

An optimality property of approximated solution computed by Hessenberg method

MEHDI NAJAFI-KALYANI* AND FATEMEH P. A. BEIK

Department of Mathematics, Vali-e-Asr University of Rafsanjan, PO Box 518, Rafsanjan, Iran

Received ; accepted ??

Abstract. We revisit the implementation of the Krylov subspace method based on the Hessenberg process for general linear operator equations. It is established that at each step, the computed approximate solution by the corresponding approach can be regarded as the minimizer of a certain norm of residual corresponding to the obtained approximate solution of system. Test problems are numerically examined for solving tensor equations with cosine transform product arising from image restoration to compare the performance of Krylov subspace methods in conjunction with the Tikhonov regularization technique based on Hessenberg and Arnoldi processes.

AMS subject classifications: 65F10, 15A09

Key words: Krylov subspace method, Tensor equation, Tikhonov regularization, Cosine transform product, Hessenberg process, Arnoldi process, Image processing

1. Introduction

A multidimensional array of data is called a tensor whose modes stand for its number of indices. Throughout this paper, vectors and matrices are respectively denoted by lowercase and capital letters, and tensors are represented by Euler script.

For the sake of generality, we consider the following linear operator equation

$$\mathcal{F}(\mathcal{X}) = \mathcal{G}, \quad (1)$$

where $\mathcal{F}(\cdot)$ is a given linear operator from $\mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ onto $\mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$. The tensor equation in the form (1) incorporates several class of recently mentioned tensor equations in the literature including multilinear systems [6, 9, 19], Sylvester matrix equation [4, 5, 7], Stein tensor equation [4] and etc. Basically, special cases of Eq. (1) appear in numerous areas such as Markov process [8], physics [9] and numerical discretization of (high order) partial differential equations [4, 5, 7, 19] from engineering problems.

Based on the Arnoldi process, several variants of Krylov subspace methods have been developed in the literature for solving systems in the form (1), see [4, 5, 9, 10] and references therein. To be more precise, let us consider the Sylvester tensor equation (STE) as a special case of (1) and review some of recently proposed methods to solve STEs. To this end, first, we need to present the definitions of mode- n product [22].

*Corresponding author. *Email addresses:* `m.najafi.uk@gmail.com` (Mehdi Najafi-Kalyani), `f.beik@vru.ac.ir` (Fateme P. A. Beik)

Definition 1. The n -mode (matrix) product of a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ with a matrix $U \in \mathbb{R}^{J \times I_n}$ is denoted by $\mathcal{X} \times_n U$ and is of size

$$I_1 \times \cdots \times I_{n-1} \times J \times I_{n+1} \times \cdots \times I_N,$$

and its elements are defined as follows:

$$(\mathcal{X} \times_n U)_{i_1 \cdots i_{n-1} j i_{n+1} \cdots i_N} = \sum_{i_n=1}^{I_n} x_{i_1 i_2 \cdots i_N} u_{j i_n}.$$

Consider the STE as follows:

$$\mathcal{X} \times_1 A^{(1)} + \mathcal{X} \times_2 A^{(2)} + \cdots + \mathcal{X} \times_N A^{(N)} = \mathcal{D}, \quad (2)$$

where the right-hand side tensor $\mathcal{D} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ and coefficient matrices $A^{(n)} \in \mathbb{R}^{I_n \times I_n}$ ($n = 1, 2, \dots, N$) are known and $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is unknown. In the literature, several variants of the Krylov subspace methods were proposed for solving the above STE, see [5, 7, 10, 23, 26] and the references therein. In particular, Kressner and Tobler [23] applied Krylov subspace methods based on (extended) Arnoldi process for the case that the right-hand side \mathcal{D} is a tensor of low rank. For the case that \mathcal{D} does not necessarily have a low rank, Chen and Lu [10] developed the generalized minimal residual (GMRES) method in tensor framework. The tensor form of the full orthogonalization method (FOM) was presented in [6]. STEs with dense coefficient matrices $A^{(1)}, A^{(2)}, \dots, A^{(N)}$ can possibly arise from discretization of three-dimensional partial differential equations by spectral methods [24, 25]. In [26], it is observed that using the Hessenberg process instead of Arnoldi process can lead to a computationally cheaper Krylov subspace method when the coefficient matrices in the STE are dense.

