**Journal of Mathematical Modeling**
Vol. 10, No. 1, 2022, pp. 21-37. Research Article

JMM

# Discrete cosine transform LSQR methods for multidimensional ill-posed problems

**Mohamed El Guide[†], Alaa El Ichi[‡], Khalide Jbilou[§*]**

[†]*Centre for Behavioral Economics and Decision Making(CBED), FGSES, Mohammed VI Polytechnic University, Green City, Morocco*
[‡]*Laboratoire de Mathmatiques, Informatique et Applications, Sécurité de l'Information LABMIA-SI, University Mohamed V, Rabat Morocco; University Littoral Cote d'Oplae, France*
[§]*LMPA, 50 rue F. Buisson, ULCO Calais, France; Mohammed VI Polytechnic University, Green City, Morocco*
*Email(s): aa@guilan.ac.ir, bb@guilan.ac.ir, jbilou@univ-littoral.fr*

**Abstract.** We propose new tensor Krylov subspace methods for ill-posed linear tensor problems such as color or video image restoration. Those methods are based on the tensor-tensor discrete cosine transform that gives fast tensor-tensor product computations. In particular, we will focus on the tensor discrete cosine versions of GMRES, Golub-Kahan bidiagonalisation and LSQR methods. The presented numerical tests show that the methods are very fast and give good accuracies when solving some linear tensor ill-posed problems.

*Keywords*: Discrete cosine product, Golub-Kahan bidiagonalisation, GMRES, LSQR, tensor Krylov subspaces.
*AMS Subject Classification 2010*: 65F10, 65F22.

## 1 Introduction

The aim of this paper is to solve the following tensor problem

$$\min_{\mathcal{X}} \|\mathscr{M}(\mathcal{X}) - \mathcal{C}\|_F, \tag{1}$$

where $\mathscr{M}$ is a linear operator that could be described as

$$\mathscr{M}(\mathcal{X}) = \mathcal{A} \star_c \mathcal{X} \text{ or } \mathscr{M}(\mathcal{X}) = \mathcal{A} \star_c \mathcal{X} \star_c \mathcal{B},$$

where $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ is a three mode tensor, $\mathcal{X} \in \mathbb{R}^{n_2 \times s \times n_3}$, $\mathcal{B} \in \mathbb{R}^{s \times s \times n_3}$ and $\mathcal{C} \in \mathbb{R}^{n_1 \times s \times n_3}$ are three mode tensors, and $\star_c$ is the cosine product to be also defined later. Applications of such problems arise

*Corresponding author.
Received: 7 April 2021/ Revised: 15 May 2021/ Accepted: 23 May 2021
DOI: 10.22124/jmm.2021.19303.1659

 

from signal processing [22], data mining [23], computer vision and so many other modern applications in machine learning. For large scale problems, we have to take advantage of the multidimensional structure to build rapid and robust iterative methods. Tensor Krylov subspace methods could be useful and very fast solvers for those tensor problems.

In the present paper, we will be interested in developing robust and fast iterative tensor Krylov-based subspace methods using tensor-tensor products such as the tensor cosine product [1]. In many applications such as in image or video processing, the obtained discrete problems are very ill-conditioned and we should add some regularization techniques such as the generalized cross validation method. Standard and global Krylov subspace methods are suitable when dealing with grayscale images, e.g, [2, 5, 7]. However, these methods might be time consuming to numerically solve problems related to multi channel images (e.g., color images, hyper-spectral images and videos).

In this paper, we will show that the tensor-tensor product between third-order tensors allows the application of the global iterative methods, such as the global Arnoldi and global Golub-Kahan algorithms. The tensor form of the proposed Krylov methods, together with using the fast cosine transform (DCT) to compute the $c$-product between third-order tensors can be efficiently implemented on many modern computers and allows to significantly reduce the overall computational complexity. It is also worth mentioning that our approaches can be naturally generalized to higher-order tensors in a recursive manner.

This paper is organized as follows. We shall first present in Section 2 some symbols and notations used throughout the paper. We also recall some definitions related to the cosine product between two tensors. In Section 3, we present some inexpensive approaches based on cosine global Krylov subspace methods combined with regularization techniques to solve the obtained ill-posed tensor problem (1) . Section 4 is dedicated to some numerical experiments.

## 2 Definitions and notations

A tensor is a multidimensional array of data. The number of indices of a tensor is called modes or ways. Notice that a scalar can be regarded as a zero mode tensor, first mode tensors are vectors and matrices are second mode tensor. The order of a tensor is the dimensionality of the array needed to represent it, also known as ways or modes. For a given $N$-mode (or order-$N$) tensor $\mathcal{X} \in \mathbb{R}^{n_1 \times n_2 \times n_3 \times \cdots \times n_N}$, the notation $x_{i_1,\ldots,i_N}$ (with $1 \leq i_j \leq n_j$ and $j = 1,\ldots,N$) stands for the element $(i_1,\ldots,i_N)$ of the tensor $\mathcal{X}$.

Fibers are the higher-order analogue of matrix rows and columns. A fiber is defined by fixing all the indexes except one. A matrix column is a mode-1 fiber and a matrix row is a mode-2 fiber. Third-order tensors have column, row and tube fibers. An element $c \in \mathbb{R}^{1\times 1\times n}$ is called a tubal-scalar of length $n$. More details are found in [18, 20].

In the present paper, we will consider only 3-order tensors and show how to use them in color image and video processing.

### 2.1 Discrete cosine transformation

In this subsection, we recall some definitions and properties of the discrete cosine transformation and the $c$-product. The discrete cosine transformation (DCT) plays a very important role in the definition of the

$c$-product of tensors. The DCT on a vector $v \in \mathbb{R}^n$ is defined by

$$\tilde{v} = C_n v \in \mathbb{R}^n,$$

where $C_n$ is the $n \times n$ discrete cosine transform matrix with entries

$$(C_n)_{ij} = \sqrt{\frac{2-\delta_{i1}}{n}} \cos\left(\frac{(i-1)(2j-1)\pi}{2n}\right), \quad 1 < i, j < n,$$

with $\delta_{i1} = 0$ if $i \neq 1$ and 1 if $i = 1$. It is known that the matrix $C_n$ is orthogonal, i.e,

$$C_n^T C_n = C_n C_n^T = I_n;$$

see [24]. Furthermore, for any vector $v \in \mathbb{R}^n$, the matrix vector multiplication $C_n v$ can be computed in $O(nlog(n))$ operations. Also, Ng et al. [24] showed that matrices which can be diagonalized by $C_n$ are some special Toeplitz-plus-Hankel matrices. In other words, we have

$$C_n\, \mathtt{th}(v)\, C_n^{-1} = \mathrm{Diag}(\tilde{v}),$$

where

$$\mathtt{th}(v) = \underbrace{\begin{pmatrix} v_1 & v_2 & \dots & v_n \\ v_2 & v_1 & \dots & v_3 \\ \vdots & \vdots & \ddots & \vdots \\ v_n & v_{n-1} & \dots & v_1 \end{pmatrix}}_{\text{Toeplitz}} + \underbrace{\begin{pmatrix} v_2 & \dots & v_n & 0 \\ \vdots & \iddots & \iddots & v_n \\ v_n & 0 & \dots & \vdots \\ 0 & v_n & \dots & v_2 \end{pmatrix}}_{\text{Hankel}},$$

and $\mathrm{Diag}(\tilde{v})$ is the diagonal matrix whose $i$-th diagonal element is $(\tilde{v})_i$.

