Weighted tensor Golub-Kahan-Tikhonov-type methods applied to image processing using a t-product

Dedicated to Ken Hayami on the occasion of his retirement.

Lothar Reichel* and Ugochukwu O. Ugwu†

Department of Mathematical Sciences, Kent State University, OH 44242

June 25, 2022

Abstract

This paper discusses weighted tensor Golub-Kahan-type bidiagonalization processes using the t-product. This product was introduced in [M. E. Kilmer and C. D. Martin, Factorization strategies for third order tensors, Linear Algebra Appl., 435 (2011), pp. 641–658]. A few steps of a bidiagonalization process with a weighted least squares norm are carried out to reduce a large-scale linear discrete ill-posed problem to a problem of small size. The weights are determined by symmetric positive definite (SPD) tensors. Tikhonov regularization is applied to the reduced problem. An algorithm for tensor Cholesky factorization of SPD tensors is presented. The data is a laterally oriented matrix or a general third order tensor. The use of a weighted Frobenius norm in the fidelity term of Tikhonov minimization problems is appropriate when the noise in the data has a known covariance matrix that is not the identity. We use the discrepancy principle to determine both the regularization parameter in Tikhonov regularization and the number of bidiagonalization steps. Applications to image and video restoration are considered.

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

1 Introduction

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

$$\min_{\vec{\mathcal{X}}\in\mathbb{R}^{m\times 1\times n}} \|\mathcal{A}*\vec{\mathcal{X}}-\vec{\mathcal{B}}\|_{\mathcal{M}^{-1}}^2, \quad \mathcal{A}\in\mathbb{R}^{\ell\times m\times n}, \quad \vec{\mathcal{B}}\in\mathbb{R}^{\ell\times 1\times n}, \tag{1.1}$$

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, $\vec{\mathcal{X}}$ and $\vec{\mathcal{B}}$ are laterally oriented $m\times n$ and $\ell\times n$ matrices, respectively, and $\|\cdot\|_{\mathcal{M}^{-1}}$ denotes a weighted Frobenius norm; see below. The tensor $\mathcal{A}$ specifies the model and the tensor $\vec{\mathcal{B}}$ represents measured data that is contaminated by an error $\vec{\mathcal{E}}$, i.e.,

$$\vec{\mathcal{B}}=\vec{\mathcal{B}}_{\text{true}}+\vec{\mathcal{E}}, \tag{1.2}$$

where $\vec{\mathcal{B}}_{\text{true}}\in\mathbb{R}^{\ell\times 1\times n}$ denotes the unknown error-free data tensor associated with $\vec{\mathcal{B}}$. The use of a weighted Frobenius norm is appropriate when the error $\vec{\mathcal{E}}$ is not white Gaussian.

We consider minimization problems (1.1) for which the Frobenius norm of the singular tubes of $\mathcal{A}$, which are analogues of the singular values of a matrix, decay rapidly to zero, and there are many singular tubes of tiny Frobenius norm of different orders of magnitude; see [19, 27] for examples of such problems. This makes the solution of (1.1) a linear discrete ill-posed problem. The operator $*$ denotes the t-product introduced by Kilmer and Martin [19]; see Section 2 for details.

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

† e-mail: uugwu@kent.edu

We will 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 its unique solution of minimal Frobenius norm. We would like to determine an accurate approximation of $\vec{\mathcal{X}}_{\text{true}}$ given $\mathcal{A}$ and $\vec{\mathcal{B}}$.

Problems of the form (1.1) arise, e.g., in image deblurring, see, e.g., [18, 26, 27, 28], where the goal is to recover an accurate approximation of the unavailable exact image $\vec{\mathcal{X}}_{\text{true}}$ by removing blur and noise from an available degraded image that is represented by $\vec{\mathcal{B}}$. The tensor $\mathcal{A}$ represents a blurring operator and typically is ill-conditioned. The ill-conditioning of $\mathcal{A}$, and the presence of the error $\vec{\mathcal{E}}$ in $\vec{\mathcal{B}}$ make it difficult to solve (1.1). In particular, straightforward solution of (1.1) typically gives a meaningless restoration due to a large propagated error, which stems from the error $\vec{\mathcal{E}}$ in $\vec{\mathcal{B}}$. Regularization, such as by Tikhonov's method, is required to ensure that the computed approximate solution of (1.1) is useful. Regularization techniques for (1.1) when the norm is the standard Frobenius norm have received some attention in the literature; see [12, 19, 26, 27, 28]. Here we consider replacing the the minimization problem (1.1) by a penalized weighted least squares problem of the form

$$\min_{\vec{\mathcal{X}}\in\mathbb{R}^{m\times 1\times n}} \left\{ \|\mathcal{A} * \vec{\mathcal{X}} - \vec{\mathcal{B}}\|^2_{\mathcal{M}^{-1}} + \mu^{-1}\|\vec{\mathcal{X}}\|^2_{\mathcal{L}^{-1}} \right\}, \tag{1.3}$$

where the tensors $\mathcal{M} \in \mathbb{R}^{\ell\times\ell\times n}$ and $\mathcal{L} \in \mathbb{R}^{m\times m\times n}$ are symmetric positive definite (SPD) with inverses $\mathcal{M}^{-1}$ and $\mathcal{L}^{-1}$, that determine weighted Frobenius norms; see below for a definition. The notion of positive definiteness for tensors under the t-product formalism is discussed by Beik et al. [7]. The first term in (1.3) is commonly referred to as the *fidelity term* and the second term as the *regularization term*.

The Tikhonov regularization problem (1.3) is said to be in *standard form* when $\mathcal{M}$ and $\mathcal{L}$ are equal to the identity tensor, which we denote by $\mathcal{I}$. It is well known how large-scale problems in standard form can be reduced to small problems by using Arnoldi-type or bidiagonalization-type processes; see [12, 26, 27, 28]. Tikhonov minimization problems (1.3) with $\mathcal{L}$ or $\mathcal{M}$ different from $\mathcal{I}$ are said to be in *general form*. The choice of the norm in the fidelity term should depend on properties of the error $\vec{\mathcal{E}}$ and the choice of the norm in the regularization term should depend on properties of the desired solution $\mathcal{X}_{\text{true}}$. We will discuss these choices in Section 5. Other tensor-based solution methods, that do not use the t-product, for minimization problems (1.3) in standard form are discussed in [5, 6, 11, 13].

Throughout this paper, $\|\vec{\mathcal{X}}\|_{\mathcal{N}}$ denotes the Frobenius norm of $\vec{\mathcal{X}}$ induced by an SPD tensor $\mathcal{N} \in \mathbb{R}^{m\times m\times n}$. We will refer to this norm as the $\mathcal{N}$-norm of a lateral slice $\vec{\mathcal{X}}$; see Section 2 for the definition of the $\mathcal{N}$-norm of a general third order tensor $\mathcal{X} \in \mathbb{R}^{m\times p\times n}$, $p > 1$. The $\mathcal{N}$-norm of $\vec{\mathcal{X}} \in \mathbb{R}^{m\times 1\times n}$ is defined as

$$\|\vec{\mathcal{X}}\|_{\mathcal{N}} = \sqrt{(\vec{\mathcal{X}}^T * \mathcal{N} * \vec{\mathcal{X}})_{(:,:,1)}}.$$

Thus, this norm is the square root of the $(1,1,1)$th entry of $\vec{\mathcal{X}}^T * \mathcal{N} * \vec{\mathcal{X}} \in \mathbb{R}^{1\times 1\times n}$, where the superscript ^{T} denotes transposition (defined below).

The quantity $\mu > 0$ in (1.3) denotes the regularization parameter. It decides the relative influence on the solution of (1.3) of the fidelity and the regularization terms. In the present paper, we determine μ by the discrepancy principle; see, e.g., [14] for a description and analysis of this principle. Let a bound

$$\|\vec{\mathcal{E}}\|_{\mathcal{M}^{-1}} \leq \delta \tag{1.4}$$

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

$$\|\mathcal{A} * \vec{\mathcal{X}}_\mu - \vec{\mathcal{B}}\|_{\mathcal{M}^{-1}} = \eta\delta, \tag{1.5}$$

where $\eta > 1$ is a user-specified constant that is independent of δ. Our reason for using the regularization parameter $1/\mu$ instead of μ in (1.3) will be commented on in Section 3. Other techniques, such as generalized cross validation and the L-curve criterion also can be used to determine the regularization parameter; see, e.g., [15, 17, 20, 21, 25].

We remark that minimization problems of the form (1.3) also arise in uncertainty quantification (UQ) in large-scale Bayesian linear inverse problems; see, e.g., [9, 10, 24], where $\mathcal{L}$ and $\mathcal{M}$ are

suitably defined *covariance tensors*. The techniques developed in this work can be applied to the solution of inverse UQ problems.

It is the purpose of this paper to discuss applications in image and video restoration. We consider the situation when $\mathcal{M} \neq \mathcal{I}$. Arridge et al. [3] observed that for large-scale problems in three space-dimensions, the computation of the Cholesky factorization of $\mathcal{L}$ may be prohibitively expensive. We therefore derive a solution method that does not require the Cholesky factorization of $\mathcal{L}$.

The normal equations associated with (1.3) are given by

$$(\mathcal{A}^T * \mathcal{M}^{-1} * \mathcal{A} + \mu^{-1}\mathcal{L}^{-1}) * \vec{\mathcal{X}} = \mathcal{A}^T * \mathcal{M}^{-1} * \vec{\mathcal{B}}, \tag{1.6}$$

and can be derived analogously as when $\mathcal{L}$ and $\mathcal{M}$ are the identity tensor $\mathcal{I}$; see [27] for this situation. Since the tensor $\mathcal{L}$ is SPD, equation (1.6) has a unique solution $\mathcal{X}_\mu$ for any $\mu > 0$.

The change of variables $\vec{\mathcal{X}} = \mathcal{L} * \vec{\mathcal{Y}}$ in (1.6) yields the equation

$$(\mathcal{A}^T * \mathcal{M}^{-1} * \mathcal{A} * \mathcal{L} + \mu^{-1}\mathcal{I}) * \vec{\mathcal{Y}} = \mathcal{A}^T * \mathcal{M}^{-1} * \vec{\mathcal{B}}.$$

Its solution $\vec{\mathcal{Y}}_\mu$ solves the minimization problem

$$\min_{\vec{\mathcal{Y}} \in \mathbb{R}^{m\times 1\times n}} \{\|\mathcal{A} * \mathcal{L} * \vec{\mathcal{Y}} - \vec{\mathcal{B}}\|^2_{\mathcal{M}^{-1}} + \mu^{-1}\|\vec{\mathcal{Y}}\|^2_{\mathcal{L}}\}. \tag{1.7}$$

The solution of (1.6) can be computed as $\vec{\mathcal{X}}_\mu = \mathcal{L} * \vec{\mathcal{Y}}_\mu$.

We describe two weighted t-product Golub-Kahan bidiagonalization-type processes for the approximate solution of (1.7), and thereby of (1.3). They generate orthonormal tensor bases for the k-dimensional tensor Krylov (t-Krylov) subspaces

$$\mathbb{K}_k(\mathcal{A}^T * \mathcal{M}^{-1} * \mathcal{A} * \mathcal{L}, \mathcal{A}^T * \mathcal{M}^{-1} * \vec{\mathcal{B}}) \quad \text{and} \quad \mathbb{K}_k(\mathcal{A} * \mathcal{L} * \mathcal{A}^T * \mathcal{M}^{-1}, \vec{\mathcal{B}}), \tag{1.8}$$

where $\mathbb{K}_k(\mathcal{A}, \vec{\mathcal{B}}) := \text{span}\{\vec{\mathcal{B}}, \mathcal{A} * \vec{\mathcal{B}}, \ldots, \mathcal{A}^{k-1} * \vec{\mathcal{B}}\}$. Each step of these processes requires two tensor-matrix product evaluations, one with $\mathcal{A}$ and one with $\mathcal{A}^T$. Application of a few $k \ll m$ steps of each bidiagonalization process to $\mathcal{A}$ reduces the large-scale problem (1.7) to a problem of small size. Specifically, k steps of the weighted t-product Golub-Kahan bidiagonalization (W-tGKB) process applied to $\mathcal{A}$ reduces $\mathcal{A}$ to a small $(k+1) \times k \times n$ lower bidiagonal tensor, while the weighted global tGKB (WG-tGKB) process reduces $\mathcal{A}$ to a small $(k+1) \times k$ lower bidiagonal matrix. The WG-tGKB process differs from the W-tGKB process in the choice of inner product and involves flattening since it reduces equation (1.7) to a problem that involves a matrix and vectors. Both the W-tGKB and WG-tGKB processes extend the generalized Golub-Kahan bidiagonalization process described in [9] for matrices to third order tensors. We refer to the solution methods based on the W-tGKB and WG-tGKB processes as the weighted t-product Golub-Kahan-Tikhonov (W-tGKT) and the weighted global t-product Golub-Kahan-Tikhonov (WG-tGKT) methods, respectively.

We also consider analogues of the minimization problems (1.3) and (1.7) in which the tensors $\vec{\mathcal{X}} \in \mathbb{R}^{m\times 1\times n}$ and $\vec{\mathcal{B}} \in \mathbb{R}^{\ell\times 1\times n}$ are replaced by general third order tensors $\mathcal{X} \in \mathbb{R}^{m\times p\times n}$ and $\mathcal{B} \in \mathbb{R}^{\ell\times p\times n}$, respectively, for some $p > 1$. That is, we consider minimization problems of the form

$$\min_{\mathcal{X} \in \mathbb{R}^{m\times p\times n}} \|\mathcal{A} * \mathcal{X} - \mathcal{B}\|^2_{\mathcal{M}^{-1}}. \tag{1.9}$$

Problems of this kind arise when restoring a color image or a sequence of consecutive gray-scale video frames. For instance, a color image restoration problem with RGB channels corresponds to $p = 3$, whereas when restoring a gray-scale video, p is the number of consecutive video frames. The minimization problems analogous to (1.3) and (1.7) are given by

$$\min_{\mathcal{X} \in \mathbb{R}^{m\times p\times n}} \{\|\mathcal{A} * \mathcal{X} - \mathcal{B}\|^2_{\mathcal{M}^{-1}} + \mu^{-1}\|\mathcal{X}\|^2_{\mathcal{L}^{-1}}\} \tag{1.10}$$

and

$$\min_{\mathcal{Y} \in \mathbb{R}^{m\times p\times n}} \{\|\mathcal{A} * \mathcal{L} * \mathcal{Y} - \mathcal{B}\|^2_{\mathcal{M}^{-1}} + \mu^{-1}\|\mathcal{Y}\|^2_{\mathcal{L}}\}, \tag{1.11}$$

respectively.