It is known that replacing the Hessenberg process by the Arnoldi process can lead to cost-effective Krylov subspace methods, see [18, 29]. This fact motivated several researchers to extend Krylov subspace methods based on Hessenberg process for solving different types of linear operator equations in the form (1). For instance, the block Changing Minimal Residual method based on the Hessenberg (CMRH) method was proposed in [1] to solve linear systems of the form $AX = B$ where A is nonsymmetric. The weighted and flexible versions of block CMRH were also presented in [2]. Gu et al. [13] proposed a restarted Hessenberg method to solve shifted nonsymmetric linear systems. In [14], the restarted CMRH process was presented for solving multi-shifted linear systems with non-Hermitian coefficient matrices. Recently, a Hessenberg-type method was applied for the solution of PageRank problems, see [15]. Brief discussions are included in Appendix A to recall Hessenberg and Arnoldi processes associated with linear operator $\mathcal{F}(\cdot)$ in the tensor equation (1) and compare their computational costs.

The remainder of this paper is organized as follows: In Section 2, we briefly explain how the Hessenberg method is used for solving (1). The optimality property of approximate solution obtained by the Hessenberg method is established in Section 3. Numerical experiments are reported in Section 4 to disclose comparison results between Hessenberg and Arnoldi processes in conjunction with the Tikhonov regularization technique for a class of tensor equations arising from image restoration. We finish the paper by some concluding remarks in Section 5.

2. An overview on the Hessenberg method

In this section, we briefly review the implementation of Hessenberg method for (1). To do so, we need to recall the inner product between two tensors and its induced norm. The inner product between two same size tensors \mathcal{X} and \mathcal{Y} in $\mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is given by

$$\langle \mathcal{X}, \mathcal{Y} \rangle = \sum_{i_1=1}^{I_1} \sum_{i_2=1}^{I_2} \cdots \sum_{i_N=1}^{I_N} x_{i_1 i_2 \cdots i_N} y_{i_1 i_2 \cdots i_N}. \quad (3)$$

Given a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, the induced norm from the above inner product is defined by

$$\|\mathcal{X}\|^2 = \sum_{i_1=1}^{I_1} \sum_{i_2=1}^{I_2} \cdots \sum_{i_N=1}^{I_N} x_{i_1 i_2 \cdots i_N}^2.$$

Corresponding to the tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, frontal slices or column tensors of \mathcal{X} have the following form:

$$\underbrace{\mathcal{X}_{:: \cdots : k}}_{(N-1)\text{-times}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_{N-1}}, \quad k = 1, 2, \dots, I_N,$$

see [22] for further details. When \mathcal{X} is a tensor of order three, we also use the notation $\mathcal{X}^{(k)}$ to denote its k -th frontal slice.

Constructing iterative schemes based on Hessenberg process for solving (1) follows from similar strategy used in [26] and related details are omitted here. Basically, one can develop the Hessenberg method by constructing the basis $\{\mathcal{V}_1, \mathcal{V}_2, \dots, \mathcal{V}_m\}$ for the following Krylov subspace

$$\mathcal{K}_m(\mathcal{F}, \mathcal{R}_0) = \text{span}\{\mathcal{R}_0, \mathcal{F}(\mathcal{R}_0), \dots, \mathcal{F}^{m-1}(\mathcal{R}_0)\}, \quad (4)$$

using Algorithm 1 in Appendix A such that

$$\langle \mathcal{V}_{i+1}, \mathcal{Y}_j \rangle = 0, \quad \text{for } j = 1, 2, \dots, i,$$

in which the linearly independent tensors \mathcal{Y}_i s of order $I_1 \times I_2 \times \cdots \times I_N$ are available, $\mathcal{R}_0 = \mathcal{G} - \mathcal{F}(\mathcal{X}_0)$ and the initial guess $\mathcal{X}_0 \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is given.

In what follows, we define

$$\bar{H}_m := \begin{pmatrix} H_m \\ e_m^\top h_{m+1, m} \end{pmatrix},$$

where the (i, j) -th entry of H_m is denoted by h_{ij} being computed in Lines 5 and 8 of Algorithm 1.