## 2.2 Properties of the cosine product

In this subsection, we briefly review some concepts and notations, that play a central role for the elaboration of the tensor iterative methods based on the c-product; see [17] for more details on the c-product.

Let $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ be a third-order tensor, the operations `mat` and its reverse `ten` are defined by

$$\mathtt{mat}(\mathcal{A}) = \underbrace{\begin{pmatrix} A_1 & A_2 & \dots & A_n \\ A_2 & A_1 & \dots & A_3 \\ \vdots & \vdots & \ddots & \vdots \\ A_n & A_{n-1} & \dots & A_1 \end{pmatrix}}_{\text{Block Toeplitz}} + \underbrace{\begin{pmatrix} A_2 & \dots & A_n & 0 \\ \vdots & \iddots & \iddots & A_n \\ A_n & 0 & \dots & \vdots \\ 0 & A_n & \dots & A_2 \end{pmatrix}}_{\text{Block Hankel}} \in \mathbb{R}^{n_1 n_3 \times n_2 n_3},$$

where the matrices $A_i \in \mathbb{R}^{n_1 \times n_2}, i = 1, \dots, n_3$ are the frontal slices of the tensor $\mathcal{A}$. The reverse operation denoted by `ten` is such that

$$\mathtt{ten}(\mathtt{mat}(\mathcal{A})) = \mathcal{A}.$$

Let $\widetilde{\mathcal{A}}$ be the tensor obtained by applying the DCT on all the tubes of the tensor $\mathcal{A}$. With the MATLAB command `dct`, we have

$$\widetilde{\mathcal{A}} = \mathtt{dct}(\mathcal{A}, [\,], 3), \text{ and } \mathtt{idct}(\widetilde{\mathcal{A}}, [\,], 3) = \mathcal{A},$$

where idct denotes the inverse discrete cosine transform.

The transpose of a tensor $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ is a tensor of size $(n_2 \times n_1 \times n_3)$ denoted by $\mathcal{A}^T$ and such that the $i$-th frontal slice of $\widetilde{\mathcal{A}^T} = \texttt{dct}(\mathcal{A}^T, [\,], 3)$ is equal to $\mathcal{A}^{(i)T}$ for $i = 1, \ldots, n_3$. where $\mathcal{A}^{(i)}$ denotes the $i$-th frontal slice of $\widetilde{\mathcal{A}} = \texttt{dct}(\mathcal{A}, [\,], 3)$.

**Remark 1.** *Notice that the tensor $\widetilde{\mathcal{A}}$ can be computed by using the 3-mode product defined in [18] as follows:*

$$\widetilde{\mathcal{A}} = \mathcal{A} \times_3 M,$$

*where $M$ is the $n_3 \times n_3$ invertible matrix given by*

$$M = W^{-1} C_{n_3} (I + Z), \tag{2}$$

*and $C_{n_3}$ denotes the $n_3 \times n_3$ discrete cosine transform DCT matrix, $W = \text{diag}(C_{n_3}(:,1))$ is the diagonal matrix made of the first column of the DCT matrix, $Z$ is an $n_3 \times n_3$ circulant matrix which can be computed in* MATLAB *using the command* $\texttt{W} = \text{diag}(\text{ones}(\texttt{n3} - 1, 1), 1)$ *and $I$ the $n_3 \times n_3$ identity matrix; see [17] for more details.*

Let $\mathbf{A}$ be the matrix

$$\mathbf{A} = \begin{pmatrix} A^{(1)} & & & \\ & A^{(2)} & & \\ & & \ddots & \\ & & & A^{(n_3)} \end{pmatrix} \in \mathbb{R}^{n_3 n_1 \times n_3 n_2}, \tag{3}$$

where the matrices $A^{(i)}$'s are the frontal slices of the tensor $\widetilde{\mathcal{A}}$. The block matrix $\texttt{mat}(\mathcal{A})$ can also be block diagonalized using the DCT matrix and this gives

$$(C_{n_3} \otimes I_{n_1})\, \texttt{mat}(\mathcal{A})\, (C_{n_3}^T \otimes I_{n_2}) = \mathbf{A}.$$

**Definition 1.** *The **c-product** between two tensors $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ and $\mathcal{B} \in \mathbb{R}^{n_2 \times m \times n_3}$ is an $n_1 \times m \times n_3$ tensor given by:*

$$\mathcal{A} \star_c \mathcal{B} = \texttt{ten}(\texttt{mat}(\mathcal{A})\texttt{mat}(\mathcal{B})).$$

Notice that from the relation (3), we can show that the product $\mathcal{C} = \mathcal{A} \star_c \mathcal{B}$ is equivalent to $\mathbf{C} = \mathbf{A}\,\mathbf{B}$. The following algorithm allows us to compute, in an efficient way, the $c$-product of the tensors $\mathcal{A}$ and $\mathcal{B}$, see [17].

**Algorithm 1.** *Computing the c-product.*
***Inputs**: $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ and $\mathcal{B} \in \mathbb{R}^{n_2 \times m \times n_3}$*
***Output**: $\mathcal{C} = \mathcal{A} \star_c \mathcal{B} \in \mathbb{R}^{n_1 \times m \times n_3}$*

1. *Compute $\widetilde{\mathcal{A}} = \texttt{dct}(\mathcal{A}, [\,], 3)$ and $\widetilde{\mathcal{B}} = \texttt{dct}(\mathcal{B}, [\,], 3)$.*
2. *Compute each frontal slices of $\widetilde{\mathcal{C}}$ by $C^{(i)} = A^{(i)} B^{(i)}$.*
3. *Compute $\mathcal{C} = \texttt{idct}(\widetilde{\mathcal{C}}, [\,], 3)$.*

For the $c$-product, we have the following definitions and remarks:

**Definition 2.** *The identity tensor $\mathcal{I}_{n_1 n_1 n_3}$ is the tensor such that all frontal slice of $\widetilde{\mathcal{I}}_{n_1 n_1 n_3}$ is the identity matrix $I_{n_1 n_1}$. An $n_1 \times n_1 \times n_3$ tensor $\mathcal{A}$ is invertible, if there exists a tensor $\mathcal{B}$ of order $n_1 \times n_1 \times n_3$ such that*

$$\mathcal{A} \star_c \mathcal{B} = \mathcal{I}_{n_1 n_1 n_3} \qquad \text{and} \qquad \mathcal{B} \star_c \mathcal{A} = \mathcal{I}_{n_1 n_1 n_3}.$$

*In that case, we set $\mathcal{B} = \mathcal{A}^{-1}$. It is clear that $\mathcal{A}$ is invertible if and only if $\mathtt{mat}(\mathcal{A})$ is invertible. The inner scalar product is defined by*

$$\langle \mathcal{A}, \mathcal{B} \rangle = \sum_{i_1=1}^{n_1} \sum_{i_2=1}^{n_2} \sum_{i_3=1}^{n_3} a_{i_1 i_2 i_3} b_{i_1 i_2 i_3},$$

*and he corresponding norm is given by $\|\mathcal{A}\|_F = \sqrt{\langle \mathcal{A}, \mathcal{A} \rangle}$. An $n_1 \times n_1 \times n_3$ tensor $\mathcal{Q}$ is orthogonal if $\mathcal{Q}^T \star_c \mathcal{Q} = \mathcal{Q} \star_c \mathcal{Q}^T = \mathcal{I}_{n_1 n_1 n_3}$.*

**Remark 2.** *Another interesting way for computing the scalar product and the associated norm is as follows: $\langle \mathcal{A}, \mathcal{B} \rangle = \langle \mathbf{A}, \mathbf{B} \rangle$ and $\|\mathcal{A}\|_F = \|\mathbf{A}\|_F$, where the block diagonal matrix $\mathbf{A}$ is defined by (3).*

We now introduce the new $c$-diamond tensor-tensor product.