We present three solution methods for (1.10). The first two work with each lateral slice $\vec{\mathcal{B}}_j$, $j = 1, 2, \ldots, p$, of $\mathcal{B}$ independently, i.e., they consider (1.11) as p separate minimization problems and apply the W-tGKT or WG-tGKT methods to each one of these problems. The resulting methods for (1.10) are referred to as the W-tGKT$_p$ and WG-tGKT$_p$ methods, respectively, and sometimes simply as p-methods. The third method works with the lateral slices of $\mathcal{B}$ simultaneously and will be referred to as the weighted generalized global tGKT (WGG-tGKT) method. This method uses the weighted generalized global tGKB (WGG-tGKB) process introduced in Section 4 to reduce (1.11) to a problem of small size. The WGG-tGKT method requires less CPU time than the p-methods because it uses larger chunks of data at a time than the other methods in our comparison.

Numerical experiments in [26, 27, 28] and in Section 5 show that the p-methods often yield restorations of higher quality than the WGG-tGKT method. This behavior may be attributed to the facts that the p-methods determine p regularization parameters, and generally use p different t-Krylov subspaces (1.8). The WGG-tGKT and WG-tGKT$_p$ methods involve flattening and require additional product definition to the t-product. They are related to the global tensor Krylov subspace methods described in [5, 6, 11, 12, 13, 26, 27, 28].

The organization of this paper is as follows. Section 2 introduces notation and preliminaries associated with the t-product, while Section 3 discusses the W-tGKT and W-tGKT$_p$ methods as well as the W-tGKB process. Section 4 presents the weighted global t-Krylov subspace methods. The WGG-tGKT method and WGG-tGKB process are described in Section 4.1. Section 4.2 presents the WG-tGKB process, and the WG-tGKT and WG-tGKT$_p$ methods. In the computed examples in Section 5, we take $\mathcal{M} \neq \mathcal{I}$ and discuss the performance of these methods when $\mathcal{L} \neq \mathcal{I}$ and $\mathcal{L} = \mathcal{I}$. Concluding remarks are presented in Section 6.

2 Notation and Preliminaries

This section reviews and modifies results by El Guide et al. [12] and Kilmer et al. [18] to suit the current framework. In this paper, a tensor is a multidimensional array of real numbers. We denote third order tensors by calligraphic script letters, e.g., $\mathcal{A}$, matrices by capital letters, e.g., A, vectors by lower case letters, e.g., a, and tubal scalars (tubes) by boldface letters, e.g., $\mathbf{a}$. Tubes of third order tensors are obtained by fixing any two of the indices, while slices are obtained by keeping one of the indices fixed; see Kolda and Bader [22]. The jth lateral slice (also referred to as the jth tensor column) is a laterally oriented matrix denoted by $\vec{\mathcal{A}}_j$. The kth frontal slice is denoted by $\mathcal{A}^{(k)}$ and is a matrix.

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 $\texttt{unfold}(\mathcal{A})$ returns an $\ell n \times m$ matrix with the frontal slices, whereas the $\texttt{fold}$ operator folds back the unfolded tensor $\mathcal{A}$, i.e.,

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

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,

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

Definition 2.1. (t-product [19]) 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

$$\mathcal{C} = \texttt{fold}(\texttt{bcirc}(\mathcal{A}) \cdot \texttt{unfold}(\mathcal{B})), \tag{2.12}$$

where $\cdot$ stands for the standard matrix-matrix product.

The block circulant matrix $\texttt{bcirc}(\mathcal{A})$ can be block diagonalized by the discrete Fourier transform (DFT) 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

$$\bar{A} := \texttt{blockdiag}(\widehat{A}^{(1)}, \widehat{A}^{(2)}, \dots, \widehat{A}^{(n)}) = (F_n \otimes I_\ell) \cdot \texttt{bcirc}(\mathcal{A}) \cdot (F_n^* \otimes I_m), \tag{2.13}$$

where $\otimes$ stands for Kronecker product and F_n^* denotes the conjugate transpose of F_n. 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 [19] 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., $\widehat{\cdot}$.

Using (2.13), the t-product (2.12) can be evaluated as

$$\mathcal{A} * \mathcal{B} = \texttt{fold}\Big((F_n^* \otimes I_\ell) \cdot \big((F_n \otimes I_\ell) \cdot \texttt{bcirc}(\mathcal{A}) \cdot (F_n^* \otimes I_m)\big) \cdot (F_n \otimes I_m) \cdot \texttt{unfold}(\mathcal{B})\Big). \tag{2.14}$$

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 $(F_n \otimes I_m) \cdot \texttt{unfold}(\mathcal{B})$ in $\mathcal{O}(pmn \log_2(n))$ arithmetic floating point operations (flops); see Kilmer and Martin [19] for details.

The computations (2.14) 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 and let $\widehat{\mathcal{C}}(:, :, i)$ denote the ith frontal slice of $\widehat{\mathcal{C}}$. Then the t-product (2.14) 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 $\widehat{\mathcal{B}}$, i.e.,

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

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

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 [19]. The tensor transpose is analogous to the matrix transpose. Specifically, 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 $\mathcal{I} \in \mathbb{R}^{m \times m \times n}$ is a tensor whose first frontal slice, $\mathcal{I}^{(1)}$, is the $m \times m$ identity matrix, and all other frontal slices, $\mathcal{I}^{(i)}$, $i = 2, 3, \dots, n$, are zero matrices; see [19].

The notion of orthogonality under the t-product formalism is well defined and similar to the matrix case; see Kilmer and Martin [19]. A tensor $\mathcal{Q} \in \mathbb{R}^{m \times m \times n}$ is said to be orthogonal if $\mathcal{Q}^T * \mathcal{Q} = \mathcal{Q} * \mathcal{Q}^T = \mathcal{I}$. The lateral slices of $\mathcal{Q}$ are orthonormal if they satisfy

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

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 zero. Given the tensor $\mathcal{A}$ and an orthogonal tensor $\mathcal{Q}$ of compatible dimension, we have

$$\|\mathcal{Q} * \mathcal{A}\|_F = \|\mathcal{A}\|_F;$$

see [19] for a proof. 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 [19].

A tensor $\mathcal{A} \in \mathbb{R}^{m\times m\times n}$ has 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 [19].

Let $\mathcal{A} \in \mathbb{R}^{\ell\times m\times n}$, $\ell \geq m$. Then the tensor singular value decomposition (tSVD), introduced by Kilmer and Martin [19], is given by

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

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

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

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

$$\|\mathbf{s}_1\|_F \geq \|\mathbf{s}_2\|_F \geq \cdots \geq \|\mathbf{s}_m\|_F.$$

The tubal rank of $\mathcal{A}$ is the number of nonzero singular tubes of $\mathcal{A}$; see Kilmer et al. [18].

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

$$\|\vec{\mathcal{X}}\|_F = \sqrt{\left(\vec{\mathcal{X}}^T * \vec{\mathcal{X}}\right)_{(:,:,1)}}; \tag{2.15}$$

see [18]. Thus, the square of the Frobenius norm of $\vec{\mathcal{X}}$ is the $(1,1,1)$th entry of the tubal scalar $\vec{\mathcal{X}}^T * \vec{\mathcal{X}} \in \mathbb{R}^{1\times 1\times n}$, which is denoted by $\left(\vec{\mathcal{X}}^T * \vec{\mathcal{X}}\right)_{(:,:,1)}$.

A tensor $\mathcal{M} \in \mathbb{R}^{m\times m\times n}$ is said to be positive definite (PD) if it satisfies

$$\left(\vec{\mathcal{X}}^T * \mathcal{M} * \vec{\mathcal{X}}\right)_{(:,:,1)} > 0$$

for all nonzero lateral slices $\vec{\mathcal{X}} \in \mathbb{R}^{m\times 1\times n}$; see [7]. Moreover, $\mathcal{M}$ is symmetric if each frontal slice $\widehat{\mathcal{M}}^{(i)}$, $i = 1, 2, \ldots, n$, of $\widehat{\mathcal{M}}$ is Hermitian. Hence, a tensor $\mathcal{M}$ is SPD if $\widehat{\mathcal{M}}^{(i)}$ is Hermitian PD; see [18].

Let $\mathcal{M} \in \mathbb{R}^{m\times m\times n}$ be an SPD tensor. The $\mathcal{M}$-norm of a general third order tensor $\mathcal{X} \in \mathbb{R}^{m\times p\times n}$ with $p > 1$ is given by

$$\|\mathcal{X}\|_{\mathcal{M}} = \sqrt{\texttt{trace}\Big(\left(\mathcal{X}^T * \mathcal{M} * \mathcal{X}\right)_{(:,:,1)}\Big)}.$$

The quantity $\left(\mathcal{X}^T * \mathcal{M} * \mathcal{X}\right)_{(:,:,1)}$ is a $p \times p$ matrix. It represents the first frontal slice of the tensor $\mathcal{X}^T * \mathcal{M} * \mathcal{X} \in \mathbb{R}^{p\times p\times n}$, and $\texttt{trace}(M)$ denotes the trace of the matrix M.

Algorithm 1 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

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

The tubal scalar $\mathbf{a}$ is not necessarily invertible; see [18]. We remark that $\mathbf{a}$ is invertible only 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 face of the $1 \times 1 \times n$ tubal scalar $\mathbf{a}$, while $\vec{\mathcal{V}}^{(j)}$ is a vector with m entries, and the jth face of $\vec{\mathcal{V}} \in \mathbb{R}^{m\times 1\times n}$. Steps 3 and 7 of Algorithm 1 use the fact that for an SPD matrix $M \in \mathbb{R}^{m\times m}$ and $x \in \mathbb{R}^m$, we have

$$\|x\|_M = \sqrt{x^T M x}.$$

An analogue of Algorithm 1 for $\mathcal{M} = \mathcal{I}$ is described in [18].

Algorithm 1: Normalize($\vec{\mathcal{X}}, \mathcal{M}$)

Input: $\vec{\mathcal{X}} \in \mathbb{R}^{m\times 1\times n} \neq \vec{\mathcal{O}}$, $\mathcal{M}$ is an $m \times m \times n$ SPD tensor
Output: $\vec{\mathcal{V}}$, $\mathbf{a}$ with $\vec{\mathcal{X}} = \vec{\mathcal{V}} * \mathbf{a}$ and $\|\vec{\mathcal{V}}\|_{\mathcal{M}} = 1$

$\vec{\mathcal{V}} \leftarrow \texttt{fft}(\vec{\mathcal{X}}, [\,], 3)$, $\mathcal{M} \leftarrow \texttt{fft}(\mathcal{M}, [\,], 3)$
for $j = 1, 2, \ldots, n$ **do**
  $\mathbf{a}^{(j)} \leftarrow \|\vec{\mathcal{V}}^{(j)}\|_{\mathcal{M}^{(i)}}$ ($\vec{\mathcal{V}}^{(j)}$ is a vector)
  if $\mathbf{a}^{(j)} > \texttt{tol}$ **then**
    $\vec{\mathcal{V}}^{(j)} \leftarrow \frac{1}{\mathbf{a}^{(j)}}\vec{\mathcal{V}}^{(j)}$
  else
    $\vec{\mathcal{V}}^{(j)} \leftarrow \texttt{randn}(m, 1)$; $\mathbf{a}^{(j)} \leftarrow \|\vec{\mathcal{V}}^{(j)}\|_{\mathcal{M}^{(i)}}$; $\vec{\mathcal{V}}^{(j)} \leftarrow \frac{1}{\mathbf{a}^{(j)}}\vec{\mathcal{V}}^{(j)}$; $\mathbf{a}^{(j)} \leftarrow 0$
  end
end
$\vec{\mathcal{V}} \leftarrow \texttt{ifft}(\vec{\mathcal{V}}, [\,], 3)$; $\mathbf{a} \leftarrow \texttt{ifft}(\mathbf{a}, [\,], 3)$

Algorithm 2 determines the tensor Cholesky decomposition of an SPD tensor $\mathcal{M}$. This algorithm is mimetic of the Cholesky factorization of a matrix M. Step 3 of Algorithm 2 carries out the Cholesky factorization of the Hermitian matrix $\widehat{\mathcal{M}}^{(i)}$ in the Fourier domain. This algorithm will be used in Section 5.

