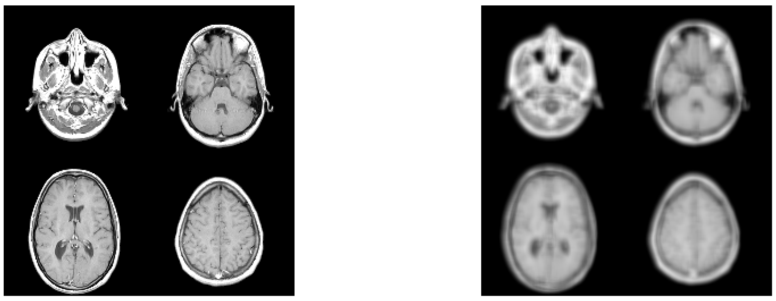
Figure 1: True image (left), blurred noisy image (right) for $\widetilde{\delta} = 10^{-3}$.

The frames 1, 8, 15 and 22 of a 3D MRI Data Set in MATLAB make up the true image `MRI image` shown on the left-hand side of Figure 1. The image is stored as a tensor column, $\vec{\mathcal{X}}_{\text{true}} \in \mathbb{R}^{256\times1\times256}$, using the `twist` operator and blurred by $\mathcal{A}$ defined above. The blurred and noisy image $\vec{\mathcal{B}}$ is shown in Figure 1 (right) using the `squeeze` operator. It is generated by $\vec{\mathcal{B}} = \mathcal{A} * \vec{\mathcal{X}}_{\text{true}} + \vec{\mathcal{E}}$ with $\vec{\mathcal{E}}$ representing "noise" determined as described above. The classical Golub-Kahan-Tikhonov (GKT) regularization method for the approximate solution of the minimization problem

$$\min_{x\in\mathbb{R}^{300^2}} \{\|(A \otimes A)x - b\|_2^2 + \mu^{-1}\|Lx\|_2^2\}$$

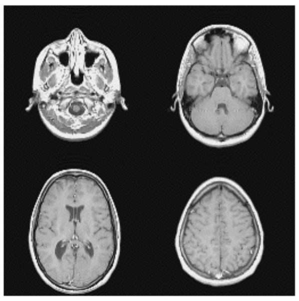
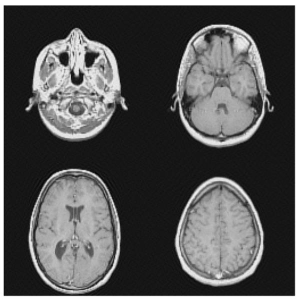

Figure 2: Reconstructed images by tGKT after 18 iterations (left) and GKT after 68 iterations (right) for $\widetilde{\delta} = 10^{-3}$.

is also considered, where $L = \mathcal{L}_1^{(1)} \in \mathbb{R}^{(300^2-2)\times 300^2}$ is defined in (3.36). For this problem, the blur- and noise-contaminated image is generated by

$$b = (A \otimes A)\texttt{unfold}(\vec{\mathcal{X}}_{\text{true}}) + \texttt{unfold}(\vec{\mathcal{E}}).$$

The restored images computed by the tGKT and GKT methods for the noise level $\widetilde{\delta} = 10^{-3}$ are shown in Figure 2 using the `squeeze` and MATLAB `reshape` operators, respectively. Table 4 shows the number of steps required to satisfy the discrepancy principle by each method, the regularization parameters, the relative errors, as well as the CPU time required. The GKT and G-tGKT methods involve flattening, require the most bidiagonalization steps, and produce restorations of the same or worse quality for both noise levels. Moreover, the G-tGKT method is the slowest. The tGKT method, which does not involve flattening, yields restorations of the best quality for both noise levels.

Noise level	Method	k	μ_k	Relative error	CPU time (secs)
10^{-3}	tGKT	18	1.65e+07	4.32e-02	2.53e+02
	G-tGKT	68	2.98e+07	5.78e-02	3.18e+03
	GKT	68	4.76e+06	5.78e-02	5.12e+01
10^{-2}	tGKT	7	1.82e+05	1.40e-01	4.40e+01
	G-tGKT	15	9.99e+04	1.47e-01	1.74e+02
	GKT	15	1.59e+04	1.47e-01	2.00e+00

Table 4: Results for Example 6.3.