**Definition 3.** *Let $\mathcal{A} = [\mathcal{A}_1, \ldots, \mathcal{A}_p] \in \mathbb{R}^{n_1 \times ps \times n_3}$, where $\mathcal{A}_i \in \mathbb{R}^{n_1 \times s \times n_3}$, $i = 1, ..., p$ and let $\mathcal{B} = [\mathcal{B}_1, \ldots, \mathcal{B}_l] \in \mathbb{R}^{n_1 \times \ell s \times n_3}$ with $\mathcal{B}_j \in \mathbb{R}^{n_1 \times s \times n_3}$, $j = 1, ...\ell$. Then, the product $\mathcal{A}^T \diamondsuit \mathscr{B}$ is the $p \times \ell$ matrix given b:*

$$(\mathcal{A}^T \diamondsuit \mathscr{B})_{i,j} = \langle \mathcal{A}_i, \mathcal{B}_j \rangle.$$

## 3 Tensor discrete cosine global Krylov subspace methods

In this section, we propose iterative methods based on tensor cosine global Arnoldi and cosine global Golub Kahan bidiagonlization (cosine-GGKB), combined with Tikhonov regularization, to solve some discrete ill posed problems. We consider the following discrete ill-posed tensor equation

$$\mathcal{A} \star_c \mathcal{X} = \mathcal{C}, \quad \mathcal{C} = \widehat{\mathcal{C}} + \mathcal{N}, \tag{4}$$

where $\mathcal{A} \in \mathbb{R}^{n \times m \times p}$, $\mathcal{X}$, $\mathcal{N}$ (additive noise) and $\mathcal{C}$ are tensors in $\mathbb{R}^{n \times s \times p}$. In color image processing, $p = 3$, $\mathcal{A}$ represents the blurring tensor, $\mathcal{C}$ the blurry and noisy observed image, $\mathcal{X}$ is the image that we would like to restore and $\mathcal{N}$ is an unknown additive noise. Therefore, to stabilize the recovered image, regularization techniques are needed. There are several techniques to regularize the linear inverse problem given by equation (4); for the matrix case, see for example, [2,5,11,12]. All of these techniques stabilize the restoration process by adding a regularization term, depending on some priori knowledge of the unknown image. One of the most regularization method is due to Tikhonov and is given as follows

$$\min_{\mathcal{X}} \{ \|\mathcal{A} \star_c \mathcal{X} - \mathcal{C}\|_F^2 + \lambda \|\mathcal{X}\|_F^2 \}. \tag{5}$$

Many techniques for choosing a suitable value of $\lambda$ have been analysed and illustrated in the literature; see, e.g., [6, 11, 12, 28] and references therein. In this paper we will use the discrepancy principle and the Generalized Cross Validation (GCV) techniques.

### 3.1 The tensor discrete cosine GMRES

Let $\mathcal{A} \in \mathbb{R}^{n\times n\times p}$ and $\mathcal{V} \in \mathbb{R}^{n\times s\times p}$. We introduce the tensor Krylov subspace $\mathcal{TK}_m(\mathcal{A},\mathcal{V})$ associated to the cosine-product, defined for the pair $(\mathcal{A},\mathcal{V})$ as follows

$$\mathcal{TK}_m(\mathcal{A},\mathcal{V}) = \text{Tspan}\{\mathcal{V},\mathcal{A}\star_c\mathcal{V},\ldots,\mathcal{A}^{m-1}\star_c\mathcal{V}\} = \left\{\mathcal{Z}\in\mathbb{R}^{n\times s\times p}, \mathcal{Z} = \sum_{i=1}^{m}\alpha_i\left(\mathcal{A}^{i-1}\star_c\mathcal{V}\right)\right\},$$

where $\alpha_i \in \mathbb{R}$, $\mathcal{A}^{i-1}\star_c\mathcal{V} = \mathcal{A}^{i-2}\star_c\mathcal{A}\star_c\mathcal{V}$, for $i = 2,\ldots,m$ and $\mathcal{A}^0$ is the identity tensor. In the following algorithm, we define the Tensor cosine-global Arnoldi algorithm.

**Algorithm 2.** *Tensor discrete cosine Arnoldi.*

1. **Input.** $\mathcal{A}\in\mathbb{R}^{n\times n\times p}$, $\mathcal{V}\in\mathbb{R}^{n\times s\times p}$ *and the positive integer m.*
2. *Set* $\beta = \|\mathcal{V}\|_F$, $\mathcal{V}_1 = \dfrac{\mathcal{V}}{\beta}$.
3. *For* $j = 1,\ldots,m$*:*
   (a) $\mathcal{W} = \mathcal{A}\star_c\mathcal{V}_j$.
   (b) *for* $i = 1,\ldots,j$*:*
      i. $h_{i,j} = \langle\mathcal{V}_i,\mathcal{W}\rangle$.
      ii. $\mathcal{W} = \mathcal{W} - h_{i,j}\,\mathcal{V}_i$.
   (c) *End for.*
   (d) $h_{j+1,j} = \|\mathcal{W}\|_F$. *If* $h_{j+1,j} = 0$, *stop; else:*
   (e) $\mathcal{V}_{j+1} = \mathcal{W}/h_{j+1,j}$.
4. *End*

It is not difficult to show that after $m$ steps of Algorithm 2, the tensors $\mathcal{V}_1,\ldots,\mathcal{V}_m$ form an orthonormal basis of the tensor Krylov subspace $\mathcal{TK}_m(\mathcal{A},\mathcal{V})$. Let $\mathbb{V}_m$ be the $(n\times(sm)\times p)$ tensor with frontal slices $\mathcal{V}_1,\ldots,\mathcal{V}_m$ and let $\widetilde{H}_m$ be the $(m+1)\times m$ upper Hesenberg matrix whose elements are the $h_{i,j}$'s defined by Algorithm 2. Let $H_m$ be the matrix obtained from $\widetilde{H}_m$ by deleting its last row; $H_{.,j}$ will denote the $j$-th column of the matrix $H_m$ and $\mathcal{A}\star_c\mathbb{V}_m$ is the $(n\times(sm)\times p)$ tensor with frontal slices $\mathcal{A}\star_c\mathcal{V}_1,\ldots,\mathcal{A}\star_c\mathcal{V}_m$:

$$\mathbb{V}_m := [\mathcal{V}_1,\ldots,\mathcal{V}_m] \quad\text{and}\quad \mathcal{A}\star_c\mathbb{V}_m := [\mathcal{A}\star_c\mathcal{V}_1,\ldots,\mathcal{A}\star_c\mathcal{V}_m].$$

We introduce the product $\circledast$ defined by

$$\mathbb{V}_m\circledast y = \sum_{j=1}^{m} y_j\mathcal{V}_j,\;\; y = (y_1,\ldots,y_m)^T\in\mathbb{R}^m, \text{ and } \mathbb{V}_m\circledast H_m = \left[\mathbb{V}_m\circledast H_{.,1},\ldots,\mathcal{V}_m\circledast H_{.,m}\right].$$

With the above notations, we can easily prove the results of the following proposition.