Algorithm 2: Tensor Cholesky factorization

Input: SPD tensor $\mathcal{M} \in \mathbb{R}^{m\times m\times n}$
Output: Tensor Cholesky factor $\mathcal{R} \in \mathbb{R}^{m\times m\times n}$, where $\mathcal{M} = \mathcal{R}^T * \mathcal{R}$

$\widehat{\mathcal{M}} = \texttt{fft}(\mathcal{M}, [\,], 3)$
for $i = 1$ **to** n **do**
  $\widehat{\mathcal{R}}^{(i)} \leftarrow \texttt{chol}(\widehat{\mathcal{M}}^{(i)})$, where `chol` denotes MATLAB's Cholesky factorization operator. Thus, $\widehat{\mathcal{R}}^{(i)}$ is an upper triangular complex matrix.
end
$\mathcal{R} \leftarrow \texttt{ifft}(\widehat{\mathcal{R}}, [\,], 3)$

Introduce the tensors

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

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 $y = [y_1, y_2, \ldots, y_k]^T \in \mathbb{R}^k$. El Guide et al. [12] define two products denoted by $\circledast$:

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

We conclude this section by modifying some notions introduced by El Guide et al. [12]. Let $\mathcal{M} = [m_{iik}] \in \mathbb{R}^{m\times m\times n}$ and 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

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

Then

$$\langle \mathcal{C}, \mathcal{D}\rangle_{\mathcal{M}} = \sum_{i=1}^{m}\sum_{j=1}^{p}\sum_{k=1}^{n} c_{ijk}[\mathcal{M} * \mathcal{D}]_{ijk}, \qquad \langle \vec{\mathcal{C}}, \vec{\mathcal{D}}\rangle_{\mathcal{M}} = \sum_{i=1}^{m}\sum_{k=1}^{p} c_{i1k}[\mathcal{M} * \vec{\mathcal{D}}]_{i1k}.$$

Specifically, $\langle \mathcal{C}, \mathcal{D}\rangle_{\mathcal{M}} = \langle \mathcal{C}, \mathcal{M} * \mathcal{D}\rangle$, and $\langle \vec{\mathcal{C}}, \vec{\mathcal{D}}\rangle_{\mathcal{M}} = \langle \vec{\mathcal{C}}, \mathcal{M} * \vec{\mathcal{D}}\rangle$. Let

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

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 $\mathcal{B}_j \in \mathbb{R}^{\ell\times s\times n}$, $\vec{\mathcal{B}}_j \in \mathbb{R}^{\ell\times 1\times n}$ $j = 1, 2, \ldots, p$. The weighted T-diamond products $\mathbb{A}^T \lozenge_{\mathcal{M}} \mathbb{B}$ and $\mathcal{A}^T \lozenge_{\mathcal{M}} \mathcal{B}$ yield $m \times p$ matrices with entries

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

The weighted T-diamond product, $\lozenge_{\mathcal{M}}$, is the weighted analogue of the T-diamond product, $\lozenge$, used by El Guide et al. [12].

Notation	Description
A	matrix
$A(:, j)$	jth column of A
$A(j, :)$	jth row of A
$\mathcal{A}$ or $\mathbb{A}$	tensor
$\mathcal{A}_j$	jth column of $\mathbb{A}$
$\vec{\mathcal{A}}$	lateral slice
$\vec{\mathcal{A}}_j$	jth lateral slice of $\mathcal{A}$
$\mathcal{A}^{(i)}$ or $A^{(i)}$	ith frontal slice of $\mathcal{A}$
$\widehat{\mathcal{A}}$	FFT of $\mathcal{A}$ along the third mode
$\mathbf{a}$	tube
$(\mathbf{a})_{(:,:,1)}$	first face of $\mathbf{a}$
a	vector with jth entry a_j
$\mathcal{I}$	identity tensor
I	identity matrix
$\vec{e}_1$	first lateral slice of $\mathcal{I}$
e_1	first column of I
$*$	t-product
$\lozenge$	T-diamond product
$\mathbb{A} \circledast a$	$\mathbb{A} \circledast a = \sum_j a_j \mathcal{A}_j$
$\mathcal{A} \circledast a$	$\mathcal{A} \circledast a = \sum_j a_j \vec{\mathcal{A}}_j$
$\mathbb{A} \circledast A$	$\mathbb{A} \circledast A = [\mathbb{A} \circledast A(:, 1), \ldots, \mathbb{A} \circledast A(:, \text{end})]$
$\mathcal{A} \circledast A$	$\mathcal{A} \circledast A = [\mathcal{A} \circledast A(:, 1), \ldots, \mathcal{A} \circledast A(:, \text{end})]$
$\langle \mathcal{A}, \mathcal{B} \rangle$	$\langle \mathcal{A}, \mathcal{B} \rangle = \sum_{ijk} a_{ijk} b_{ijk}$
$\langle \vec{\mathcal{A}}, \vec{\mathcal{B}} \rangle$	$\langle \vec{\mathcal{A}}, \vec{\mathcal{B}} \rangle = \sum_{ik} a_{i1k} b_{i1k}$
$\mathbb{A}^T \lozenge \mathbb{B}$	$[\mathbb{A}^T \lozenge \mathbb{B}]_{ij} = \langle \mathcal{A}_i,\ \mathcal{B}_j \rangle$
$\mathcal{A}^T \lozenge \mathcal{B}$	$[\mathcal{A}^T \lozenge \mathcal{B}]_{ij} = \langle \vec{\mathcal{A}}_i,\ \vec{\mathcal{B}}_j \rangle$
$\langle \mathcal{A}, \mathcal{B} \rangle_{\mathcal{M}}$	$\langle \mathcal{C}, \mathcal{D} \rangle_{\mathcal{M}} = \langle \mathcal{C}, \mathcal{M} * \mathcal{D} \rangle$
$\langle \vec{\mathcal{A}}, \vec{\mathcal{B}} \rangle_{\mathcal{M}}$	$\langle \vec{\mathcal{C}}, \vec{\mathcal{D}} \rangle_{\mathcal{M}} = \langle \vec{\mathcal{C}}, \mathcal{M} * \vec{\mathcal{D}} \rangle$
$\mathbb{A}^T \lozenge_{\mathcal{M}} \mathbb{B}$	$[\mathbb{A}^T \lozenge_{\mathcal{M}} \mathbb{B}]_{ij} = \langle \mathcal{A}_i,\ \mathcal{B}_j \rangle_{\mathcal{M}}$
$\mathcal{A}^T \lozenge_{\mathcal{M}} \mathcal{B}$	$[\mathcal{A}^T \lozenge_{\mathcal{M}} \mathcal{B}]_{ij} = \langle \vec{\mathcal{A}}_i,\ \vec{\mathcal{B}}_j \rangle_{\mathcal{M}}$
$\|a\|_A$	$\|a\|_A = \sqrt{a^T A a}$
$\|\mathcal{A}\|_F$	$\|\mathcal{A}\|_F = \sqrt{\sum_{ijk} a_{ijk}^2}$
$\|\vec{\mathcal{A}}\|_F$	$\|\vec{\mathcal{A}}\|_F = \sqrt{\left(\vec{\mathcal{A}}^T * \vec{\mathcal{A}}\right)_{(:,:,1)}}$
$\|\vec{\mathcal{A}}\|_{\mathcal{M}}$	$\|\vec{\mathcal{x}}\|_{\mathcal{M}} = \sqrt{\left(\vec{\mathcal{A}}^T * \mathcal{M} * \vec{\mathcal{A}}\right)_{(:,:,1)}}$
$\|\mathcal{A}\|_{\mathcal{M}}$	$\|\mathcal{A}\|_{\mathcal{M}} = \sqrt{\texttt{trace}\Big(\left(\mathcal{A}^T * \mathcal{M} * \mathcal{A}\right)_{(:,:,1)}\Big)}$

Table 1: Summary of notation.

3 The Weighted t-Product Golub-Kahan-Tikhonov Method

This section extends the generalized Golub-Kahan bidiagonalization (gen_GKB) process described in [9] for matrices to third order tensors. The gen_GKB process was first proposed by Benbow [4] for generalized least squares problems. Applications of this process to UQ in large-scale Bayesian

linear inverse problems have recently been described in [9, 10, 24]. For additional applications; see, e.g., [1, 2, 23]. We will refer to the tensor version of the `gen_GKB` process as the weighted t-product Golub-Kahan bidiagonalization (W-tGKB) process. The latter process is described by Algorithm 3 below. Its global versions are presented in Section 4.

We also discuss the computation of an approximate solution of the minimization problems (1.3) and (1.10) with the aid of the partial W-tGKB process. Typically, only a few, say $k \ll m$, steps of this process are required to reduce large-scale problems (1.3) and (1.10) to problems of small size. Specifically, k steps of the W-tGKB process applied to $\mathcal{A}$ in (1.3) with initial tensor $\vec{\mathcal{B}}$ reduces $\mathcal{A}$ to a small lower bidiagonal tensor, which we denote by $\bar{\mathcal{P}}_k \in \mathbb{R}^{(k+1)\times k\times n}$. The reduction process is described by Algorithm 3.

Algorithm 3: The partial weighted tensor Golub-Kahan bidiagonalization (W-tGKB) process

Input: $\mathcal{A} \in \mathbb{R}^{\ell\times m\times n}$, $\mathcal{B} \in \mathbb{R}^{\ell\times p\times n}$ such that $\mathcal{A}^T * \mathcal{M}^{-1} * \mathcal{B} \neq \mathcal{O}$, $\mathcal{L} \in \mathbb{R}^{m\times m\times n}$, $\mathcal{M} \in \mathbb{R}^{\ell\times\ell\times n}$ are SPD, $k \geq 1$

Output: Tensors in the decompositions (3.16)

$[\vec{\mathcal{Q}}_1, \mathbf{z}_1] \leftarrow \texttt{Normalize}(\vec{\mathcal{B}}, \mathcal{M}^{-1})$ with $\mathbf{z}_1$ invertible
$[\vec{\mathcal{W}}_1, \mathbf{c}_1] \leftarrow \texttt{Normalize}(\mathcal{A}^T * \mathcal{M}^{-1} * \vec{\mathcal{Q}}_1, \mathcal{L})$ with $\mathbf{c}_1$ invertible
for $i = 1, 2, \ldots, k$ **do**
  $\vec{\mathcal{Q}} \leftarrow \mathcal{A} * \mathcal{L} * \vec{\mathcal{W}}_i - \vec{\mathcal{Q}}_i * \mathbf{c}_i$
  $[\vec{\mathcal{Q}}_{i+1}, \mathbf{z}_{i+1}] \leftarrow \texttt{Normalize}(\vec{\mathcal{Q}}, \mathcal{M}^{-1})$
  if $i < k$ **then**
    $\vec{\mathcal{W}} \leftarrow \mathcal{A}^T * \mathcal{M}^{-1} * \vec{\mathcal{Q}}_{i+1} - \vec{\mathcal{W}}_i * \mathbf{z}_{i+1}$
    $[\vec{\mathcal{W}}_{i+1}, \mathbf{c}_{i+1}] \leftarrow \texttt{Normalize}(\vec{\mathcal{W}}, \mathcal{L})$
  end
end

Algorithm 3 produces the partial W-tGKB decompositions

$$\mathcal{A} * \mathcal{L} * \mathcal{W}_k = \mathcal{Q}_{k+1} * \bar{\mathcal{P}}_k, \quad \mathcal{A}^T * \mathcal{M}^{-1} * \mathcal{Q}_k = \mathcal{W}_k * \mathcal{P}_k^T, \tag{3.16}$$

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 $\mathcal{P}_k$ denotes the leading $k \times k \times n$ subtensor of $\bar{\mathcal{P}}_k$. The lateral slices $\vec{\mathcal{W}}_j \in \mathbb{R}^{m\times 1\times n}$, $j = 1, 2, \ldots, k$, and $\vec{\mathcal{Q}}_{i+1} \in \mathbb{R}^{\ell\times 1\times n}$, $i = 0, 1, \ldots, k$, generated by Algorithm 3 are orthonormal tensor bases for the t-Krylov subspaces

$$\mathbb{K}_k(\mathcal{A}^T * \mathcal{M}^{-1} * \mathcal{A} * \mathcal{L}, \mathcal{A}^T * \mathcal{M}^{-1} * \vec{\mathcal{B}}) \quad \text{and} \quad \mathbb{K}_{k+1}(\mathcal{A} * \mathcal{L} * \mathcal{A}^T * \mathcal{M}^{-1}, \vec{\mathcal{B}}),$$

respectively. The lateral slices $\vec{\mathcal{W}}_j$ and $\vec{\mathcal{Q}}_i$ define the tensors

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

which satisfy

$$\mathcal{W}_k^T * \mathcal{L} * \mathcal{W}_k = \mathcal{I}_k \quad \text{and} \quad \mathcal{Q}_{k+1}^T * \mathcal{M}^{-1} * \mathcal{Q}_{k+1} = \mathcal{I}_{k+1}. \tag{3.17}$$

Theorem 3.1 below shows that the application of the operators $\mathcal{L}$ and $\mathcal{M}$ in the spatial domain is equivalent to their application in the Fourier domain. Application of these operators in the spatial domain may reduce the computing time since transformation of $\mathcal{L}$ and $\mathcal{M}$ to and from the Fourier domain is not required. We apply the SPD tensors $\mathcal{L}$ and $\mathcal{M}$ in the spatial domain in the computed examples of Section 5.

Theorem 3.1. Let the first frontal slice $\mathcal{L}^{(1)}$ of the tensor $\mathcal{L} \in \mathbb{R}^{s\times m\times n}$ be nonzero and let the remaining frontal slices, $\mathcal{L}^{(i)}$, $i = 2, 3, \ldots, n$, be zero matrices. Let $\mathcal{X} \in \mathbb{R}^{m\times p\times n}$ have frontal slices $\mathcal{X}^{(i)}$, $i = 1, 2, \ldots, n$. Define

$$\mathcal{L}(\mathcal{X}) := \mathcal{L} * \mathcal{X}.$$

Then

$$\mathcal{L}(\mathcal{X})^{(i)} = \mathcal{L}^{(1)}\mathcal{X}^{(i)}, \quad i = 1, 2, \ldots, n.$$

Proof: We have

$$\begin{aligned}
\mathcal{L} * \mathcal{X} &= \texttt{fold}\left(\texttt{bcirc}(\mathcal{L}) \cdot \texttt{unfold}(\mathcal{X})\right) \\
&= \texttt{fold}\left(\begin{bmatrix} \mathcal{L}^{(1)} & O & \ldots & O \\ O & \mathcal{L}^{(1)} & \ldots & O \\ \vdots & \vdots & \ddots & \vdots \\ O & O & \ldots & \mathcal{L}^{(1)} \end{bmatrix} \begin{bmatrix} \mathcal{X}^{(1)} \\ \mathcal{X}^{(2)} \\ \vdots \\ \mathcal{X}^{(n)} \end{bmatrix}\right) \\
&= \texttt{fold}\left(\begin{bmatrix} \mathcal{L}^{(1)}\mathcal{X}^{(1)} \\ \mathcal{L}^{(1)}\mathcal{X}^{(2)} \\ \vdots \\ \mathcal{L}^{(1)}\mathcal{X}^{(n)} \end{bmatrix}\right),
\end{aligned} \tag{3.18}$$

and the proof follows. □

3.1 The W-tGKT method

This subsection describes the W-tGKT method for the approximate solution of least squares problems of the form (1.3). We also apply this method p times to determine an approximate solution of (1.10) by working with the data tensor slices $\vec{\mathcal{B}}_j$, $j = 1, 2, \ldots, p$, of $\mathcal{B}$ independently.