Suppose that $\tilde{\mathcal{V}}_m$ and $\tilde{\mathcal{Y}}_m$ are $(N+1)$ -mode tensors with column tensors \mathcal{V}_i s and \mathcal{Y}_i s for $i = 1, 2, \dots, m$. The following theorem is useful to derive Krylov subspace methods based on Hessenberg process for linear operator equations in the form (1). The proof of theorem follows from similar strategies used in [5, 26].

Theorem 1. Let $\tilde{\mathcal{W}}_m$ be the $(N+1)$ -mode tensor with column tensors $\mathcal{W}_j := \mathcal{F}(\mathcal{V}_j)$ for $j = 1, \dots, m$. Then the following statements hold

$$\tilde{\mathcal{W}}_m = \tilde{\mathcal{V}}_{m+1} \times_{(N+1)} \bar{H}_m^\top, \quad (5)$$

$$\tilde{\mathcal{W}}_m = \tilde{\mathcal{V}}_m \times_{(N+1)} H_m^\top + h_{m+1,m} \mathcal{Z} \times_{(N+1)} E_m, \quad (6)$$

in which \mathcal{Z} is an $(N+1)$ -mode tensor with “ m ” column tensors $0, \dots, 0, \mathcal{V}_{m+1}$ and E_m is an $m \times m$ matrix of the form $E_m = [0, \dots, 0, e_m]$ where e_m is the m -th column of the identity matrix of order m .

Let $\mathcal{V}_1, \mathcal{V}_2, \dots, \mathcal{V}_m$ be a basis for $\mathcal{K}_m(\mathcal{F}, \mathcal{R}_0)$ produced via Algorithm 1. The m -th approximate solution \mathcal{X}_m is determined such that

$$\mathcal{X}_m \in \mathcal{X}_0 + \mathcal{K}_m(\mathcal{F}, \mathcal{R}_0),$$

which implies that

$$\mathcal{X}_m = \mathcal{X}_0 + \sum_{i=1}^m \mathcal{V}_i y_m^{(i)}. \quad (7)$$

The following orthogonality conditions in the Hessenberg method are imposed

$$\langle \mathcal{R}_m, \mathcal{Y}_i \rangle = 0, \quad \text{for } i = 1, 2, \dots, m, \quad (8)$$

to obtain the unknown vector $y_m = (y_m^{(1)}; y_m^{(2)}; \dots; y_m^{(m)})$ where $\mathcal{R}_m = \mathcal{G} - \mathcal{F}(\mathcal{X}_m)$ and the MATLAB notation $(w_1; w_2; \dots; w_m)$ represents the vector $(w_1, w_2, \dots, w_m)^\top$. Using Theorem 1, is not difficult to verify that y_m satisfies

$$H_m y_m = \beta e_1, \quad (9)$$

with $\beta = \langle \mathcal{R}_0, \mathcal{Y}_1 \rangle$. It can be verified that

$$\mathcal{R}_m = \mathcal{G} - \mathcal{F}(\mathcal{X}_m) = -h_{m+1,m} y_m^{(m)} \mathcal{V}_{m+1},$$

see [26] for more details.

3. An optimality property of the Hessenberg method

Let $\mathcal{F} : \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N} \rightarrow \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}$ be a given arbitrary invertible linear operator, i.e., $\mathcal{F}(\mathcal{X}) = 0$ implies $\mathcal{X} = 0$. In this section, we show that the computed approximate solution by the Hessenberg method at each step satisfies an optimality property. To this end, first, we need to recall a special case of contracted tensor product.

Definition 2. [5] The \boxtimes^N product between N -mode tensors $\mathcal{X} \in \mathbb{R}^{I_1 \times \dots \times I_{N-1} \times I_N}$ and $\mathcal{Y} \in \mathbb{R}^{I_1 \times \dots \times I_{N-1} \times \tilde{I}_N}$ is defined as an $I_N \times \tilde{I}_N$ matrix whose (i, j) -th entry is

$$[\mathcal{X} \boxtimes^N \mathcal{Y}]_{ij} = \text{tr}(\mathcal{X}_{::\dots:i} \boxtimes^{N-1} \mathcal{Y}_{::\dots:j}), \quad N = 3, 4, \dots,$$

where

$$\mathcal{X} \boxtimes^2 \mathcal{Y} = \mathcal{X}^\top \mathcal{Y}, \quad \mathcal{X} \in \mathbb{R}^{I_1 \times I_2}, \mathcal{Y} \in \mathbb{R}^{I_1 \times \tilde{I}_2}.$$