**Proposition 1.** *Suppose that m steps of Algorithm 2 have been run. Then, the following statements hold:*

$$\begin{aligned}
\mathcal{A} \star_c \mathbb{V}_m &= \mathbb{V}_m \circledast H_m + h_{m+1,m}\left[\mathcal{O}_{n\times s\times p}, \ldots, \mathcal{O}_{n\times s\times p}, \mathcal{V}_{m+1}\right], \\
\mathcal{A} \star_c \mathbb{V}_m &= \mathbb{V}_{m+1} \circledast \widetilde{H}_m, \\
\mathbb{V}_m^T \diamondsuit \mathcal{A} \star_c \mathbb{V}_m &= H_m, \\
\mathbb{V}_{m+1}^T \diamondsuit \mathcal{A} \star_c \mathbb{V}_m &= \widetilde{H}_m, \\
\mathbb{V}_m^T \diamondsuit \mathbb{V}_m &= I_m, \\
\|\mathbb{V}_m \circledast y\|_F &= \|y\|_2,\ y \in \mathbb{R}^m,
\end{aligned}$$

*where $I_m$ the identity matrix and $\mathcal{O}_{n\times s\times p}$ is the tensor of size $(n \times s \times p)$ having all its entries equal to zero.*

In the sequel, we briefly present the tensor discrete cosine GMRES algorithm to solve the problem (5). Let $\mathcal{X}_0 \in \mathbb{R}^{n\times s\times p}$ be an arbitrary initial guess with the corresponding residual $\mathcal{R}_0 = \mathcal{C} - \mathcal{A} \star_c \mathcal{X}_0$. The aim of tensor cosine GMRES method is to find and approximate solution $\mathcal{X}_m$ approximating the exact solution $\mathcal{X}^*$ such that

$$\mathcal{X}_m = \mathcal{X}_0 + \mathbb{V}_m \circledast y,$$

where $y = y_{m,\lambda_m} \in \mathbb{R}^m$ solves the projected regularized minimization problem

$$\begin{aligned}
y_{m,\lambda_m} &= \arg\min_{y\in\mathbb{R}^m} \left( \|\beta e_1 - \widetilde{H}_m y\|_2^2 + \lambda_m^2 \|y\|_2^2 \right), \\
&= \arg\min_{y\in\mathbb{R}^m} \left\| \begin{pmatrix} \widetilde{H}_m \\ \lambda_m I_m \end{pmatrix} y - \begin{pmatrix} \beta e_1 \\ 0 \end{pmatrix} \right\|_2^2,
\end{aligned} \tag{6}$$

where $\beta = \|\mathcal{R}_0\|$ and $e_1$ the first canonical basis vector in $\mathbb{R}^{m+1}$. The minimizer $y_{m,\lambda_m}$ can also be computed as the solution of the following normal equations associated with (6)

$$\widetilde{H}_{m,\lambda_m} y = \widetilde{H}_m^T \beta e_1, \quad \widetilde{H}_{m,\lambda_m} = (\widetilde{H}_m^T \widetilde{C}_m + \lambda_m^2 I_m). \tag{7}$$

Note that since the Tikhonov problem (7) is now a matrix one with small dimension as $m$ is generally small, $\lambda_m$, can thereby be inexpensively computed by some techniques such as the GCV method [11] or the L-curve criterion [5, 7, 12]. In this paper we consider the generalized cross-validation (GCV) method to choosing the regularization parameter [11, 28]. We take advantage of the SVD decomposition of the low dimensional matrix $\widetilde{H}_m$ to obtain a more simple and computable expression of $GCV(\lambda_m)$. Consider the SVD decomposition $\widetilde{C}_k = U\Sigma V^T$. Then, the GCV function can be expressed as (see [28])

$$GCV(\lambda_m) = \frac{\displaystyle\sum_{i=1}^{m} \left( \frac{\widetilde{g}_i}{\sigma_i^2 + \lambda_m^2} \right)^2}{\left( \displaystyle\sum_{i=1}^{m} \frac{1}{\sigma_i^2 + \lambda_m^2} \right)^2}, \tag{8}$$

where $\sigma_i$ is the $i$th singular value of the matrix $\widetilde{H}_m$ and $\widetilde{g} = \beta_1 U^T e_1$. The restarted tensor discrete cosine GMRES algorithm is summarized as follows

**Algorithm 3.** *Restarted tensor discrete cosine GMRES (DC-GMRES(m)) method with Tikhonov regularization.*
**Input.** $\mathcal{A} \in \mathbb{R}^{n\times n\times p}$, $\mathcal{C}, \mathcal{X}_0 \in \mathbb{R}^{n\times s\times p}$, *an integer m for restarting, a maximum number of iterations* $Iter_{max}$ *and a tolerance* $tol > 0$.
**Output.** $\mathcal{X}_m \in \mathbb{R}^{n\times s\times p}$ *approximate solution of the system (1).*

1. *Set* $k = 0$ *and compute* $\mathcal{R}_0 = \mathcal{C} - \mathcal{A} \star_c \mathcal{X}_0$.
2. *Apply Algorithm 2 to the pair* $(\mathcal{A}, \mathcal{R}_0)$ *to compute* $\mathbb{V}_m$ *and* $\widetilde{H}_m$.
3. *Determine* $\lambda_m$ *as the parameter minimizing the GCV function given by (8).*
4. *Compute the regularized solution* $y_{m,\lambda_m}$ *of the problem (6).*
5. *Compute the approximate solution* $\mathcal{X}_m = \mathcal{X}_0 + \mathbb{V}_m \circledast y_{m,\lambda_m}$.
6. *If* $\|\mathcal{R}_m\|_F < tol$ *or* $k > itermax$, *stop, else set* $\mathcal{X}_0 = \mathcal{X}_m, k = k+1$ *and go to Step 4.*

### 3.2 The tensor discrete cosine Golub-Kahan method

We consider the tensor least squares problem

$$\min_{\mathcal{X}} \{\|\mathcal{A} \star_c \mathcal{X} - \mathcal{C}\|_F^2\}, \tag{9}$$

where $\mathcal{A} \in \mathbb{R}^{n\times \ell\times p}$ and $\mathcal{C} \in \mathbb{R}^{n\times s\times p}$. Instead of using the tensor cosine Arnoldi, we can use a dicrete cosine version of the tensor Lanczos process to generate a new basis that can be used for the projection. We will use the tensor Golub Kahan algorithm related to the c-product and summarized in Algorithm 4.

**Algorithm 4.** *The Tensor discrete cosine Bidiagonalisation Golub Kahan algorithm.*

1. **Input.** *The tensors* $\mathcal{A}$, $\mathcal{C}$ *and an integer m.*
2. *Set* $\beta_1 = \|\mathcal{C}\|_F$, $\alpha_1 = \|\mathcal{A}^T \star_c \mathcal{U}_1\|_F$, $\mathcal{U}_1 = \mathcal{C}/\beta_1$ *and* $\mathcal{V}_1 = (\mathcal{A}^T \star_c \mathcal{U}_1)/\alpha_1$.
3. *for* $j = 1, \ldots, m$:
   (a) $\widetilde{\mathcal{U}} = \mathcal{A} \star_c \mathcal{V}_j - \alpha_j \mathcal{U}_j$.
   (b) $\beta_{j+1} = \|\widetilde{\mathcal{U}}\|_F$.
   (c) $\mathcal{U}_{j+1} = \widetilde{\mathcal{U}}/\beta_{j+1}$.
   (d) $\widetilde{\mathcal{V}} = \mathcal{A}^T \star_c \mathcal{U}_{j+1} - \beta_{j+1}\mathcal{V}_j$.
   (e) $\alpha_{j+1} = \|\widetilde{\mathcal{V}}\|_F$.
   (f) $\mathcal{V}_{j+1} = \widetilde{\mathcal{V}}/\alpha_{j+1}$.