Example 6.4. (Color image) This example illustrates image restoration with the regularization operator $\mathcal{L} = \mathcal{L}_1 \in \mathbb{R}^{298\times 300\times 300}$. The true image, shown on the left-hand side of Figure 3, is a `flower` image[1] of size $300 \times 300 \times 3$. This image is stored as a tensor $\mathcal{X}_{\text{true}} \in \mathbb{R}^{300\times 3\times 300}$ using the `multi_twist` operator and blurred by the tensor $\mathcal{A} \in \mathbb{R}^{300\times 300\times 300}$ that is generated by using the function `blur`. Specifically, we choose $N = 300$, $\sigma = 3$, and `band` $= 12$, and fold the first block column of the block circulant matrix with Toeplitz blocks (BCTB) matrix generated by (6.71) and

$$\texttt{zz} = [\texttt{z(1) fliplr(z(end} - \texttt{length(z)} + \texttt{2 : end))}], \quad A_1 = \frac{1}{\sqrt{2\pi\sigma}}\texttt{toeplitz(z,zz)},$$

$$A_2 = \frac{1}{\sqrt{2\pi\sigma}}\texttt{toeplitz(z)}, \;\; \mathcal{A}^{(i)} = A_1(i,1)A_2, \;\; i = 1, 2, \ldots, 300,$$

where A_1 is a circulant matrix, and A_2, a Toeplitz matrix.

The computed condition numbers for the tensor slices $\mathcal{A}^{(i)}$ are $\texttt{cond}(\mathcal{A}^{(1)}) = \cdots = \texttt{cond}(\mathcal{A}^{(12)}) = 7.6\cdot 10^8$. The slices $\mathcal{A}^{(i)}$ have infinite condition number for $i \geq 13$. We determine the regularization parameter by the bisection method over the interval $[10^{-3}, 10^5]$ using the discrepancy principle

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

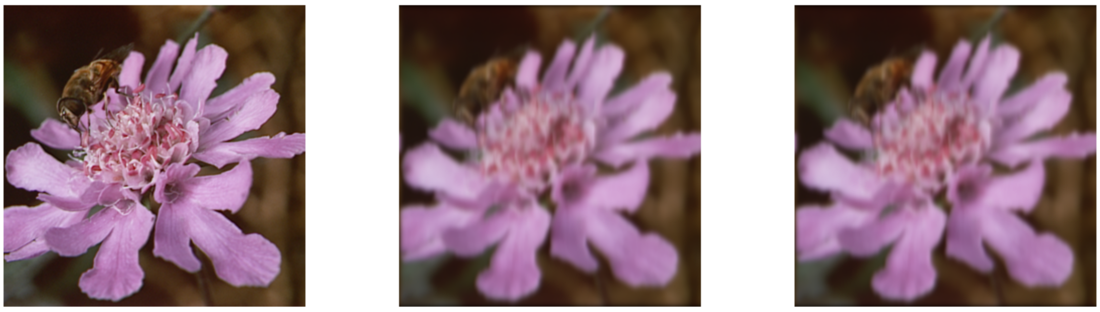

Figure 3: True image (left), middle - right: blurred and noisy images displayed for the problems (1.8) and (6.72), respectively, for $\widetilde{\delta} = 10^{-3}$.

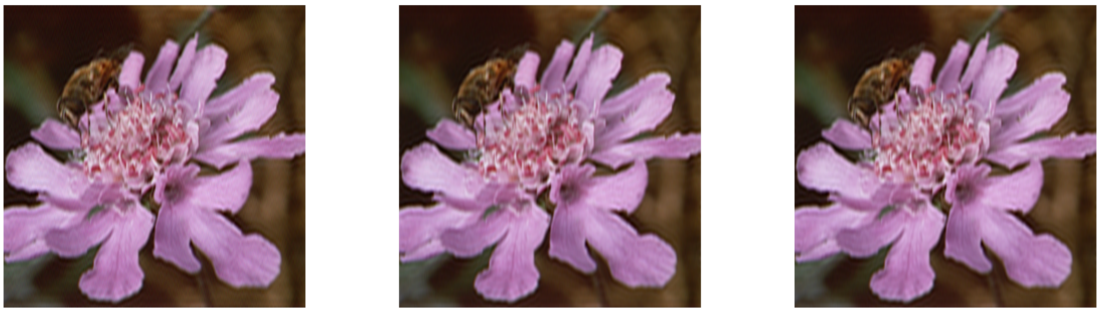

Figure 4: Reconstructed images by `nested_`tGKT$_p$ after 26 iterations (left), G-tGKT$_p$ (middle), and G-GKT method after 34 iterations (right) for $\widetilde{\delta} = 10^{-3}$.

with $\eta = 1.2$. The blurred and noisy image $\mathcal{B} \in \mathbb{R}^{300\times 3\times 300}$ is shown in Figure 3 (middle) using the `multi_squeeze` operator, and determined by $\mathcal{B} = \mathcal{A} * \mathcal{X}_{\text{true}} + \mathcal{E}$, where the error tensor $\mathcal{E} \in \mathbb{R}^{300\times 3\times 300}$ is generated according to (6.70).