Let $\vec{\mathcal{Y}} = \mathcal{W}_k * \vec{\mathcal{Z}}$. Substituting the left-hand side of the W-tGKB decompositions (3.16) into (1.7) and using (3.17) yields

$$\min_{\vec{\mathcal{Z}}\in\mathbb{R}^{k\times 1\times n}} \{\|\bar{\mathcal{P}}_k * \vec{\mathcal{Z}} - \mathcal{Q}_{k+1}^T * \vec{\mathcal{B}}\|_F^2 + \mu^{-1}\|\vec{\mathcal{Z}}\|_F^2\}. \tag{3.19}$$

Since $\vec{\mathcal{B}} = \vec{\mathcal{Q}}_1 * \mathbf{z}_1$ (cf. Algorithm 3), we get

$$\mathcal{Q}_{k+1}^T * \vec{\mathcal{B}} = \vec{e}_1 * \mathbf{z}_1 \in \mathbb{R}^{(k+1)\times 1\times n}, \tag{3.20}$$

where the $(1, 1, 1)$th entry of $\vec{e}_1 \in \mathbb{R}^{m\times 1\times n}$ is 1 and the remaining entries are zero. Substituting (3.20) into (3.19), we obtain

$$\min_{\vec{\mathcal{Z}}\in\mathbb{R}^{k\times 1\times n}} \{\|\bar{\mathcal{P}}_k * \vec{\mathcal{Z}} - \vec{e}_1 * \mathbf{z}_1\|_F^2 + \mu^{-1}\|\vec{\mathcal{Z}}\|_F^2\}. \tag{3.21}$$

The equivalency of (1.3) and (3.21) follows from (3.17). Thus,

$$\min_{\vec{\mathcal{Y}}\in\mathbb{K}_k} \{\|\mathcal{A} * \mathcal{L} * \vec{\mathcal{Y}} - \vec{\mathcal{B}}\|_{\mathcal{M}^{-1}}^2 + \mu^{-1}\|\vec{\mathcal{Y}}\|_{\mathcal{L}}^2\} \iff \min_{\vec{\mathcal{Z}}\in\mathbb{R}^{k\times 1\times n}} \{\|\bar{\mathcal{P}}_k * \vec{\mathcal{Z}} - \vec{e}_1 * \mathbf{z}_1\|_F^2 + \mu^{-1}\|\vec{\mathcal{Z}}\|_F^2\}, \tag{3.22}$$

where $\mathbb{K}_k = \mathbb{K}_k(\mathcal{A}^T * \mathcal{M}^{-1} * \mathcal{A} * \mathcal{L}, \mathcal{A}^T * \mathcal{M}^{-1} * \vec{\mathcal{B}})$.

A suitable way to compute the solution of the minimization problem on the right-hand side of (3.22) is to solve the least squares problem

$$\min_{\vec{\mathcal{Z}}\in\mathbb{R}^{k\times 1\times n}} \left\| \begin{bmatrix} \bar{\mathcal{P}}_k \\ \mu^{-1/2}\mathcal{I} \end{bmatrix} * \vec{\mathcal{Z}} - \begin{bmatrix} \vec{e}_1 * \mathbf{z}_1 \\ \vec{\mathcal{O}} \end{bmatrix} \right\|_F, \tag{3.23}$$

using [27, Algorithm 3]. The solution $\vec{\mathcal{Z}}_{\mu,k}$ can be expressed as

$$\vec{\mathcal{Z}}_{\mu,k} = (\bar{\mathcal{P}}_k^T * \bar{\mathcal{P}}_k + \mu^{-1}\mathcal{I})^{-1} * \bar{\mathcal{P}}_k^T * \vec{e}_1 * \mathbf{z}_1, \tag{3.24}$$

and the associated approximate solution of (1.3) is given by

$$\vec{\mathcal{X}}_{\mu,k} = \mathcal{L} * \mathcal{W}_k * \vec{\mathcal{Z}}_{\mu,k}.$$

The discrepancy principle (1.5) is used to determine the regularization parameter and the number of steps of the W-tGKB process. We show below that (1.5) is equivalent to

$$\|\bar{\mathcal{P}}_k * \vec{\mathcal{Z}}_{\mu,k} - \vec{e}_1 * \mathbf{z}_1\|_F = \eta\delta. \tag{3.25}$$

Thus, we have to determine $\mu > 0$ so that (3.25) holds. Define the function

$$\phi_k(\mu) := \|\bar{\mathcal{P}}_k * \vec{\mathcal{Z}}_{\mu,k} - \vec{e}_1 * \mathbf{z}_1\|_F^2. \tag{3.26}$$

Substituting (3.24) into (3.26), and using the identity

$$\mathcal{I} - \bar{\mathcal{P}}_k * (\bar{\mathcal{P}}_k^T * \bar{\mathcal{P}}_k + \mu^{-1}\mathcal{I})^{-1} * \bar{\mathcal{P}}_k^T = (\mu\bar{\mathcal{P}}_k * \bar{\mathcal{P}}_k^T + \mathcal{I})^{-1} \tag{3.27}$$

as well as (2.15), we obtain

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

Therefore, (3.25) becomes

$$\phi_k(\mu) - (\eta\delta)^2 = 0. \tag{3.28}$$

The dependence of $\phi_k(\mu)$ on k for a fixed $\mu > 0$ is established in [27]. Moreover, $\phi_k(\mu)$ is shown to be a decreasing and convex function of $\mu > 0$. We can define $\phi_k(0) = \|\vec{e}_1 * \mathbf{z}_1\|_F^2$ by continuity. In the computed examples of Section 5, we use the bisection method to solve (3.28). The following result establishes the equivalence between the discrepancy principle (1.5) and (3.25).

Proposition 3.1. Let $\phi_k(\mu)$ be defined by (3.26), and assume that $\mu = \mu_k$ solves $\phi_k(\mu) = \eta^2\delta^2$ and that $\vec{\mathcal{Z}}_{\mu,k}$ solves

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

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

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

Proof: Substituting $\vec{\mathcal{X}}_{\mu,k} = \mathcal{L} * \vec{\mathcal{Y}}_{\mu,k}$ and $\vec{\mathcal{Y}}_{\mu,k} = \mathcal{W}_k * \vec{\mathcal{Z}}_{\mu,k}$ into (1.5), and using the left-hand side decomposition of (3.16), as well as (3.20) and (3.17), shows that

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

We refer to the solution method for (1.3) described above as the W-tGKT method. It is implemented by Algorithm 4 below with $p = 1$.

The remainder of this subsection describes an algorithm for the approximate solution of (1.10) by solving the minimization problem (1.11). Let the data tensor $\mathcal{B}$ have the lateral data slices $\vec{\mathcal{B}}_j$, $j = 1, 2, \ldots, p$, and let $\vec{\mathcal{B}}_{j,\text{true}}$ denote the unknown error-free data slice associated with the available error-contaminated slice $\vec{\mathcal{B}}_j$. Assume that bounds for the norm of the errors

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

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

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

cf. (1.2) and (1.4). Let $\vec{\mathcal{X}}_j = \mathcal{L} * \vec{\mathcal{Y}}_j$, $j = 1, 2, \ldots, p$, and consider (1.11) as p separate Tikhonov minimization problems

$$\min_{\vec{\mathcal{Y}}_j \in \mathbb{R}^{m\times 1\times n}} \{\|\mathcal{A} * \mathcal{L} * \vec{\mathcal{Y}}_j - \vec{\mathcal{B}}_j\|_{\mathcal{M}^{-1}}^2 + \mu^{-1}\|\vec{\mathcal{Y}}_j\|_{\mathcal{L}}^2\}, \quad j = 1, 2, \ldots, p. \tag{3.29}$$

We apply the W-tGKT method described above to solve the p problems (3.29) independently by using Algorithm 4, and we refer to this solution method for (1.10) by solving the problems (3.29) independently as the W-tGKT$_p$ method.

Algorithm 4: The W-tGKT$_p$ method for the approximate solution of (1.10) by solving the problems (3.29) independently

Input: $\mathcal{A}, \vec{\mathcal{B}}_1, \vec{\mathcal{B}}_2, \ldots, \vec{\mathcal{B}}_p, \delta_1, \delta_2, \ldots, \delta_p, \mathcal{L}, \mathcal{M}, \eta > 1, k_{\text{init}} = 2$
Output: Approximate solution of (1.10) when $p > 1$ and of (1.3) when $p = 1$

$\vec{e}_1 \leftarrow \mathcal{I}(:, 1, :)$
for $j = 1, 2, \ldots, p$ **do**
  $k \leftarrow k_{\text{init}}$, $[\vec{\mathcal{Q}}_1, \mathbf{z}_1] \leftarrow \texttt{Normalize}(\vec{\mathcal{B}}_j, \mathcal{M}^{-1})$
  Compute $\mathcal{W}_k$, $\mathcal{Q}_{k+1}$ and $\bar{\mathcal{P}}_k$ by Algorithm 3, and let $\vec{e}_1 \leftarrow \mathcal{I}(:, 1, :)$
  Solve the minimization problem

$$\min_{\vec{\mathcal{Z}} \in \mathbb{R}^{k \times 1 \times n}} \|\bar{\mathcal{P}}_k * \vec{\mathcal{Z}} - \vec{e}_1 * \mathbf{z}_1\|_F$$

  for $\vec{\mathcal{Z}}_k$ by using [27, Algorithm 3]
  while $\|\bar{\mathcal{P}}_k * \vec{\mathcal{Z}}_k - \vec{e}_1 * \mathbf{z}_1\|_F \geq \eta \delta_j$ **do**
    $k \leftarrow k + 1$
    `Go to step 3`
  end
  Determine the regularization parameter by the discrepancy principle, i.e., compute the zero $\mu_k > 0$ of

$$\xi_k(\mu_k) := \|\bar{\mathcal{P}}_k * \vec{\mathcal{Z}}_{j,\mu_k} - \vec{e}_1 * \mathbf{z}_1\|_F^2 - \eta^2 \delta_j^2$$

  and the associated solution $\vec{\mathcal{Z}}_{j,\mu_k}$ of

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

  by using [27, Algorithm 3]
  Compute $\vec{\mathcal{Y}}_{j,\mu_k} \leftarrow \mathcal{W}_k * \vec{\mathcal{Z}}_{j,\mu_k}$, $\vec{\mathcal{X}}_{j,\mu_k} \leftarrow \mathcal{L} * \vec{\mathcal{Y}}_{j,\mu_k}$
end

4 The WGG-tGKT and WG-tGKT Methods

We describe the WGG-tGKT and WG-tGKT methods for the approximate solution of (1.3) and (1.10). The latter method is designed to compute an approximate solution of (1.3). When applied to solve (1.10), the method works with the lateral slices of $\vec{\mathcal{B}}_j$, $j = 1, 2, \ldots, p$, of $\mathcal{B}$ one at a time and independently. We refer to the resulting method for solving (1.10) as the WG-tGKT$_p$ method. The WGG-tGKT method solves (1.10) by working with the lateral slices of the tensor $\mathcal{B}$ simultaneously.

4.1 The WGG-tGKT method

This subsection describes the weighted generalized global t-product Golub-Kahan bidiagonalization (WGG-tGKB) process and discusses how it can be applied to reduce the large-scale problem (1.10) to a problem of small size. This is achieved by applying $k \ll m$ steps of the WGG-tGKB process to $\mathcal{A}$. We assume that there is no breakdown of this process. This is the generic situation. The WGG-tGKB process, which is described by Algorithm 5, yields the decompositions

$$\mathcal{A} * \mathcal{L} * \mathbb{W}_k = \mathbb{Q}_{k+1} \circledast \bar{P}_k, \quad \mathcal{A}^T * \mathcal{M}^{-1} * \mathbb{Q}_k = \mathbb{W}_k \circledast P_k^T, \tag{4.30}$$

where

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

As usual, the tensors $\mathcal{L} \in \mathbb{R}^{m\times m\times n}$ and $\mathcal{M} \in \mathbb{R}^{\ell\times\ell\times n}$ are assumed to be SPD. Moreover,

$$\mathbb{Q}_{k+1}^T \Diamond_{\mathcal{M}^{-1}} \mathbb{Q}_{k+1} = I_{k+1}, \quad \mathbb{W}_k^T \Diamond_{\mathcal{L}} \mathbb{W}_k = I_k, \tag{4.31}$$

and

$$\begin{aligned} \mathcal{A} * \mathcal{L} * \mathbb{W}_k &= [\mathcal{A} * \mathcal{L} * \mathcal{W}_1, \mathcal{A} * \mathcal{L} * \mathcal{W}_2, \ldots, \mathcal{A} * \mathcal{L} * \mathcal{W}_k] \in \mathbb{R}^{\ell\times kp\times n}, \\ \mathbb{Q}_{k+1} \circledast \bar{P}_k &= [\mathbb{Q}_{k+1} \circledast \bar{P}_k(:,1), \mathbb{Q}_{k+1} \circledast \bar{P}_k(:,2), \ldots, \mathbb{Q}_{k+1} \circledast \bar{P}_k(:,k)] \in \mathbb{R}^{\ell\times kp\times n}, \end{aligned} \tag{4.32}$$

where $\mathcal{A}^T * \mathcal{M}^{-1} * \mathbb{Q}_k$ and $\mathbb{W}_k \circledast P_k^T$ are defined similarly as (4.32).

Details of the computations of the WGG-tGKB process are described by Algorithm 5. The tensors $\mathcal{W}_j \in \mathbb{R}^{m\times p\times n}$, $j = 1, 2, \ldots, k$, and $\mathcal{Q}_{i+1} \in \mathbb{R}^{\ell\times p\times n}$, $i = 0, 1, \ldots, k$, generated by the algorithm form orthogonal tensor bases for the t-Krylov subspaces $\mathbb{K}_k(\mathcal{A}^T * \mathcal{M}^{-1} * \mathcal{A} * \mathcal{L}, \mathcal{A}^T * \mathcal{M}^{-1} * \mathcal{B})$ and $\mathbb{K}_{k+1}(\mathcal{A} * \mathcal{L} * \mathcal{A}^T * \mathcal{M}^{-1}, \mathcal{B})$, respectively. The lower bidiagonal matrix $\bar{P}_k$ in (4.32) is given by

$$\bar{P}_k = \begin{bmatrix} \alpha_1 & & & & \\ \beta_2 & \alpha_2 & & & \\ & \beta_3 & \alpha_3 & & \\ & & \ddots & \ddots & \\ & & & \beta_k & \alpha_k \\ & & & & \beta_{k+1} \end{bmatrix} \in \mathbb{R}^{(k+1)\times(k)}, \tag{4.33}$$

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

$$\mathcal{B} = \mathbb{Q}_{k+1} \circledast e_1\beta_1, \quad e_1 = [1, 0, \ldots, 0]^T, \tag{4.34}$$

is easily deduced from Algorithm 5.