Let $\widetilde{C}_m$ be the upper bidiagonal $((m+1)\times m)$ matrix

$$\widetilde{C}_m = \begin{bmatrix} \alpha_1 & & & \\ \beta_2 & \alpha_2 & \ddots & \\ & \ddots & \ddots & \\ & & \beta_m & \alpha_m \\ & & & \beta_{m+1} \end{bmatrix},$$

and let $C_m$ be the $(m\times m)$ matrix obtain by deleting the last row of $\widetilde{C}_m$. We denote by $C_{:,j}$ the $j$-th column of the matrix $C_m$. Let $\mathbb{V}_m$ and $\mathcal{A}\star_c\mathbb{V}_m$ be the $(\ell\times(sm)\times p)$ and $(n\times(sm)\times p)$ tensors with frontal slices $\mathcal{V}_1,\ldots,\mathcal{V}_m$ and $\mathcal{A}\star_c\mathcal{V}_1,\ldots,\mathcal{A}\star_c\mathcal{V}_m$, respectively, and let $\mathbb{U}_m$ and $\mathcal{A}^T\star_c\mathbb{U}_m$ be the $(n\times(sm)\times p)$ and $(\ell\times(sm)\times p)$ tensors with frontal slices $\mathcal{U}_1,\ldots,\mathcal{U}_m$ and $\mathcal{A}^T\star_c\mathcal{U}_1,\ldots,\mathcal{A}^T\star_c\mathcal{U}_m$, respectively. We set

$$\begin{aligned} \mathbb{U}_m &:= [\mathcal{U}_1,\ldots,\mathcal{U}_m], \quad \text{and} \quad \mathcal{A}\star_c\mathbb{V}_m := [\mathcal{A}\star_c\mathcal{V}_1,\ldots,\mathcal{A}\star_c\mathcal{V}_m], \\ \mathbb{V}_m &:= [\mathcal{V}_1,\ldots,\mathcal{V}_m], \quad \text{and} \quad \mathcal{A}^T\star_c\mathbb{U}_m := [\mathcal{A}^T\star_c\mathcal{U}_1,\ldots,\mathcal{A}^T\star_c\mathcal{U}_m]. \end{aligned}$$

**Proposition 2.** *The tensors produced by the tensor cosine Golub-Kahan algorithm satisfy the following relations*

$$\begin{aligned} \mathcal{A}\star_c\mathbb{V}_m &= \mathbb{U}_{m+1}\circledast\widetilde{C}_m, \\ \mathcal{A}^T\star_c\mathbb{U}_m &= \mathbb{V}_m\circledast\widetilde{C}_m^T, \\ \mathbb{U}_{m+1}\circledast(\beta_1 e_1) &= \mathcal{C}, \\ \|\mathbb{U}_{m+1}\circledast z\|_F &= \|z\|_2, \end{aligned}$$

*where* $e_1=(1,0,\ldots,0)^T\in\mathbb{R}^{m+1}$ *and* $z$ *is a vector of* $\mathbb{R}^{m+1}$.

To solve the least squares problem (9), we consider approximations defined as

$$\mathcal{X}_m = \mathbb{V}_m\circledast y_m, \tag{10}$$

satisfying the minimization property of the corresponding residual. As we explained earlier, the problems that we are concerned with are ill-posed problems and then regularization techniques are highly recommended in those cases. But as the problem is very large, we apply the regularization process to the projected problem derived from the minimization of the residual. This leads to a low dimensional Tikhonov formulation and then we seek for $y=y_m\in\mathbb{R}^m$ that solves the low dimensional linear system of equations

$$(\widetilde{C}_m^T\widetilde{C}_m+\lambda_m^2 I_m)y = \alpha_1\widetilde{C}_m^T e_1, \qquad \alpha_1=\|\mathcal{C}\|_F,$$

which is also equivalent to solving the least-squares problem

$$\min_{y\in\mathbb{R}^m}\left\|\begin{bmatrix}\lambda_m\widetilde{C}_m\\ I_m\end{bmatrix}y-\alpha_1\lambda_m e_1\right\|_2. \tag{11}$$

The regularized parameter $\lambda_m$ is computed by using the GCV function given by (8). The following algorithm summarizes the main steps of the described method.

**Algorithm 5.** *The tensor discrete cosine Golub-Kahan (DC-GK) method.*

1. **Input.** *The tensors* $\mathcal{A}$, $\mathcal{C}$.
2. *Determine the orthonormal bases* $\mathbb{U}_{m+1}$ *and* $\mathbb{V}_m$ *of tensors, and the bidiagonal* $C_m$ *and* $\widetilde{C}_m$ *matrices with Algorithm 4.*
3. *Determine* $\lambda_m$ *using GCV function.*
4. *Determine* $y_{m,\lambda_m}$ *by solving (11) and then compute* $X_{m,\lambda_m}$ *by (10).*

In the next section, we derive a direct computation of the approximate Golub-Kahan solution by using a discrete cosine LSQR algorithm.

### 3.3 The discrete cosine-LQSR method

In this section, we develop the tensor version of the well know LSQR algorithm introduced in [9] based on c-product formalism. Let $\mathcal{A} \in \mathbb{R}^{n\times \ell \times p}$ be a tensor and let $\mathcal{C} \in \mathbb{R}^{n\times s\times p}$ a starting tensor.

The purpose of the tensor cosine LSQR method is to find, at some step $k$, an approximation $\mathcal{X}_k$ of the solution $\mathcal{X}^*$ of the problem (9), such that

$$\mathcal{X}_k = \mathbb{V}_k \circledast y_k,$$

where $y_k \in \mathbb{R}^k$. The associated residual is given by

$$\mathcal{R}_k = \mathcal{C} - \mathcal{A} \star_c \mathcal{X}_k = \beta_1 \mathcal{U}_1 - \mathbb{U}_{k+1} \circledast \widetilde{C}_k \circledast y_k = \mathbb{U}_{k+1} \circledast (\beta_1 e_1 - \widetilde{C}_k y_k),$$

and using Proposition 2, we get

$$\|\mathbb{U}_{k+1} \circledast (\beta_1 e_1 - \widetilde{C}_k y_k)\|_F = \|\beta_1 e_1 - \widetilde{C}_k y_k\|_2.$$

This minimization problem is accomplished by using the QR decomposition, where a unitary matrix $Q_k$ is determined so that

$$Q_k[\widetilde{C}_k \quad \beta_1 e_1] = \begin{bmatrix} R_k & f_k \\ 0 & \bar{\phi}_{k+1} \end{bmatrix} = \begin{bmatrix} \rho_1 & \theta_2 & & & & \phi_1 \\ & \rho_2 & \theta_3 & & & \vdots \\ & & \ddots & \ddots & & \vdots \\ & & & \rho_{k-1} & \theta_k & \phi_{k-1} \\ & & & & \rho_k & \phi_k \\ & & & & & \bar{\phi}_{k+1} \end{bmatrix},$$

where $\rho_i, \theta i$ are scalars. The matrix $Q_k$ is a product of plane rotations designed to eliminate the sub-diagonals of $\widetilde{C}_k$. This gives the following simple recurrence relation :

$$\begin{bmatrix} c_k & s_k \\ -s_k & c_k \end{bmatrix} \begin{bmatrix} \bar{\rho}_k & 0 & \bar{\phi}_k \\ \beta_{k+1} & \alpha_{k+1} & 0 \end{bmatrix} = \begin{bmatrix} \rho_k & \theta_{k+1} & \phi_k \\ 0 & \bar{\rho}_{k+1} & \bar{\phi}_{k+1} \end{bmatrix},$$