We also consider the image deblurring problem

$$(A_1 \otimes A_2)X = B, \tag{6.72}$$

where the blur- and noise-contaminated available image generated by $B = (A_1 \otimes A_2)X_{\text{true}} + E$ is displayed on the right-hand side of Figure 3. The desired unavailable blur- and noise-free image $X_{\text{true}} \in \mathbb{R}^{300^2\times 3}$ is the matricized three-channeled image $\mathcal{X}_{\text{true}} \in \mathbb{R}^{300\times 300\times 3}$. The unknown noise in the matrix B is represented by $E \in \mathbb{R}^{300^2\times 3}$, which is the matricized "noise" tensor $\mathcal{E} \in \mathbb{R}^{300\times 300\times 3}$. The problem (6.72) is solved by the global Golub-Kahan-Tikhonov (G-GKT) regularization method described in [4]. Its implementation is analogous to the approach described in Section 5.1.

The restored images determined by the `nested_`tGKT$_p$, G-tGKT$_p$, and G-GKT methods are shown in Figure 4 for $\widetilde{\delta} = 10^{-3}$ using the `multi_squeeze` operator. The relative errors as well as CPU times are displayed in Table 5. We see that the tGKT$_p$ and `nested_`tGKT$_p$ methods yield restorations of the highest and the same quality for $\widetilde{\delta} = 10^{-3}$. Solution methods that involve flattening, such as the G-GKT, G-tGKT$_p$, and GG-tGKT methods, give restorations of the worst quality for $\widetilde{\delta} = 10^{-3}$ and require the most CPU time. Specifically, the G-GKT method is the fastest and yields restorations of the same quality as the G-tGKT$_p$ method, which is the slowest method for both noise levels. The GG-tGKT method requires more bidiagonalization steps than the `nested_`tGKT$_p$ method. Both methods produce restorations of the worst quality for $\widetilde{\delta} = 10^{-3}$ and $\widetilde{\delta} = 10^{-2}$, respectively.

Example 6.5. (Video restoration) This example considers the restoration of the first six consecutive frames of the `Xylophone` video from MATLAB with $\mathcal{L} = \mathcal{L}_2 \in \mathbb{R}^{239\times 240\times 240}$. Each video frame is in `MP4` format and has 240×240 pixels.

The first six blur- and noise-free frames are stored as a tensor $\mathcal{X}_{\text{true}} \in \mathbb{R}^{240\times 6\times 240}$ using the `multi_twist` operator and are blurred by the tensor $\mathcal{A} \in \mathbb{R}^{240\times 240\times 240}$, which is generated similarly

Noise level	Method	k	μ_k	Relative error	CPU time (secs)
10^{-3}	tGKT$_p$	-	-	6.11e-02	1.79e+03
	`nested_`tGKT$_p$	26	-	6.11e-02	8.72e+02
	G-tGKT$_p$	-	-	6.18e-02	3.61e+03
	GG-tGKT	34	4.64e+01	6.22e-02	1.98e+03
	G-GKT	34	6.47e+03	6.18e-02	1.41e+02
10^{-2}	tGKT$_p$	-	-	8.79e-02	9.58e+01
	`nested_`tGKT$_p$	5	-	1.02e-01	5.00e+01
	G-tGKT$_p$	-	-	8.69e-02	1.72e+02
	GG-tGKT	7	1.66e-01	8.74e-02	9.07e+01
	G-GKT	7	2.86e+02	8.48e-02	5.00e+00

Table 5: Results for Example 6.4.

as in Example 6.4 with

$$\mathcal{A}^{(i)} = \frac{1}{2\pi\sigma^2} A_2(i,1)A_2, \quad i = 1, 2, \ldots, 240, \quad N = 240, \quad \sigma = 2.5, \quad \texttt{band} = 12.$$

The condition numbers of the slices of $\mathcal{A}$ are $\texttt{cond}(\mathcal{A}^{(1)}) = \cdots = \texttt{cond}(\mathcal{A}^{(12)}) = 1.4 \cdot 10^7$ and the slices $\mathcal{A}^{(i)}$ have infinite condition number for $i \geq 13$. We determine the regularization parameter(s) by the bisection method over the interval $[10^1, 10^7]$ using the discrepancy principle with $\eta = 1.2$. The blurred and noisy frames are generated by $\mathcal{B} = \mathcal{A} * \mathcal{X}_{\text{true}} + \mathcal{E} \in \mathbb{R}^{240\times 6\times 240}$ with $\mathcal{E} \in \mathbb{R}^{240\times 6\times 240}$ defined by (6.70).