Proposition 4.1. Let the tensors $\mathbb{Q}_{k+1}$ and $\mathcal{M}$ be defined as above. Let $y \in \mathbb{R}^{k+1}$. Then

$$\|\mathbb{Q}_{k+1} \circledast y\|_{\mathcal{M}^{-1}} = \|y\|_2, \tag{4.35}$$

where $\|\cdot\|_2$ denotes the Euclidean vector norm.

Proof:

$$\|\mathbb{Q}_{k+1} \circledast y\|_{\mathcal{M}^{-1}}^2 = \left\langle \sum_{i=1}^{k+1} y_j \mathcal{Q}_j, \sum_{i=1}^{k+1} y_j \mathcal{Q}_j \right\rangle_{\mathcal{M}^{-1}} = \sum_{i=1}^{k+1} y_j^2 \langle \mathcal{Q}_j, \mathcal{Q}_j \rangle_{\mathcal{M}^{-1}} = \sum_{i=1}^{k+1} y_j^2 = \|y\|_2^2,$$

since the tensors $\mathcal{Q}_j$, $j = 1, 2, \ldots, k$, are orthogonal, i.e.,

$$\langle \mathcal{Q}_i, \mathcal{Q}_j \rangle_{\mathcal{M}^{-1}} = \begin{cases} 1 & i = j, \\ 0 & i \neq j. \end{cases}$$

□

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

$$\|\mathcal{Q} \circledast y\|_{\mathcal{M}^{-1}} = \|y\|_2; \tag{4.36}$$

see [12] for the analogous result when $\mathcal{M}^{-1} = \mathcal{I}$.

We compute an approximate solution of (1.10) similarly as described in Subsection 3.1. Thus, letting $\mathcal{Y} = \mathbb{W}_k \circledast z$ and using the left-hand side of (4.30), as well as (4.34) and (4.31), the minimization problem (1.11) reduces to

$$\min_{z\in\mathbb{R}^k} \{\|\bar{P}_k z - e_1\beta_1\|_2^2 + \mu^{-1}\|z\|_2^2\}. \tag{4.37}$$

Algorithm 5: Partial weighted generalized global tGKB process

Input: $\mathcal{A} \in \mathbb{R}^{\ell\times m\times n}$, $\mathcal{B} \in \mathbb{R}^{\ell\times p\times n}$ such that $\mathcal{A}^T * \mathcal{M}^{-1} * \mathcal{B} \neq \mathcal{O}$, $\mathcal{L} \in \mathbb{R}^{m\times m\times n}$, $\mathcal{M} \in \mathbb{R}^{\ell\times\ell\times n}$ are SPD, $k \geq 1$

Output: Matrices and tensors in the decompositions (4.30)

$\beta_1 \leftarrow \|\mathcal{B}\|_{\mathcal{M}^{-1}}$, $\mathcal{Q}_1 \leftarrow \frac{1}{\beta_1}\mathcal{B}$
$\alpha_1 \leftarrow \|\mathcal{A}^T * \mathcal{M}^{-1} * \mathcal{Q}_1\|_{\mathcal{L}}$, $\mathcal{W}_1 \leftarrow \frac{1}{\alpha_1}(\mathcal{A}^T * \mathcal{M}^{-1} * \mathcal{Q}_1)$
for $j = 1, 2, \ldots, k$ **do**
  $\mathcal{Q} \leftarrow \mathcal{A} * \mathcal{L} * \mathcal{W}_j - \alpha_j \mathcal{Q}_j$
  $\beta_{j+1} \leftarrow \|\mathcal{Q}\|_{\mathcal{M}^{-1}}$, $\mathcal{Q}_{j+1} \leftarrow \mathcal{Q}/\beta_{j+1}$
  if $j < k$ **then**
    $\mathcal{W} \leftarrow \mathcal{A}^T * \mathcal{M}^{-1} * \mathcal{Q}_{j+1} - \beta_{j+1}\mathcal{W}_j$
    $\alpha_{j+1} \leftarrow \|\mathcal{W}\|_{\mathcal{L}}$, $\mathcal{W}_{j+1} \leftarrow \mathcal{W}/\alpha_{j+1}$
  end
end

The minimization problem (4.37) is analogous to (3.23). We compute its solution by solving

$$\min_{z\in\mathbb{R}^k} \left\| \begin{bmatrix} \bar{P}_k \\ \mu^{-1/2} I \end{bmatrix} z - \begin{bmatrix} e_1\beta_1 \\ 0 \end{bmatrix} \right\|_2. \tag{4.38}$$

Denote the solution by $z_{\mu,k}$. Then the associated approximate solution of (1.10) is given by

$$\vec{\mathcal{X}}_{\mu,k} = \mathcal{L} * \mathbb{W}_k \circledast z_{\mu,k}.$$

We determine the regularization parameter μ by the discrepancy principle based on the $\mathcal{M}^{-1}$-norm. This assumes knowledge of a bound

$$\|\mathcal{E}\|_{\mathcal{M}^{-1}} \leq \delta$$

for the error tensor $\mathcal{E}$ in $\mathcal{B}$. Thus, we choose $\mu > 0$ so that the solution $z_{\mu,k}$ of (4.38) satisfies

$$\|\bar{P}_k z_{\mu,k} - e_1\beta_1\|_2 = \eta\delta. \tag{4.39}$$

Define the function

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

where $z_{\mu,k}$ solves (4.38). Using an identity analogous to (3.27), we obtain

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

The discrepancy principle (4.39) can be satisfied for reasonable values of $\eta\delta$ by choosing a large enough value of k; see [27, Proposition 4.3]. It can be shown that the function $\mu \mapsto \psi_k(\mu)$ is decreasing and convex with $\psi_k(0) = \beta_1^2$. The following result is a matrix analogue of Proposition 3.1.

Proposition 4.2. Let μ_k solve $\psi_k(\mu) = \eta^2\delta^2$ and suppose that $z_{\mu,k}$ is the solution of (4.37) with $\mu = \mu_k$. Let $\mathcal{Y}_{\mu,k} = \mathbb{W}_k \circledast z_{\mu,k}$. Then the associated approximate solution $\mathcal{X}_{\mu,k} = \mathcal{L} * \mathcal{Y}_{\mu,k}$ of (1.10) satisfies

$$\|\mathcal{A} * \mathcal{X}_{\mu,k} - \mathcal{B}\|_{\mathcal{M}^{-1}}^2 = \beta_1^2 e_1^T(\mu\bar{P}_k\bar{P}_k^T + I)^{-2}e_1. \tag{4.40}$$

Proof: Substituting $\mathcal{X}_{\mu,k} = \mathcal{L} * \mathcal{Y}_{\mu,k}$ and $\mathcal{Y}_{\mu,k} = \mathbb{W}_k \circledast z_{\mu,k}$ into left-hand side of (4.40), using the left-hand side of (4.30), (4.31) and (4.34), as well as left-hand side of (4.36), gives

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

We refer to the solution method described above as the WGG-tGKT method. This method is implemented by Algorithm 6.

Algorithm 6: The WGG-tGKT method for computing an approximate solution of (1.10)

Input: $\mathcal{A}$, $\mathcal{B}$, δ, $\mathcal{L}$, $\mathcal{M}$, $\eta > 1$, $k_{\text{init}} = 2$
Output: Approximate solution of (1.10)

$k \leftarrow k_{\text{init}}$, $\beta_1 \leftarrow \|\mathcal{B}\|_{\mathcal{M}^{-1}}$, $\mathcal{Q}_1 \leftarrow \frac{1}{\beta_1}\mathcal{B}$

Compute $\mathbb{W}_k$, $\mathbb{Q}_{k+1}$, and $\bar{P}_k$ by Algorithm 5

Let $z_k \in \mathbb{R}^k$ solve the minimization problem

$$\min_{z\in\mathbb{R}^k} \|\bar{P}_k z - e_1\beta_1\|_2$$

while $\|\bar{P}_k z_k - e_1\beta_1\|_2 \geq \eta\delta$ **do**
    $k \leftarrow k + 1$
    `Go to step 2`
end

Determine the regularization parameter by the discrepancy principle, i.e., compute the zero $\mu_k > 0$ of

$$\varphi_k(\mu) := \|\bar{P}_k z_{\mu,k} - e_1\beta_1\|_2^2 - \eta^2\delta^2$$

and the associated solution $z_{\mu,k}$ of

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

Compute $\mathcal{Y}_{\mu,k} \leftarrow \mathbb{W}_k \circledast z_{\mu,k}$, $\mathcal{X}_{\mu,k} \leftarrow \mathcal{L} * \mathcal{Y}_{\mu,k}$

4.2 The WG-tGKT method

The WG-tGKT method for the approximate solution of (1.3) first reduces the tensor $\mathcal{A}$ in (1.3) to a small lower bidiagonal matrix by carrying out a few, say k, steps of the weighted global t-product Golub-Kahan bidiagonalization (WG-tGKB) process, which is described by Algorithm 7. We assume that k is small enough to avoid breakdown. This is the generic situation. Algorithm 7 yields the partial WG-tGKB decompositions

$$\mathcal{A} * \mathcal{L} * \mathcal{W}_k = \mathcal{Q}_{k+1} \circledast \bar{B}_k, \quad \mathcal{A}^T * \mathcal{M}^{-1} * \mathcal{Q}_k = \mathcal{W}_k \circledast B_k^T, \tag{4.41}$$

where

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

and

$$\mathcal{Q}_{k+1}^T \lozenge_{\mathcal{M}^{-1}} \mathcal{Q}_{k+1} = I_{k+1}, \quad \mathcal{W}_k^T \lozenge_{\mathcal{L}} \mathcal{W}_k = I_k.$$

The tensors $\mathcal{L}$ and $\mathcal{M}$ are assumed to be SPD. The expressions in (4.41) are defined similarly to (4.32), and the bidiagonal matrix $\bar{B}_k \in \mathbb{R}^{(k+1)\times k}$ has a form analogous to (4.33). The tensors $\vec{\mathcal{W}}_j \in \mathbb{R}^{\ell\times 1\times n}$, $j = 1, 2, \ldots, k$, and $\vec{\mathcal{Q}}_{i+1} \in \mathbb{R}^{m\times 1\times n}$, $i = 0, 1, \ldots, k$, generated by Algorithm 7 form orthonormal tensor bases for the t-Krylov subspaces $\mathbb{K}_k(\mathcal{A}^T * \mathcal{M}^{-1} * \mathcal{A} * \mathcal{L}, \mathcal{A}^T * \mathcal{M}^{-1} * \vec{\mathcal{B}})$ and $\mathbb{K}_{k+1}(\mathcal{A} * \mathcal{L} * \mathcal{A}^T * \mathcal{M}^{-1}, \vec{\mathcal{B}})$, respectively.

Algorithm 7: Partial weighted global tGKB (WG-tGKB) process

Input: $\mathcal{A} \in \mathbb{R}^{\ell\times m\times n}$, $\vec{\mathcal{B}} \in \mathbb{R}^{\ell\times 1\times n}$ such that $\mathcal{A}^T * \mathcal{M}^{-1} * \vec{\mathcal{B}} \neq \vec{\mathcal{O}}$, $\mathcal{L} \in \mathbb{R}^{m\times m\times n}$ and $\mathcal{M} \in \mathbb{R}^{\ell\times \ell\times n}$ are SPD, $k \geq 1$

Output: Matrices and tensors in the decompositions (4.41)

$\beta_1 \leftarrow \|\vec{\mathcal{B}}\|_{\mathcal{M}^{-1}}$, $\vec{\mathcal{Q}}_1 \leftarrow \frac{1}{\beta_1}\vec{\mathcal{B}}$

$\alpha_1 \leftarrow \|\mathcal{A}^T * \mathcal{M}^{-1} * \vec{\mathcal{Q}}_1\|_{\mathcal{L}}$, $\vec{\mathcal{W}}_1 \leftarrow \frac{1}{\alpha_1}(\mathcal{A}^T * \mathcal{M}^{-1} * \vec{\mathcal{Q}}_1)$

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

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

  $\beta_{j+1} \leftarrow \|\vec{\mathcal{Q}}\|_{\mathcal{M}^{-1}}$, $\vec{\mathcal{Q}}_{j+1} \leftarrow \vec{\mathcal{Q}}/\beta_{j+1}$

  if $j < k$ **then**

    $\vec{\mathcal{W}} \leftarrow \mathcal{A}^T * \mathcal{M}^{-1} * \vec{\mathcal{Q}}_{j+1} - \beta_{j+1}\vec{\mathcal{W}}_j$

    $\alpha_{j+1} \leftarrow \|\vec{\mathcal{W}}\|_{\mathcal{L}}$, $\vec{\mathcal{W}}_{j+1} \leftarrow \vec{\mathcal{W}}/\alpha_{j+1}$

  end

end

Let $\vec{\mathcal{Y}} = \mathcal{W}_k \circledast y$. Then following a similar approach as in Subsection 4.1, we reduce (1.3) to the Tikhonov minimization problem in standard form

$$\min_{z\in\mathbb{R}^k} \{\|\bar{B}_k z - e_1\beta_1\|_2^2 + \mu^{-1}\|z\|_2^2\}. \tag{4.42}$$

The solution of (4.42) determines an approximate solution of (1.3). We refer to this approach of computing an approximate solution of (1.3) as the WG-tGKT method.

The solution method for (1.10) defined by applying the WG-tGKT method p times to the problems (3.29) independently is referred to as the WG-tGKT$_p$ method. This p-method is described by Algorithm 8.