where $\bar{\rho}_1 = \alpha_1$ and $\bar{\phi}_1 = \beta_1$ and the scalars $s_k, c_k$ are the nontrivial element of $Q_{k+1,k}$ the $k$-th plane rotation. We get

$$R_k y_k = f_k,$$

and the approximate solution is given by:

$$\mathcal{X}_k = (\mathbb{V}_k \circledast R_k^{-1}) \circledast f_k.$$

Let

$$\mathbb{V}_k \circledast R_k^{-1} = \mathbb{P}_k = [\mathcal{P}_1 \dots \mathcal{P}_k],$$

then we have

$$\mathcal{X}_k = \mathbb{P}_k \circledast f_k.$$

Notice that the tensor $\mathcal{P}_k$ can be computed from $\mathcal{P}_{k-1}$ and $\mathcal{V}_k$ as follows:

$$\mathcal{P}_k = (\mathcal{V}_k - \theta_k \mathcal{P}_{k-1}) \rho_k^{-1}.$$

We also have $f_k = \begin{bmatrix} f_{k-1} \\ \phi_k \end{bmatrix}$ in which $\phi_k = c_k \bar{\phi}_k$. Finally, $\mathcal{X}_k$ can be computed as follows

$$\mathcal{X}_k = \mathcal{X}_{k-1} + \phi_k \mathcal{P}_k.$$

Furthermore, we have

$$\|\mathcal{R}_k\|_F = |\bar{\phi}_{k+1}|.$$

For ill posed problems, as it is the case for image or video restorations, we could have situations where the residual norm is small enough but the error norm is still large. As it is observed for those problems, the residual and the error norms could decrease in DC-LSQR till some iteration $k$ and then the norm of the error becomes to increase. One possibility to overcome these situations is to stop the iterations at some optimal $k_{opt}$. The L-curve criterion [5, 12] could be useful to determine such optimal index $k_{opt}$. The method suggests to plot the curve $(\|\mathcal{R}_k\|, \|\mathcal{X}_k\|)$. Intuitively, the best regularization parameter should lie on the corner of the L-curve corresponding to the point on the curve with maximum curvature. We notice that all the results of this section can be extended to the tensor equation $\mathscr{M}(\mathcal{X}) = \mathcal{A} \star_c \mathcal{X} \star_c \mathcal{B}$. The next algorithm which is named discrete cosine LSQR (DC-LSQR) algorithm, describes the whole process.

**Algorithm 6.** *The discrete cosine LSQR (DC-LSQR) algorithm.*

1. **Input.** *The tensors $\mathcal{A}$, $\mathcal{C}$, $\mathcal{X}_0 =$ , itermax, the maximum number of allowed iterations and a tolerance $tol > 0$ for the stopping criterion.*
2. *Set $\beta_1 = \|\mathcal{C}\|_F$, $\alpha_1 = \|\mathcal{A}^T \star_c \mathcal{U}_1\|_F$, $\mathcal{U}_1 = \mathcal{C}/\beta_1$ and $\mathcal{V}_1 = (\mathcal{A}^T \star_c \mathcal{U}_1)/\alpha_1$, $\mathcal{W}_1 = \mathcal{V}_1$ , $\bar{\rho}_1 = \alpha_1$ and $\bar{\phi}_1 = \beta_1$.*
3. *for $j = 1, \dots,$ itermax:*
   *(a) $\mathcal{W}_j = \mathcal{A} \star_c \mathcal{V}_j - \alpha_j \mathcal{U}_j$, $\beta_{j+1} = \|\mathcal{W}_j\|_F$ and $\mathcal{U}_{j+1} = \mathcal{W}_j/\beta_{j+1}$.*
   *(b) $\widetilde{\mathcal{V}} = \mathcal{A}^T \star_c \mathcal{U}_{j+1} - \beta_{j+1} \mathcal{V}_j$, $\alpha_{j+1} = \|\widetilde{\mathcal{V}}\|_F$ and $\mathcal{V}_{j+1} = \widetilde{\mathcal{V}}/\alpha_{j+1}$.*

(c) $\rho_j = (\bar{\rho}_j^2 + \beta_{j+1}^2)^{\frac{1}{2}}$, $c_j = \frac{\bar{\rho}_j}{\rho_j}$ *and* $s_j = \frac{\beta_{j+1}}{\rho_j}$.

(d) $\theta_{j+1} = s_j \alpha_{j+1}$ *and* $\bar{\rho}_{j+1} = c_j \alpha_{j+1}$.

(e) $\phi_j = c_j \bar{\phi}_j$ *and* $\bar{\phi}_{j+1} = -s_j \bar{\phi}_j$.

(f) $\mathcal{X}_i = \mathcal{X}_{i-1} + \frac{\phi_j}{\rho_j} \mathcal{W}_j$; $\mathcal{W}_{j+1} = \mathcal{V}_{j+1} - \frac{\theta_{j+1}}{\rho_j} \mathcal{W}_j$.

(g) *If* $|\bar{\phi}_{j+1}| < tol$ *stop.*

## 4 Numerical results

In this section, we give some numerical tests on the methods described in this paper. We compared the performances of the tensor discrete cosine GMRES describes in Algorithm 3, the tensor discrete cosine Golub-Kahan (DC-GK) algorithm given by Algorithm 5 and the tensor discrete cosine LSQR (DC-LSQR) described in Algorithm 6 , when applied to the restoration of blurred and noisy color images. All computations were carried out using the Matlab environment on an Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz (8 CPUs) computer with 12 GB of RAM. The computations were done with approximately 15 decimal digits of relative accuracy. Let $\widehat{X}^{(1)}$, $\widehat{X}^{(2)}$, and $\widehat{X}^{(3)}$ be the $n \times n$ matrices that constitute the three channels of the original error-free color image $\widehat{\mathcal{X}}$, and $\widehat{C}^{(1)}$, $\widehat{C}^{(2)}$, and $\widehat{C}^{(3)}$ the $n \times n$ matrices associated with error-free blurred color image $\widehat{\mathcal{C}}$. We consider that both cross-channel and within-channel blurring take place in the blurring process of the original image. The full blurring model

$$\mathcal{A} \star_c \widehat{\mathcal{X}} \star_c \mathcal{B} = \widehat{\mathcal{C}},$$

where $\mathcal{A}$ is a 3-way tensor such that $\mathcal{A}(:,:,1) = \alpha \mathbf{A}^{(2)}$, $\mathcal{A}(:,:,2) = \beta \mathbf{A}^{(2)}$ and $\mathcal{A}(:,:,3) = \gamma \mathbf{A}^{(2)}$ and $\mathcal{B}$ is a 3-way tensor with $\mathcal{B}(:,:,1) = (\mathbf{A}^{(1)})^T$, $\mathcal{B}(:,:,2) = 0$ and $\mathcal{B}(:,:,3) = 0$. $\alpha$, $\beta$ and $\gamma$ are the entries of the following matrix

$$\mathbf{A}_{\text{color}} = \begin{bmatrix} \alpha & \gamma & \beta \\ \beta & \alpha & \gamma \\ \gamma & \beta & \alpha \end{bmatrix},$$

obtained from [13], that models the cross-channel blurring, in which each row sums is one. $\mathbf{A}^{(1)} \in \mathbb{R}^{n \times n}$ and $\mathbf{A}^{(2)} \in \mathbb{R}^{n \times n}$ define within-channel blurring and they model the horizontal within blurring and the vertical within blurring matrices, respectively. To test the performance of algorithms, the within blurring matrices $A^{(i)}$ have the following entries:

$$a_{k\ell} = \begin{cases} \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(k-\ell)^2}{2\sigma^2}\right), & |k-\ell| \leq r, \\ 0, & \text{otherwise.} \end{cases}$$