The true fourth frame is displayed in Figure 5 (left), and the blurred and noisy fourth frame is shown in Figure 5 (middle) using the `squeeze` operator. The restored images of the fourth frame determined by the G-tGKT$_p$, `nested_`tGKT$_p$, G-tGKT$_p$, and GG-tGKT methods are shown in Figure 5 (right) and Figure 6 using the `squeeze` operator. The relative errors and CPU times are displayed in Table 6. The methods G-tGKT$_p$ and G-tGKT$_p$, which work with a lateral slice of the data tensor at a time, are seen to yield the best quality restorations. Methods that involve flattening such as, the GG-tGKT and G-tGKT$_p$ are the slowest. The `nested_`tGKT$_p$ method yields the worst quality restorations but is the fastest.

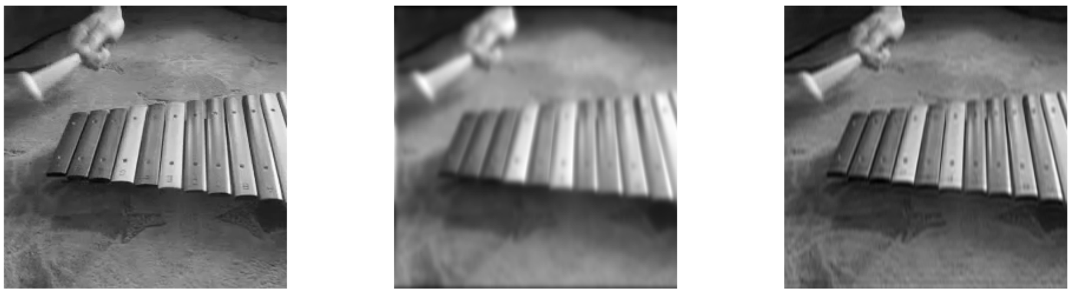

Figure 5: True image (left), blurred noisy image (middle), reconstructed image by the tGKT$_p$ method (right) for $\widetilde{\delta} = 10^{-3}$.

7 Conclusion

This paper presents the theoretical background and computational framework for the regularization of large-scale linear discrete ill-posed problem with a t-product structure, and discusses applications to (color) image and video restorations to illustrate the performance of the proposed methods. The quality of the restorations in the computed examples show the potential superiority of tensorizing over matricizing or vectorizing ill-posed problems for tensors. Solution methods that do not involve flattening are a good compromise for accuracy and speed. The interlacing property for the Frobenius norm of the singular tubes is discussed and applied. The regularization parameter(s) and the

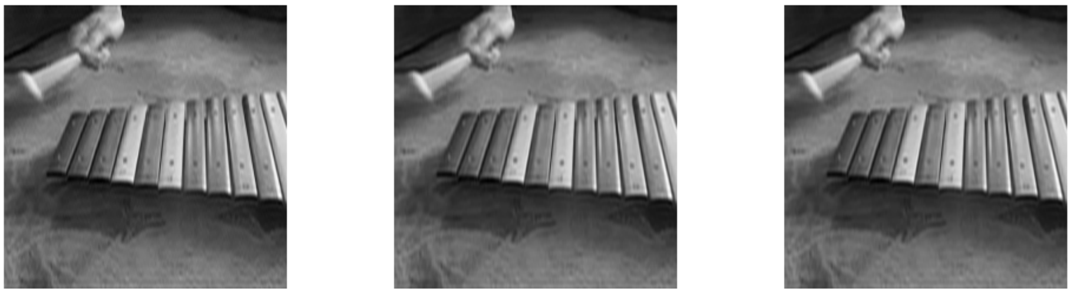
Figure 6: Reconstructed images by the `nested_`tGKT$_p$ method after 22 iterations (left), G-tGKT$_p$ (middle), and the GG-tGKT method (right) after 26 iterations for $\widetilde{\delta} = 10^{-3}$.

Noise level	Method	k	μ_k	Relative error	CPU time (secs)
10^{-3}	tGKT$_p$	-	-	4.26e-02	7.31e+02
	`nested_`tGKT$_p$	22	-	4.42e-02	2.76e+02
	G-tGKT$_p$	-	-	4.26e-02	1.70e+03
	GG-tGKT	26	1.50e+04	4.35e-02	8.46e+02
10^{-2}	tGKT$_p$	-	-	6.49e-02	5.07e+01
	`nested_`tGKT$_p$	4	-	1.10e-01	2.48e+01
	G-tGKT$_p$	-	-	6.69e-02	9.55e+01
	GG-tGKT	6	3.32e+02	6.70e-02	2.75e+01

Table 6: Results for Example 6.5.

required number of steps of Golub-Kahan-type bidiagonalization methods are determined by the discrepancy principle.