Algorithm 8: The WG-tGKT$_p$ method for the approximate solution of (1.10).

Input: $\mathcal{A}$, $\vec{\mathcal{B}}_1, \vec{\mathcal{B}}_2, \ldots, \vec{\mathcal{B}}_p$, $\mathcal{L}$, $\mathcal{M}$, $\delta_1, \delta_2, \ldots, \delta_p$, $\eta > 1$, $k_{\text{init}} = 2$

Output: Approximate solution of (1.10)

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

  $k \leftarrow k_{\text{init}}$, $\beta_1 \leftarrow \|\vec{\mathcal{B}}_j\|_{\mathcal{M}^{-1}}$, $\vec{\mathcal{Q}}_1 \leftarrow \frac{1}{\beta_1}\vec{\mathcal{B}}_j$

  Compute $\mathcal{W}_k$, $\mathcal{Q}_{k+1}$, and $\bar{B}_k$ by Algorithm 7

  Solve the minimization problem

$$\min_{z\in\mathbb{R}^k} \|\bar{B}_k z - e_1\beta_1\|_2$$

  for z_k

  while $\|\bar{B}_k z_k - e_1\beta_1\|_2 \geq \eta\delta_j$ **do**

    $k \leftarrow k + 1$

    `Go to step 3`

  end

  Determine the regularization parameter by the discrepancy principle, i.e., compute the zero $\mu_k > 0$ of

$$\varphi_k(\mu_k) := \|\bar{B}_k z_{j,\mu_k} - e_1\beta_1\|_2^2 - \eta^2\delta_j^2$$

  and the associated solution z_{j,μ_k} of

$$\min_{z\in\mathbb{R}^k} \left\| \begin{bmatrix} \bar{B}_k \\ \mu_k^{-1/2} I \end{bmatrix} z - \begin{bmatrix} e_1\beta_1 \\ 0 \end{bmatrix} \right\|_2$$

  Compute $\mathcal{Y}_{j,\mu_k} \leftarrow \mathcal{W}_k \circledast z_{j,\mu_k}$, $\vec{\mathcal{X}}_{j,\mu_k} \leftarrow \mathcal{L} * \mathcal{Y}_{j,\mu_k}$

end

5 Numerical Examples

All computations reported in this section are carried out in MATLAB R2021a with about 15 significant decimal digits on a Dell computer running Windows 10 with 11th Gen Intel(R) Core(TM) i7-1165G7 @ 2.80GHz and 16 GB RAM. We compare the performance of the W-tGKT and WG-tGKT methods for the solution of (1.3) and (1.10), and the WGG-tGKT method for solving the latter problem. The implementation of the described methods is carried out in the spatial domain in Examples 5.1-5.3 using (3.18). All examples are concerned with image or video restoration.

Let $\vec{\mathcal{X}}_{\text{method}} \in \mathbb{R}^{m\times 1\times n}$ denote the approximate solution of (1.3) computed by a particular solution method, and let $\vec{\mathcal{X}}_{\text{true}} \in \mathbb{R}^{m\times 1\times n}$ stand for the desired solution (e.g., a blur- and noise-free image). To compare the performance of the solution methods discussed, we tabulate the relative errors in the Frobenius norm,

$$E_{\text{method}} = \frac{\|\vec{\mathcal{X}}_{\text{method}} - \vec{\mathcal{X}}_{\text{true}}\|_F}{\|\vec{\mathcal{X}}_{\text{true}}\|_F}. \tag{5.43}$$

We also display the Peak Signal-to-Noise Ratio,

$$\text{PSNR} := 10\,\log_{10}\left(\frac{\text{MAX}_{\vec{\mathcal{X}}_{\text{true}}}}{\sqrt{\text{MSE}}}\right),$$

where $\text{MAX}_{\vec{\mathcal{X}}_{\text{true}}}$ stands for the maximum pixel value of the desired image $\vec{\mathcal{X}}_{\text{true}}$. The mean square error is given by

$$\text{MSE} = \frac{1}{mn}\sum_{i=1}^{m}\sum_{k=1}^{n}\left(\vec{\mathcal{X}}_{\text{true}}(i,1,k) - \mathcal{X}_{\text{method}}(i,1,k)\right)^2.$$

The relative error and the PSNR of computed approximate solutions $\mathcal{X}_{\text{method}}$ of (1.10), whose data is given by a tensor $\mathcal{B} \in \mathbb{R}^{m\times p\times n}$ for some $p > 1$, are determined analogously. For the sake of brevity, we only explicitly discuss the situation when the data is a tensor slice $\vec{\mathcal{B}} \in \mathbb{R}^{m\times 1\times n}$.

We generate a "noise" tensor $\vec{\mathcal{E}}$ that simulates the error in the data tensor $\vec{\mathcal{B}} = \vec{\mathcal{B}}_{\text{true}} + \vec{\mathcal{E}}$ in (1.3) and has a specified covariance tensor as described below.

Proposition 5.1. Let the entries of the tensor $\vec{\mathcal{E}}_\rho \in \mathbb{R}^{\ell\times 1\times n}$ be normally distributed with mean zero and variance $\rho > 0$, and let $\mathcal{C}$ be a tensor of compatible size. Then the covariance tensor for $\vec{\mathcal{E}} = \mathcal{C} * \vec{\mathcal{E}}_\rho$ is $\rho^2\mathcal{C} * \mathcal{C}^T$.

Proof: This result is well known in the situation when $\vec{\mathcal{E}}_\rho$ is a vector and $\mathcal{C}$ is a matrix; see, e.g., [8, Lemma 2.1.1] for a proof for this case. The statistical properties of $\vec{\mathcal{E}}$ are independent of the operations `fold` and `unfold` in the definition (2.12) of the t-product. We apply [8, Lemma 2.1.1] since $\texttt{bcirc}(\mathcal{C})$ is a matrix and $\texttt{unfold}(\vec{\mathcal{E}}_\rho)$ is a vector. We also use the fact that $\texttt{bcirc}(\mathcal{C}^T) = (\texttt{bcirc}(\mathcal{C}))^T$, and $\texttt{bcirc}(\mathcal{C} * \mathcal{C}^T) = \texttt{bcirc}(\mathcal{C}) \cdot \texttt{bcirc}(\mathcal{C}^T)$. Let $\texttt{Cov}(\cdot)$ denote the covariance of a known quantity. Then

$$\begin{aligned}
\texttt{Cov}(\vec{\mathcal{E}}) &= \texttt{fold}(\texttt{Cov}(\texttt{bcirc}(\mathcal{C})\ \texttt{unfold}(\vec{\mathcal{E}}_\rho))) \\
&= \rho^2\texttt{fold}(\texttt{bcirc}(\mathcal{C}) \cdot (\texttt{bcirc}(\mathcal{C}))^T) \\
&= \rho^2\texttt{fold}(\texttt{bcirc}(\mathcal{C} * \mathcal{C}^T) \cdot \texttt{unfold}(\mathcal{I})) \\
&= \rho^2\mathcal{C} * \mathcal{C}^T. \qquad \square
\end{aligned}$$

Let the tensor $\vec{\mathcal{E}}_\rho \in \mathbb{R}^{\ell\times 1\times n}$ be defined as in Proposition 5.1 and let the tensor $\mathcal{C}$ be of compatible size. The entries of the tensor

$$\vec{\mathcal{E}} := \mathcal{C} * \vec{\mathcal{E}}_\rho \tag{5.44}$$

simulate noise with covariance tensor $\rho^2\mathcal{C} * \mathcal{C}^T$. We will use $\vec{\mathcal{E}}$ to simulate noise in $\vec{\mathcal{B}} \in \mathbb{R}^{\ell\times 1\times n}$, i.e., $\vec{\mathcal{B}} = \vec{\mathcal{B}}_{\text{true}} + \vec{\mathcal{E}}$; cf. (1.2). Introduce the scaled covariance tensor for $\vec{\mathcal{E}}$,

$$\mathcal{M} = \mathcal{C} * \mathcal{C}^T, \tag{5.45}$$

where we assume that the tensor (5.45) is positive definite. This is the generic situation. We refer to the quotient

$$\widetilde{\delta} := \frac{\|\vec{\mathcal{E}}\|_{\mathcal{M}^{-1}}}{\|\vec{\mathcal{B}}_{\rm true}\|_F} = \frac{\|\vec{\mathcal{E}}_\rho\|_F}{\|\vec{\mathcal{B}}_{\rm true}\|_F} \tag{5.46}$$

as the *noise level* of the error $\vec{\mathcal{E}}$ in $\vec{\mathcal{B}}$. Equality of the right-hand side of (5.46) holds since

$$\begin{aligned} \|\vec{\mathcal{E}}\|_{\mathcal{M}^{-1}}^2 &= \|\mathcal{C} * \vec{\mathcal{E}}_\rho\|_{\mathcal{M}^{-1}}^2 = \left(\vec{\mathcal{E}}_\rho^T * \mathcal{C}^T * \mathcal{M}^{-1} * \mathcal{C} * \vec{\mathcal{E}}_\rho\right)_{(:,:,1)} \\ &= \left(\vec{\mathcal{E}}_\rho^T * \vec{\mathcal{E}}_\rho\right)_{(:,:,1)} = \|\vec{\mathcal{E}}_\rho\|_F^2. \end{aligned}$$

In the computed examples below, we prescribe the noise level (5.46) and adjust ρ in the noise tensor $\vec{\mathcal{E}}_\rho$ to obtain a tensor (5.44) that corresponds to the desired noise level. Specifically, since $\|\vec{\mathcal{E}}_\rho\|_F = \rho\|\vec{\mathcal{E}}_1\|_F$, we can adjust $\rho > 0$ to obtain a "noise tensor" $\vec{\mathcal{E}}_\rho$ of a specified noise level. Knowledge of the noise level allows us to apply the discrepancy principle to determine the regularization parameter(s) as described in the previous sections.

In actual applications, the tensor $\mathcal{M}$ or an estimate thereof often are known and typically are symmetric positive definite. In our numerical experiments, we let

$$\mathcal{M} = \widetilde{\mathcal{L}}_1^T * \widetilde{\mathcal{L}}_1 + \omega\mathcal{I}, \tag{5.47}$$

with $\omega > 0$. Here $\widetilde{\mathcal{L}}_1$ is a tensor, whose first frontal slice is the upper bidiagonal matrix

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

and the remaining frontal slices $\widetilde{\mathcal{L}}_1^{(i)} \in \mathbb{R}^{\ell\times\ell}$, $i = 2, 3, \ldots, n$, are zero matrices. We compute the tensor Cholesky factorization of (5.47) by Algorithm 2 to obtain the Cholesky factor $\mathcal{R}$ and generate the noise tensor by

$$\vec{\mathcal{E}} = \mathcal{R}^T * \vec{\mathcal{E}}_\rho.$$

Thus, equation (5.45) holds with $\mathcal{C} = \mathcal{R}^T$. We adjust $\rho > 0$ to achieve a specified noise level as described above.

The SPD tensors $\mathcal{D}_\gamma$ defined next are used as regularization tensor $\mathcal{L}$ in the Tikhonov minimization problems (1.3) and (1.10). Let

$$\mathcal{D}_\gamma = \frac{1}{4}(\widetilde{\mathcal{L}}_2^T * \widetilde{\mathcal{L}}_2 + \alpha\mathcal{I}), \quad \gamma \in \{1, 2\}, \tag{5.48}$$

where $\alpha > 0$ and the tensor $\widetilde{\mathcal{L}}_2 \in \mathbb{R}^{m\times m\times n}$ has the tridiagonal matrix

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

as its first frontal slice, and the remaining frontal slices $\widetilde{\mathcal{L}}_2^{(i)} \in \mathbb{R}^{m\times m}$, $i = 2, 3, \ldots, n$, are zero matrices. We do not require Cholesky factorization of the tensor (5.48).

The quality of restorations obtained for $\mathcal{L} = \mathcal{I}$ and $\mathcal{L} = \mathcal{D}_\gamma$ in the computed examples is compared. We use the parameters $\omega = 0.2$ in (5.47) and $\alpha = 3$ in (5.48) in most examples. The influence on the computations of these parameters is discussed in Example 5.1. In all but the last example, we use $\gamma = 1$ in (5.48). The last example compares this choice to $\gamma = 2$.

The symbol "-" in the tables indicates that the solution method solves several subproblems and carries out different numbers of bidiagonalization steps for the subproblems or uses several regularization parameters that take on different values. We determine the regularization parameter(s) by

the bisection method over a specified interval using the discrepancy principle with safety parameter η. Specifically, we use the interval $[10^1, 10^8]$ and $\eta = 1.01$ in Example 5.2, and the interval $[10^1, 10^7]$ and $\eta = 1.1$ in Examples 5.1 and 5.3.

In all examples, the tensor $\mathcal{A}$ is a blurring operator that is constructed by using the function `blur` from [16]. We use the `squeeze` and `twist` operators described in [18], and the `multi_squeeze` and `multi_twist` operators defined in [27] to store tensors in the desired format.

This section compares many methods. To make it easier for a reader to follow the discussion, the names of the weighted global methods begin with "WG" or "WGG".

Example 5.1. (Color image restoration.) This example illustrates the performance of the W-tGKT$_p$, WG-tGKT$_p$, and WGG-tGKT methods when applied to the restoration of the `peppers` image of size $300 \times 300 \times 3$ shown in Figure 1 (left). The image is stored in the RGB format, with each frontal slice corresponding to a color. The blurring tensor $\mathcal{A} \in \mathbb{R}^{300\times300\times300}$ is generated by the MATLAB commands

$$z = [\texttt{exp}(-([\texttt{0} : \texttt{band} - \texttt{1}].^2)/(2\sigma^2)), \texttt{zeros}(\texttt{1}, \texttt{N} - \texttt{band})], \tag{5.49}$$

$$A = \frac{1}{\sigma\sqrt{2\pi}}\texttt{toeplitz}(z), \quad \mathcal{A}^{(i)} = A(i,1)A, \quad i = 1, 2, \ldots, 300, \quad \sigma = 3, \quad \text{and} \quad \texttt{band} = 12.$$

Let the true `peppers` image (assumed to be unknown) be stored as $\mathcal{X}_{\text{true}} \in \mathbb{R}^{300\times3\times300}$ by using the `multi_twist` operator. The blurred and noisy `peppers` image is generated as $\mathcal{B} = \mathcal{A} * \mathcal{X}_{\text{true}} + \mathcal{E}$, where $\mathcal{E}$ is the noise tensor (5.44). This image is displayed in Figure 1 (right) for $\widetilde{\delta} = 10^{-3}$ by using the `multi_squeeze` operator.