Note that $\sigma$ controls the amount of smoothing, i.e. the larger the $\sigma$, the more ill posed the problem. We generated a blurred and noisy tensor image $\mathcal{C} = \widehat{\mathcal{C}} + \mathcal{N}$, where $\mathcal{N}$ is a noise tensor with normally distributed random entries with zero mean and with variance chosen to correspond to a specific noise level $\nu := \|\mathcal{N}\|_F / \|\widehat{\mathcal{C}}\|_F$. To compare the effectiveness of our solution methods, we evaluate

$$\text{Relative error} = \frac{\left\| \hat{\mathcal{X}} - \mathcal{X}_{\text{restored}} \right\|_F}{\|\widehat{\mathcal{X}}\|_F},$$

Figure 1: Original RGB images: `papav256` (left) and `cat1024` (right).

and the Signal-to-Noise Ratio (SNR) defined by

$$\mathrm{SNR}(\mathcal{X}_{\mathrm{restored}}) = 10\log_{10}\frac{\|\widehat{\mathcal{X}} - E(\widehat{\mathcal{X}})\|_F^2}{\|\mathcal{X}_{\mathrm{restored}} - \widehat{\mathcal{X}}\|_F^2},$$

where $E(\widehat{\mathcal{X}})$ denotes the mean gray-level of the uncontaminated image $\widehat{\mathcal{X}}$.

In our experiments, we applied the three algorithms DC-GMRES(10), DC-GK and DC-LSQR for the reconstruction of a cross-channel blurred color images that have been contaminated by both within and cross blur, and additive noise. The cross-channel blurring is determined by the matrix

$$\mathbf{A}_{\mathrm{color}} = \begin{bmatrix} 0.8 & 0.10 & 0.10 \\ 0.10 & 0.80 & 0.10 \\ 0.10 & 0.10 & 0.80 \end{bmatrix}.$$

We consider two RGB images, `papav256` ($\widehat{\mathcal{X}} \in \mathbb{R}^{256\times256\times3}$) and `cat1024` ($\widehat{\mathcal{X}} \in \mathbb{R}^{1024\times1024\times3}$). They are shown on Figure 1. For the within-channel blurring, we let $\sigma = 4$ and $r = 6$. The associated blurred and noisy RGB images are obtained as $\mathcal{C} = \mathcal{A} * \widehat{\mathcal{X}} * \mathcal{B} + \mathcal{N}$. Given the contaminated RGB image $\mathcal{C}$, we would like to recover an approximation of the original RGB image $\widehat{\mathcal{X}}$.

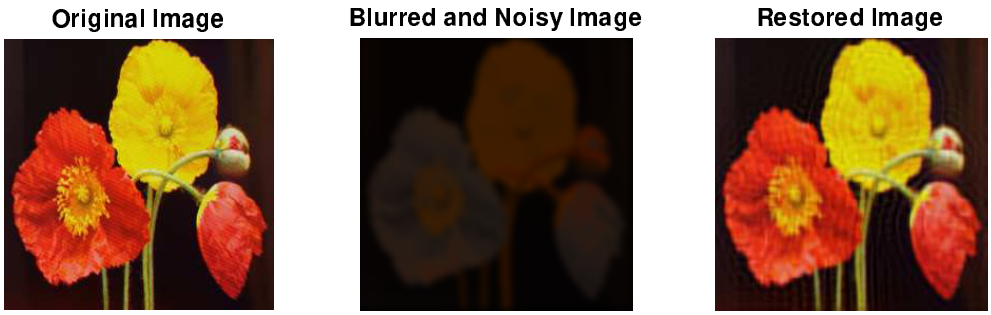


Figure 2: Test for Example 1, with DC-LSQR for `papav256`, and noise level $10^{-3}$. Original (left), noisy-blurred (center) and restored (right).

Table 1: Results for Experiment 1 with `papav256`. Noise level $10^{-3}$.

| RGB images | Method | SNR | Relative error | cpu-time (seconds) |
|---|---|---|---|---|
| `papav256` | DC-GMRES(10) | 19.25 | $8.5 \times 10^{-2}$ | 5.21 |
| | DC-GK | 23.9 | $4.7 \times 10^{-2}$ | 1.62 |
| | DC-LSQR | 21.8 | $6.8 \times 10^{-2}$ | 1.23 |

Table 2: Results for Example 1 with `papav256`. Noise level $10^{-2}$.

| RGB image | Method | SNR | Relative error | cpu-time (seconds) |
|---|---|---|---|---|
| `papav256` | DC-GMRES(10) | 17.24 | $1.5 \times 10^{-2}$ | 7.14 |
| | DC-GK | 20.4 | $7.2 \times 10^{-2}$ | 1.92 |
| | DC-LSQR | 20.2 | $8.5 \times 10^{-2}$ | 1.23 |

**Example 1.** In the first experiment, we used the `papav256` color image of size $256 \times 256 \times 3$ with two different noise levels $\nu = 10^{-2}$ or $\nu = 10^{-3}$. In Table 1 we reported the obtained SNR, the corresponding relative error norm and the required cpu-time for DC-GMRES(10), DC-GK and DC-LSQR with a noise level of $10^{-3}$.

As can be seen from those results, the DC-LSQR requires lower cpu-time as compared to the other two methods. However, DC-GK returns the best SNR. For this experiment, the optimal iteration number was $k_{opt} = 14$. A maximum number of iterations was $itermax = 10$ for DC-GMRES(10) and $mmax = 15$ for DC-GK. As we mentioned earlier, DC-GMRES(10) and DC-GK were run with the Tikhonov regularization technique (applied to the projected least squares problem) and we used GCV method for estimating the regularization parameters in each iteration of the processes. The obtained optimal parameters, at the final step were $\lambda_1 = 2.32 \times 10^{-5}$ for DC-GMRES(10) and $\lambda_1 = 1.24 \times 10^{-6}$ for DC-GK. Figure 2 shows the obtained blurred image and the restored one when using DC-LSQR method with noise level of $10^{-3}$.

In Table 2, we reported the results obtained by DC-GMRES(10), DC-GK and DC-LSQR for the color image `papav256` with a noise level of $10^{-2}$. Here also, we used $k_{opt} = 15$ for DC-LSQR, $itermax = 15$ for DC-GMRES(10) and $mmax = 20$ for DC-GK. As can be seen, the DC-LSQR returns the best results when comparing the three methods.


Figure 3: Test for Example 2, with DC-LSQR for `cat1024`, and noise level $10^{-3}$. Original (left), noisy-blurred (center) and restored (right).