Acknowledgment

The authors are grateful to the referees for comments that led to improvements of the presentation. Research by LR was supported in part by NSF grant DMS-1720259.

References

[1] 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.

[2] 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.

[3] A. H. Bentbib, M. El Guide, K. Jbilou, E. Onunwor, and L. Reichel, Solution methods for linear discrete ill-posed problems for color image restoration, BIT Numer. Math., 58 (2018), pp. 555–578.

[4] A. H. Bentbib, M. El Guide, K. Jbilou, and L. Reichel, Global Golub-Kahan bidiagonalization applied to large discrete ill-posed problems, J. Comput. Appl. Math., 322 (2017), pp. 46–56.

[5] K. Braman, Third-order tensors as linear operators on a space of matrices, Linear Algebra Appl., 433 (2010), pp. 1241–1253.

[6] A. Buccini, M. Pasha, and L. Reichel, Generalized singular value decomposition with iterated Tikhonov regularization, J. Comput. Appl. Math., 373 (2020), Art. 112276.

[7] D. Calvetti and L. Reichel, Tikhonov regularization of large linear problems, BIT Numer. Math., 43 (2003), pp. 263–283.

[8] M. Donatelli and L. Reichel, Square smoothing regularization matrices with accurate boundary conditions, J. Comput. Appl. Math., 272 (2014), pp. 334–349.

[9] L. Dykes, G. Huang, S. Noschese, and L. Reichel, Regularization matrices for discrete ill-posed problems in several space-dimensions, Numer. Linear Algebra Appl., 25 (2018), Art. e2163.

[10] 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/2005.07458.pdf

[11] M. El Guide, A. El Ichi, K. Jbilou, and R. Sadaka, On GMRES and Golub-Kahan methods via the T-product for color image processing, Electron. J. Linear Algebra, 37 (2021), pp. 524–543.

[12] A. El Ichi, K. Jbilou, and R. Sadaka, Tensor global extrapolation methods using the n-mode and the Einstein products, Mathematics, 8(8) (2020), Art. 1298.

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

[14] 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.

[15] P. C. Hansen, Rank-Deficient and Discrete Ill-Posed Problems, SIAM, Philadelphia, 1998.

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

[17] N. Hao, M. E. Kilmer, K. Braman, and R. C. Hoover, Facial recognition using tensor-tensor decompositions, SIAM J. Imaging Sci., 6 (2013), pp. 437–463.

[18] 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.

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

[20] M. Kilmer, K. Braman, and N. Hao, Third order tensors as operators on matrices: A theoretical and computational framework, Technical Report, Tufts University, Department of Computer Science, 2011.

[21] 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.

[22] M. E. Kilmer and C. D. Martin, Factorization strategies for third-order tensors, Linear Algebra Appl., 435 (2011), pp. 641–658.

[23] S. Kindermann, Convergence analysis of minimization-based noise level-free parameter choice rules for linear ill-posed problems, Electron. Trans. Numer. Anal., 38 (2011), pp. 233–257.

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

[25] W. Liu, and X. Jin, A study on T-eigenvalues of third-order tensors, Linear Algebra and its Applications, 612 (2021), pp. 357–374.

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

[27] C. D. Martin, R. Shafer, and B. LaRue, An order-p tensor factorization with applications in imaging, SIAM J. Sci. Comput., 35 (2013), pp. A474–A490.

[28] E. Newman, M. Kilmer, and L. Horesh, Image classification using local tensor singular value decompositions, 2017 IEEE 7th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP), Curacao, 2017, pp. 1-5, doi: 10.1109/CAMSAP.2017.8313137.

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

[30] L. Reichel and A. Shyshkov, A new zero-finder for Tikhonov regularization, BIT Numer. Math., 48 (2008), pp. 627–643.

[31] S. Soltani, M. E. Kilmer, and P. C. Hansen, A tensor-based dictionary learning approach to tomographic image reconstruction, BIT Numer. Math., 56 (2016), pp. 1425–1454.

[32] Z. Zhang, G. Ely, S. Aeron, N. Hao, and M. E. Kilmer, Novel factorization strategies for higher order tensors: Implications for compression and recovery of multi-linear data, arXiv preprint, 2013, https://arxiv.org/pdf/1307.0805.pdf