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

The dependence of the performance of the W-tGKT$_p$, WG-tGKT$_p$, and WGG-tGKT methods on the choice of $\omega > 0$ for $\alpha = 3$, and on the choice of $\alpha > 0$ for $\omega = 0.2$, is illustrated by Figures 2-3 for $\widetilde{\delta} = 10^{-2}$. We see from Figure 2 that smaller values of ω often give higher PSNR values of the restored image and result in higher CPU time requirement, whereas Figure 3 shows that smaller α values often result in smaller PSNR and faster computations. These observations inform our choices of ω and α in all computed examples. Recall that the parameter ω affects the properties of the noise tensor $\mathcal{E}$, while α determines the regularization tensor $\mathcal{L} = \mathcal{D}_1$.

Figure 4 shows restored images determined by the W-tGKT$_p$ and WGG-tGKT methods for $\mathcal{L}$ defined by (5.48) with $\gamma = 1$ and $\widetilde{\delta} = 10^{-3}$, and Table 2 shows PSNR values and relative errors for each method as well as the CPU time required for the computations. Table 2 compares the performance of the W-tGKT$_p$, WG-tGKT$_p$, and WGG-tGKT methods when applied to (1.10) and to the minimization problem

$$\min_{\mathcal{X}\in\mathbb{R}^{m\times p\times n}} \{\|\mathcal{A} * \mathcal{X} - \mathcal{B}\|_F^2 + \mu^{-1}\|\mathcal{X}\|_{\mathcal{L}^{-1}}^2\}. \tag{5.50}$$

We see from Table 2 that the use of weighted Frobenius norm in the fidelity term of (1.10) is more appropriate when the error tensor $\mathcal{E}$ that simulates the noise in $\mathcal{B}$ is of the form (5.44) than the standard (unweighted) Frobenius norm in (5.50). Independently of the choice of $\mathcal{L}$ and noise level, the W-tGKT$_p$ method determines restorations of the worst quality for (1.10), and of the highest quality for (5.50). This method does not involve flattening and works separately with

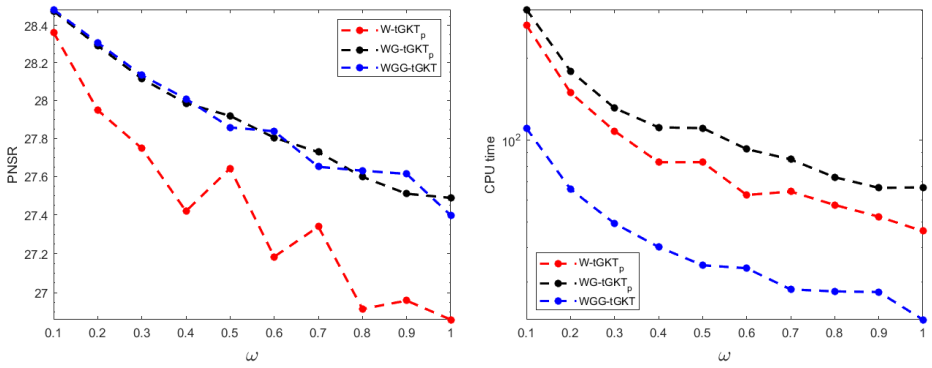

Figure 2: PSNR (left) and CPU time (right) for different ω values.

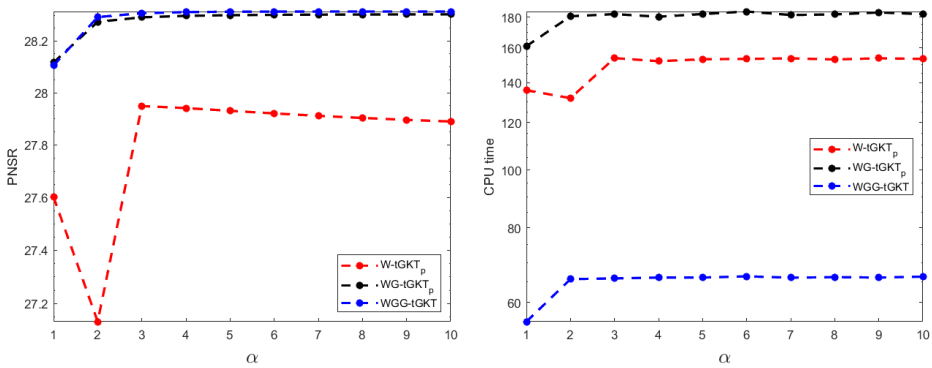

Figure 3: PSNR (left) and CPU time (right) for different α values.

Figure 4: Reconstructed images by the W-tGKT$_p$ method (left) and the WGG-tGKT method after 67 iterations (right) for $\widetilde{\delta} = 10^{-3}$.

each lateral slice of the data tensor $\mathcal{B}$. For both choices of regularization tensor $\mathcal{L}$ in (1.10), the WG-tGKT$_p$ and WGG-tGKT methods give restorations of the highest quality for $\widetilde{\delta} = 10^{-3}$ and $\widetilde{\delta} = 10^{-2}$, respectively. Among all methods considered for (5.50), the WGG-tGKT method results in restorations of the worst quality for both noise levels and for all choices of $\mathcal{L}$. This method requires less CPU time than the other methods for both minimization problems (1.10) and (5.50) because it works with the whole data tensor $\mathcal{B}$ at a time.

Example 5.2. (Medical imaging restoration.) We consider the restoration of an MRI image from MATLAB, and compare the weighted Golub-Kahan-Tikhonov (W-GKT), W-tGKT, and WG-tGKT methods applied to the solution of (1.1). The W-GKT method first reduces (1.1) to an equivalent problem involving a matrix and a vector, then applies the `gen_GKB` process described in [9] to determine an approximate solution of (1.1). Our implementation of the W-GKT method is analogous to the implementations of the W-tGKT and WG-tGKT methods. The (unknown) true

The performance of the W-tGKT$_p$, WG-tGKT$_p$, and WGG-tGKT methods when applied to (1.10).

$\widetilde{\delta}$	$\mathcal{L}$	Method	k	μ_k	PSNR	Relative error	CPU time (secs)
10^{-3}	$\mathcal{D}_1$	W-tGKT$_p$	-	-	30.60	$5.5508 \cdot 10^{-2}$	$3.20 \cdot 10^3$
		WG-tGKT$_p$	-	-	30.66	$5.5111 \cdot 10^{-2}$	$4.35 \cdot 10^3$
		WGG-tGKT	67	$2.45 \cdot 10^4$	30.65	$5.5176 \cdot 10^{-2}$	$1.55 \cdot 10^3$
	$\mathcal{I}$	W-tGKT$_p$	-	-	30.61	$5.5433 \cdot 10^{-2}$	$3.16 \cdot 10^3$
		WG-tGKT$_p$	-	-	30.64	$5.5262 \cdot 10^{-2}$	$5.12 \cdot 10^3$
		WGG-tGKT	72	$2.41 \cdot 10^4$	30.62	$5.5356 \cdot 10^{-2}$	$1.76 \cdot 10^3$
10^{-2}	$\mathcal{D}_1$	W-tGKT$_p$	-	-	27.95	$7.5297 \cdot 10^{-2}$	$1.56 \cdot 10^2$
		WG-tGKT$_p$	-	-	28.29	$7.2381 \cdot 10^{-2}$	$1.88 \cdot 10^2$
		WGG-tGKT	14	$8.92 \cdot 10^2$	28.31	$7.2248 \cdot 10^{-2}$	$6.89 \cdot 10^1$
	$\mathcal{I}$	W-tGKT$_p$	-	-	27.79	$7.6715 \cdot 10^{-2}$	$1.34 \cdot 10^2$
		WG-tGKT$_p$	-	-	28.31	$7.2265 \cdot 10^{-2}$	$1.86 \cdot 10^2$
		WGG-tGKT	14	$9.39 \cdot 10^2$	28.31	$7.2214 \cdot 10^{-2}$	$6.85 \cdot 10^1$

The performance of the W-tGKT$_p$, WG-tGKT$_p$, and WGG-tGKT methods when applied to (5.50).

$\widetilde{\delta}$	$\mathcal{L}$	Method	k	μ_k	PSNR	Relative error	CPU time (secs)
10^{-3}	$\mathcal{D}_1$	W-tGKT$_p$	-	-	30.30	$5.7421 \cdot 10^{-2}$	$8.44 \cdot 10^2$
		WG-tGKT$_p$	-	-	30.15	$5.8460 \cdot 10^{-2}$	$9.93 \cdot 10^2$
		WGG-tGKT	32	$3.91 \cdot 10^4$	30.13	$5.8565 \cdot 10^{-2}$	$3.48 \cdot 10^2$
	$\mathcal{I}$	W-tGKT$_p$	-	-	30.30	$5.7460 \cdot 10^{-2}$	$8.20 \cdot 10^2$
		WG-tGKT$_p$	-	-	30.13	$5.8601 \cdot 10^{-2}$	$1.14 \cdot 10^3$
		WGG-tGKT	34	$3.98 \cdot 10^4$	30.11	$5.8723 \cdot 10^{-2}$	$4.03 \cdot 10^2$
10^{-2}	$\mathcal{D}_1$	W-tGKT$_p$	-	-	27.43	$7.9983 \cdot 10^{-2}$	$4.25 \cdot 10^1$
		WG-tGKT$_p$	-	-	27.24	$8.1692 \cdot 10^{-2}$	$4.99 \cdot 10^1$
		WGG-tGKT	7	$1.43 \cdot 10^3$	27.20	$8.2053 \cdot 10^{-2}$	$1.69 \cdot 10^1$
	$\mathcal{I}$	W-tGKT$_p$	-	-	27.41	$8.0094 \cdot 10^{-2}$	$3.75 \cdot 10^1$
		WG-tGKT$_p$	-	-	27.32	$8.0918 \cdot 10^{-2}$	$5.39 \cdot 10^1$
		WGG-tGKT	7	$1.22 \cdot 10^3$	27.21	$8.2004 \cdot 10^{-2}$	$1.69 \cdot 10^1$

Table 2: Results for W-tGKT$_p$, WG-tGKT$_p$, and WGG-tGKT methods when applied to the restoration of `peppers` image.

MRI image is shown on the left-hand side of Figure 5.

The frontal slices $\mathcal{A}^{(i)} \in \mathbb{R}^{256\times256}$ of $\mathcal{A} \in \mathbb{R}^{256\times256\times256}$ are generated by using a modified form of the function `blur` from [16] with $\sigma = 4$ and `band` $= 7$. Specifically, the blurring tensor $\mathcal{A}$ is generated with the MATLAB commands

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

where z is defined in (5.49).

We store the true MRI image of size 256×256 as a tensor $\vec{\mathcal{X}}_{\text{true}} \in \mathbb{R}^{256\times1\times256}$ by using the `twist` operator. The blurred but noise-free image $\vec{\mathcal{B}}_{\text{true}} \in \mathbb{R}^{256\times1\times256}$ is generated by $\vec{\mathcal{B}}_{\text{true}} = \mathcal{A} * \vec{\mathcal{X}}_{\text{true}}$, and the blur- and noise-contaminated image $\vec{\mathcal{B}} = \vec{\mathcal{B}}_{\text{true}} + \vec{\mathcal{E}}$ is displayed on the right-hand side of Figure 5 by using the `squeeze` operator, where $\vec{\mathcal{E}}$ is a noise tensor analogous to (5.44).

The restored images determined by the W-tGKT and WG-tGKT methods are displayed in Figure 6 for the noise level $\widetilde{\delta} = 10^{-3}$. Table 3 illustrates that the performance of the W-GKT, W-tGKT, and WG-tGKT methods depends on the choice of the tensor $\mathcal{L}$ and on the noise level. Independently of the choice of $\mathcal{L}$, the W-tGKT method, which does not involve flattening, yields restorations of the highest quality for $\widetilde{\delta} = 10^{-3}$. The WG-tGKT method gives restorations of higher quality than the W-GKT method for $\widetilde{\delta} = 10^{-3}$. The latter method requires less CPU time than the former and is the fastest among the methods considered for both noise levels. The W-GKT and WG-tGKT methods yield restorations of almost the same quality for $\widetilde{\delta} = 10^{-2}$, and require the same number of iterations for both noise levels independently of the choice of $\mathcal{L}$. These

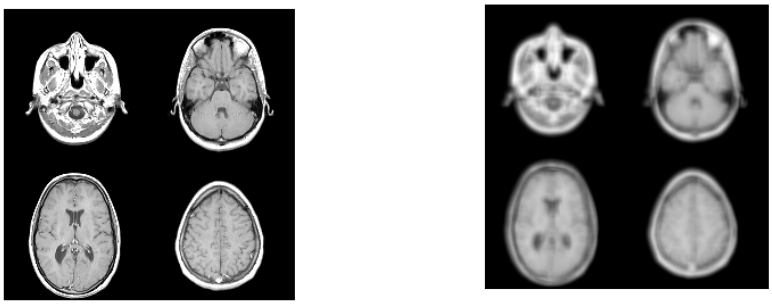

Figure 5: True image (left), blurred noisy image (right) for $\widetilde{\delta} = 10^{-3}$.

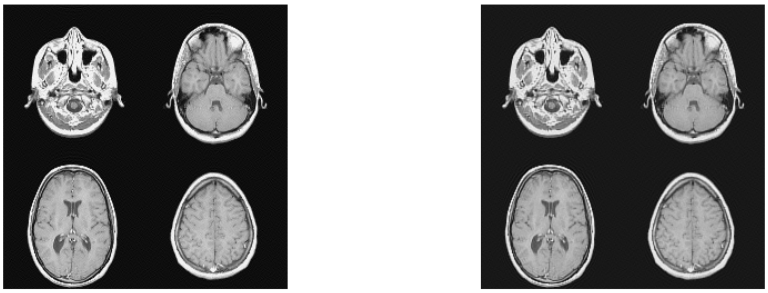

Figure 6: Restored images determined by the W-tGKT method after 28 iterations (left) and the WG-tGKT method after 107 iterations (right) for $\widetilde{\delta} = 10^{-3}$.

observations are based on the PSNR-values and relative errors shown in Table 3. Regardless of the choice of $\mathcal{L}$ and the noise level, the W-tGKT method requires the least number of iterations, while the WG-tGKT method is the slowest.