Table 5: Results for DC-Golub-Kahan, T-Golub Kahan, T-LSQR and DC-LSQR, with noise level $10^{-2}$ with the image `cat1024`.

| Method | SNR | Relative error | cpu-time (seconds) |
|---|---|---|---|
| DC-GK | 15.65 | $6.18 \times 10^{-2}$ | 30.46 |
| T-GK | 15.06 | $8.17 \times 10^{-2}$ | 36.26 |
| T-LSQR | 15.53 | $4.07 \times 10^{-2}$ | 28.35 |
| DC-LSQR | 15.85 | $1.32 \times 10^{-2}$ | 25.05 |

Table 3: Results for Example 2 with noise level $10^{-3}$.

| RGB image | Method | SNR | Relative error | cpu-time (seconds) |
|---|---|---|---|---|
| `cat1024` | DC-GMRES(10) | 14.96 | $4.54 \times 10^{-2}$ | 100.35 |
| | DC-GK | 19.25 | $6.97 \times 10^{-2}$ | 25.33 |
| | DC-LSQR | 18.87 | $5.43 \times 10^{-2}$ | 19.45 |

Table 4: Results for Example 2 with noise level $10^{-2}$.

| RGB image | Method | SNR | Relative error | cpu-time (seconds) |
|---|---|---|---|---|
| `cat1024` | DC-GMRES(10) | 14.53 | $9.62 \times 10^{-2}$ | 137.43 |
| | DC-GK | 15.87 | $8.06 \times 10^{-2}$ | 30.22 |
| | DC-LSQR | 15.75 | $8.17 \times 10^{-2}$ | 26.43 |

**Example 2.** In the second example, we used the color image `cat1024` of dimension $1024 \times 1024 \times 3$. Here also, we compared the three methods using two noise levels $\nu = 10^{-3}$ and $\nu = 10^{-2}$. Table 3 reports on the obtained results for the noise level $\nu = 10^{-3}$ . For this experiment, the optimal iteration number for DC-LSQR was 15, the maximum iteration number allowed to DC-GMRES(10) was 10 and a maximum of $mmax = 20$ iterations was for DC-BK. As can be seen from the obtained results, DC-LSQR returns the best results: for SNR and the total cpu-time. Figure 3 shows the obtained blurred image and the restored one when using DC-LSQR method with noise level of $10^{-3}$ for the image `cat1024`. Table 4 reports on the obtained results for the noise level $\nu = 10^{-2}$. For this experiment, the optimal iteration number for DC-LSQR was 20, the maximum iteration number allowed to DC-GMRES(10) was 15 and a maximum of $mmax = 25$ iterations was for DC-BK. As can be seen from this table, DC-LSQR returns the best results: for SNR and the total cpu-time. For the returned SNR, generally the two Golub Kahan based methods return similar results but the second formulation of the method which corresponds the DC-LSQR (Algorithm 6) requires less cpu-time.

We also compared the performances of DC-Golub-Kahan with the T-Golub Kahan method that uses the tensor T-product. Usually, the approximate solution with the cosine product are better. In Table 5, we compared the obtained results for DC-GK and the T-GK algorithms for the image `cat1024`. We also run experiments comparing DC-LSQR and T-LSQR (which is obtained by replacing in DC-LSQR the c-product by the T-product) for the same example and the obtained results are also reported in Table 5.

## Conclusion

In this paper, we presented three discrete cosine Krylov-based methods, namely tensor DC-GMRES, DC-GK and DC-LSQR. The second two methods use the discrete cosine Golub-Kahan bidiagonalisation algorithm that we defined in this work. DC-GMRES and DC-GK are combined with the well known Tikhonov regularization method that is applied, at each iteration for the two algorithms, to the obtained projected low dimensional ill-posed least squares minimisation problem. The reported numerical tests show that the methods are very fact and can be used as restoration techniques for color image restoration.

## References

[1] N. Ahmed, T. Natarajan, K.R. Rao, *Discrete cosine transform*, IEEE Trans. Comput. **100** (1974) 90–93.

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

[3] R. Bouyouli, K. Jbilou, R. Sadaka, H. Sadok, *Convergence properties of some block Krylov subspace methods for multiple linear systems*, J. Comput. Appl. Math. **196** (2006) 498–511.

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

[5] D. Calvetti, P.C. Hansen, L. Reichel, *L-curve curvature bounds via Lanczos bidiagonalization*, Electron. Trans. Numer. Anal. **14** (2002) 20–35.

[6] D. Calvetti, L. Reichel, *Tikhonov regularization with a solution constraint*, SIAM J. Sci. Comput. **26** (2004), 224–239.

[7] D. Calvetti, G.H. Golub, L. Reichel, *Estimation of the L-curve via Lanczos bidiagonalization*, BIT **39** (1999), 603–619.

[8] M. El Guide, A. El Ichi, K. Jbilou, F.P.A Beik, *Tensor GMRES and Golub-Kahan Bidiagonalization methods via the Einstein product with applications to image and video processing*, arXiv preprint arXiv:2005.07458.

[9] G.H. Golub, W. Kahan, *Algorithm LSQR is based on the Lanczos process and bidiagonalization procedure*, SIAM J. Numer. Anal. **2** (1965) 205–224.

[10] G.H. Golub, C.F. Van Loan, *Matrix Computations*, 3rd ed., Johns Hopkins University Press, Baltimore, 1996.

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

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

[13] P.C. Hansen, J. Nagy, D.P. O'Leary, *Deblurring Images: Matrices, Spectra and Filtering*, SIAM, Philadelphia, 2006.

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

[15] K. Jbilou, A. Messaoudi, H. Sadok, *Global FOM and GMRES algorithms for matrix equations*, Appl. Numer. Math. **31** (1999) 49–63.

[16] M.N. Kalyani, F.P.A. Beik, K. Jbilou, *On global iterative schemes based on Hessenberg process for (ill-posed) Sylvester tensor equations*, J. Comput. Appl. Math. **373** (2020) 112216.

[17] E. Kernfeld, M. Kilmer, S. Aeron, *Tensor-tensor products with invertible linear trans- forms*, Linear Algebra Appl. **485** (2015) 545–570.

[18] T.G. Kolda, B.W. Bader, *Tensor Decompositions and Applications*, SIAM Rev. **51** (2009) 455–500 .

[19] T. G. Kolda, B. W. Bader, J. P. Kenny, *Higher-order web link analysis using multilinear algebra*, Proc. 5th IEEE Int. Conf. on Data Mining, pp. 242-249, November 2005.

[20] M.E. Kimler, C.D. Martin, *Factorization strategies for third-order tensors*, Linear Algebra Appl. **435** (2011) 641–658.

[21] C. Lu, J. Feng, Y. Chen, W. Liu, Z. Lin, S. Yan, *Tensor Robust Principal Component Analysis with a New Tensor Nuclear Norm*, IEEE Trans. Pattern Anal. Mach. Intell. **42** (2019) 925–938.

[22] L. De Lathauwer, A. de Baynast, *Blind deconvolution of DS-CDMA signals by means of decomposition in rank-(l, L, L) terms*, IEEE Trans. Signal Process. **56** (2008) 1562–1571.

[23] X. Li, M.K. Ng, *Solving sparse non-negative tensor equations: algorithms and applications*, Front. Math. China **10** (2015) 649–680.

[24] M.K. Ng, R.H. Chan, W. Tang, *A fast algorithm for deblurring models with Neumann boundary conditions*, SIAM J. Sci. Comput. **21** (1999) 851–866.

[25] L. Qi, Z. Luo, *Tensor analysis: spectral theory and special tensors*, SIAM, Philadelphia, 2017.

[26] L. Sun, B. Zheng, C. Bu, Y. Wei, *Moore Penrose inverse of tensors via Einstein product*, Linear Multilinear Algebra **64** (2016) 686–698.

[27] A.N. Tikhonov, *Regularization of incorrectly posed problems*, Sov. Math., Dokl. **4** (1963) 1624–1627.

[28] G. Wahba, *Pratical approximation solutions to linear operator equations when the data are noisy*, SIAM J. Numer. Anal. **14** (1977) 651–667.