In summary, using the regularization tensor (5.48) increases the quality of the computed restorations slightly and reduces the number of iterations required by all methods in our comparison. The method W-tGKT gives the most accurate restorations for the smaller noise level.

$\widetilde{\delta}$	$\mathcal{L}$	Method	k	μ_k	PSNR	Relative error	CPU time (secs)
10^{-3}	$\mathcal{D}_1$	W-GKT	107	$9.07 \cdot 10^5$	32.52	$5.4448 \cdot 10^{-2}$	$3.23 \cdot 10^2$
		W-tGKT	28	$1.85 \cdot 10^5$	34.86	$4.1608 \cdot 10^{-2}$	$2.38 \cdot 10^2$
		WG-tGKT	107	$2.44 \cdot 10^6$	32.56	$5.4250 \cdot 10^{-2}$	$2.37 \cdot 10^3$
	$\mathcal{I}$	W-GKT	138	$3.85 \cdot 10^5$	32.19	$5.6571 \cdot 10^{-2}$	$3.20 \cdot 10^2$
		W-tGKT	38	$2.60 \cdot 10^5$	34.81	$4.1826 \cdot 10^{-2}$	$3.56 \cdot 10^2$
		WG-tGKT	138	$3.97 \cdot 10^5$	32.19	$5.6552 \cdot 10^{-2}$	$3.75 \cdot 10^3$
10^{-2}	$\mathcal{D}_1$	W-GKT	25	$1.04 \cdot 10^4$	24.74	$1.3338 \cdot 10^{-1}$	$1.74 \cdot 10^1$
		W-tGKT	10	$3.10 \cdot 10^4$	23.32	$1.5708 \cdot 10^{-1}$	$2.98 \cdot 10^1$
		WG-tGKT	25	$1.07 \cdot 10^4$	24.74	$1.3338 \cdot 10^{-1}$	$1.24 \cdot 10^2$
	$\mathcal{I}$	W-GKT	29	$8.74 \cdot 10^3$	24.74	$1.3347 \cdot 10^{-1}$	$1.46 \cdot 10^1$
		W-tGKT	12	$3.87 \cdot 10^4$	23.79	$1.4901 \cdot 10^{-1}$	$3.66 \cdot 10^1$
		WG-tGKT	29	$8.85 \cdot 10^3$	24.73	$1.3348 \cdot 10^{-1}$	$1.64 \cdot 10^2$

Table 3: Results for the W-GKT, W-tGKT, and WG-tGKT methods when applied to the restoration of the `MRI` image.

Example 5.3. (Video restoration.) This example is concerned with the restoration of the first four consecutive frames of the `Xylophone` video from MATLAB and compares the performance of the

W-tGKT$_p$, WG-tGKT$_p$, and WGG-tGKT methods for the regularization tensors $\mathcal{L} = \mathcal{I}$ and for $\mathcal{L}$ defined by (5.48) with $\gamma = 1$ or $\gamma = 2$. Each video frame is in `MP4` format and has 240×240 pixels. The blurring operator $\mathcal{A} \in \mathbb{R}^{240\times240\times240}$ is generated similarly as in Example 5.1 with $\sigma = 2.5$ and `band` $= 12$.

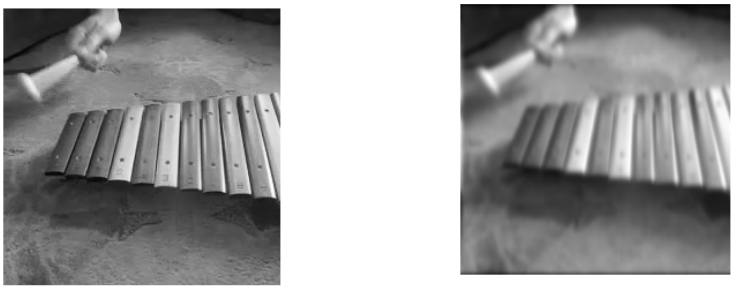

Figure 7: True fourth `video` frame (left), and blurred and noisy fourth `video` frame (right) for $\widetilde{\delta} = 10^{-3}$.

Figure 8: Restored fourth `video` frame by the W-tGKT$_p$ method (left) and the WGG-tGKT method after 50 iterations (right) for $\widetilde{\delta} = 10^{-3}$.

The first four blur- and noise-free frames are stored as a tensor $\mathcal{X}_{\text{true}} \in \mathbb{R}^{240\times4\times240}$ using the `multi_twist` operator and are blurred by the tensor $\mathcal{A}$. We generated the blur- and noise-contaminated frames as $\mathcal{B} = \mathcal{A} * \mathcal{X}_{\text{true}} + \mathcal{E}$, where $\mathcal{E}$ is a noise tensor defined by (5.44). The original fourth frame, and the corresponding blurred and noisy fourth frame are displayed in Figure 7 by using the `squeeze` operator. The restored fourth frames determined by the W-tGKT$_p$ and WGG-tGKT methods are shown in Figure 8 for $\widetilde{\delta} = 10^{-3}$ and $\mathcal{L} = \mathcal{D}_1$. Comparing the PSNR values and relative errors, as well as the CPU times, displayed in Table 4, the W-tGKT$_p$ method, independently of the choice of $\mathcal{L}$, yields restorations of the highest and of the least quality for $\widetilde{\delta} = 10^{-3}$ and $\widetilde{\delta} = 10^{-2}$, respectively. The WG-tGKT$_p$ method is seen to be the slowest and gives restorations of the highest quality for all choices of $\mathcal{L}$ for $\widetilde{\delta} = 10^{-2}$. The WGG-tGKT method is fastest for all choices of $\mathcal{L}$ and noise levels. Generally, the use of $\mathcal{L} = \mathcal{D}_2$ results in faster execution and gives restorations of higher quality for both noise levels than the other regularization tensors in our comparison.

6 Conclusion

This paper extends the generalized Golub-Kahan bidiagonalization process described in [9] for matrices to third order tensors using a t-product. This results in the weighted t-product Golub-Kahan bidiagonalization (W-tGKB) process. Global versions of the latter process also are considered, namely, the weighted global t-product Golub-Kahan bidiagonalization (WG-tGKB) and the weighted generalized global t-product Golub-Kahan bidiagonalization (WGG-tGKB) processes. The W-tGKB process does not involve flattening, but the global methods do. Only a few steps of

$\widetilde{\delta}$	$\mathcal{L}$	Method	k	μ_k	PSNR	Relative error	CPU time (secs)
10^{-3}	$\mathcal{D}_1$	W-tGKT$_p$	-	-	34.03	$4.1173 \cdot 10^{-2}$	$8.22 \cdot 10^2$
		WG-tGKT$_p$	-	-	34.03	$4.1185 \cdot 10^{-2}$	$1.25 \cdot 10^3$
		WGG-tGKT	50	$3.32 \cdot 10^4$	34.03	$4.1193 \cdot 10^{-2}$	$3.47 \cdot 10^2$
	$\mathcal{I}$	W-tGKT$_p$	-	-	34.11	$4.0797 \cdot 10^{-2}$	$8.24 \cdot 10^2$
		WG-tGKT$_p$	-	-	34.11	$4.0829 \cdot 10^{-2}$	$1.42 \cdot 10^3$
		WGG-tGKT	54	$2.12 \cdot 10^4$	34.11	$4.0835 \cdot 10^{-2}$	$4.01 \cdot 10^2$
	$\mathcal{D}_2$	W-tGKT$_p$	-	-	34.19	$4.0441 \cdot 10^{-2}$	$6.16 \cdot 10^2$
		WG-tGKT$_p$	-	-	34.14	$4.0648 \cdot 10^{-2}$	$1.12 \cdot 10^3$
		WGG-tGKT	48	$1.42 \cdot 10^4$	34.15	$4.0632 \cdot 10^{-2}$	$3.20 \cdot 10^2$
10^{-2}	$\mathcal{D}_1$	W-tGKT$_p$	-	-	31.01	$5.8289 \cdot 10^{-2}$	$3.95 \cdot 10^1$
		WG-tGKT$_p$	-	-	31.38	$5.5858 \cdot 10^{-2}$	$7.12 \cdot 10^1$
		WGG-tGKT	12	$7.44 \cdot 10^2$	31.38	$5.5860 \cdot 10^{-2}$	$2.01 \cdot 10^1$
	$\mathcal{I}$	W-tGKT$_p$	-	-	31.00	$5.8410 \cdot 10^{-2}$	$3.68 \cdot 10^1$
		WG-tGKT$_p$	-	-	31.47	$5.5285 \cdot 10^{-2}$	$7.01 \cdot 10^1$
		WGG-tGKT	12	$8.71 \cdot 10^2$	31.47	$5.5290 \cdot 10^{-2}$	$1.98 \cdot 10^1$
	$\mathcal{D}_2$	W-tGKT$_p$	-	-	31.52	$5.5006 \cdot 10^{-2}$	$4.14 \cdot 10^1$
		WG-tGKT$_p$	-	-	31.70	$5.3875 \cdot 10^{-2}$	$7.51 \cdot 10^1$
		WGG-tGKT	11	$3.65 \cdot 10^3$	31.64	$5.4267 \cdot 10^{-2}$	$1.98 \cdot 10^1$

Table 4: Results for the W-tGKT$_p$, WG-tGKT$_p$, and WGG-tGKT methods when applied to the restoration of gray-scale `video` frames.

the bidiagonalization processes are required to solve the weighted Tikhonov regularization problems of our examples. This is typical for many image and video restoration problems. The use of a regularization tensor $\mathcal{L} \neq \mathcal{I}$ often results in higher quality restorations than when $\mathcal{L} = \mathcal{I}$.

The weighted t-product Golub-Kahan-Tikhonov (W-tGKT) and weighted global t-product Golub-Kahan-Tikhonov (WG-tGKT) regularization methods for the approximate solution of (1.3) are considered. These methods are based on the W-tGKB and WG-tGKB processes, respectively. Independently of the choices of $\mathcal{L}$ considered, the W-tGKT method, which does not involve flattening, yields the best or near-best quality restorations for 0.1% noise level.

The W-tGKT and WG-tGKT methods also are applied p times to determine an approximate solution of (1.10). This leads to the W-tGKT$_p$ and WG-tGKT$_p$ methods. The weighted generalized global t-product Golub-Kahan-Tikhonov (WGG-tGKT) method for (1.10) is also discussed. This method differs from the W-tGKT$_p$ and WG-tGKT$_p$ methods in that it uses the WGG-tGKB process and works with a large amount of data at a time.

The WGG-tGKT method is the fastest, while the WG-tGKT$_p$ method is the slowest, independently of the choice of $\mathcal{L}$ and noise levels. Both methods involve flattening since they require additional product definition to the t-product. Generally, working with one lateral slice of the data tensor at a time is seen to give restorations of higher quality than working with all lateral slices simultaneously.

Acknowledgments

We would like to thank a referee for carefully reading this paper and for comments. Research by LR was supported in part by NSF grant DMS-1720259.

References

[1] M. Oriel, Generalized Golub-Kahan bidiagonalization and stopping criteria, SIAM J. Matrix Anal. Appl., 34 (2013), pp. 571–592.

[2] M. Arioli and D. Orban, Iterative methods for symmetric quasi-definite linear systems Part I: Theory. Cahier du GERAD G-2013-32. Montréal, Canada: GERAD, Montréal, QC, 2013.

[3] S. R. Arridge, M. M. Betcke, and L. Harhanen, Iterated preconditioned LSQR method for inverse problems on unstructured grids, Inverse Problems, 30 (2014), Art. 075009.

[4] S. J. Benbow, Solving generalized least-squares problems with LSQR, SIAM J. Matrix Anal. Appl., 21 (1999), pp. 166–177.

[5] 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.

[6] 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.

[7] F. P. A. Beik, A. El Ichi, K. Jbilou, and R. Sadaka, Tensor extrapolation methods with applications, Numer. Algorithms, 87 (2021), pp. 1421–1444.

[8] Å. Björck, Numerical Methods in Matrix Computations, Springer, Cham, 2015.

[9] J. Chung and A. Saibaba, Generalized hybrid iterative methods for large-scale Bayesian inverse problems, SIAM J. Sci. Comput., 39 (2007), pp. S24–S46.

[10] J. Chung, A. Saibaba, M. Brown, and E. Westman, Efficient generalized Golub-Kahan based methods for dynamic inverse problems. Inverse Problems, 34 (2018), Art. 024005.

[11] 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

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

[13] A. El Ichi, M. El Guide, and K. Jbilou, Discrete cosine transform LSQR and GMRES methods for multidimensional ill-posed problems, March 2021. https://arxiv.org/pdf/2103.11847.pdf

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

[15] 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.

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

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

[18] 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.

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

[20] S. Kindermann, 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.

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

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

[23] D. Orban and M. Arioli, Iterative Solution of Symmetric Quasi-Definite Linear Systems, SIAM, Philadelphia, 2017.

[24] A. K. Saibaba, J. Chung, and K. Petroske, Efficient Krylov subspace methods for uncertainty quantification in large Bayesian linear inverse problems, Numer. Linear Algebra Appl., 27 (2020), Art. e2325.

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

[26] L. Reichel, and U. O. Ugwu, Tensor Krylov subspace methods with an invertible linear transform product applied to image processing, Appl. Numer. Math., 166 (2021), pp. 186–207.

[27] L. Reichel and U. O. Ugwu, Tensor Golub-Kahan-Tikhonov methods applied to the solution of ill-posed problem with a t-product structure, Numer. Linear Algebra Appl., 29 (2022), e2412.

[28] L. Reichel and U. O. Ugwu, Tensor Arnoldi-Tikhonov and GMRES-type methods for ill-posed problem with t-product structure, J. Sci. Comput., 90 (2022), Art. 59.