We comment that the \boxtimes^N product between \mathcal{X} and \mathcal{Y} is a reformulation of a special case of the contracted product [11]. As pointed out in [5], it can be verified that

$$\langle \mathcal{X}, \mathcal{Y} \rangle = \text{tr}(\mathcal{X} \boxtimes^N \mathcal{Y}), \quad N = 2, 3, \dots, \quad (10)$$

for $\mathcal{X}, \mathcal{Y} \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}$.

Now we define a new inner product and its associated norm as well in order to establish the following proposition showing that computed approximate solution by the Hessenberg method is the minimizer of residual corresponding to the approximate solution of (1).

Definition 3. Let $\tilde{\mathcal{Y}}_k$ be a $(N+1)$ -mode tensor with frontal slices \mathcal{Y}_i for $i = 1, 2, \dots, m$ and $\epsilon > 0$ is given. For $\mathcal{X}, \mathcal{Z} \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}$, we define the following inner product

$$\langle \mathcal{X}, \mathcal{Z} \rangle_{\tilde{\mathcal{Y}}_k, \epsilon} = \left\langle \tilde{\mathcal{Y}}_k \boxtimes^{(N+1)} \mathcal{X}, \tilde{\mathcal{Y}}_k \boxtimes^{(N+1)} \mathcal{Z} \right\rangle + \epsilon \langle \mathcal{X}, \mathcal{Z} \rangle. \quad (11)$$

The corresponding tensor norm is given by $\|\mathcal{X}\|_{\tilde{\mathcal{Y}}_k, \epsilon}^2 = \langle \mathcal{X}, \mathcal{X} \rangle_{\tilde{\mathcal{Y}}_k, \epsilon}$.

We add the following remark to the previous definition to clarify the fact that the bilinear form (11) is an inner product.

Remark 1. Considering the equality (10), one can see that

$$\left\langle \tilde{\mathcal{Y}}_k \boxtimes^{(N+1)} \mathcal{X}, \tilde{\mathcal{Y}}_k \boxtimes^{(N+1)} \mathcal{Z} \right\rangle = \langle w_x, w_z \rangle$$

where $w_x = (\langle \mathcal{Y}_1, \mathcal{X} \rangle; \langle \mathcal{Y}_2, \mathcal{X} \rangle; \dots; \langle \mathcal{Y}_m, \mathcal{X} \rangle)$ and $w_z = (\langle \mathcal{Y}_1, \mathcal{Z} \rangle; \langle \mathcal{Y}_2, \mathcal{Z} \rangle; \dots; \langle \mathcal{Y}_m, \mathcal{Z} \rangle)$. Therefore, the bilinear form (11) is basically the summation of two inner products. Since $\epsilon > 0$, it is not difficult to verify that the bilinear form (11) is indeed an inner product.

Proposition 1. Assume that the linear operator $\mathcal{F}(\cdot)$ is invertible, i.e., $\mathcal{F}(\mathcal{X}) = 0$ implies $\mathcal{X} = 0$. Let \mathcal{X}_k be the k -th approximate solution of $\mathcal{F}(\mathcal{X}) = \mathcal{G}$ obtained after implementing Hessenberg method. Then, there exists $\hat{\epsilon} > 0$ such that

$$\|\mathcal{G} - \mathcal{F}(\mathcal{X}_k)\|_{\tilde{\mathcal{Y}}_k, \epsilon} < \|\mathcal{G} - \mathcal{F}(\hat{\mathcal{X}})\|_{\tilde{\mathcal{Y}}_k, \epsilon} \quad \text{for } 0 < \epsilon < \hat{\epsilon}, \quad (12)$$

for any $\hat{\mathcal{X}} \in \mathcal{X}_0 + \mathcal{K}_k(\mathcal{F}, \mathcal{R}_0)$ provided that $\tilde{\mathcal{Y}}_k \boxtimes^{(N+1)} \mathcal{F}(\mathcal{X}_k - \hat{\mathcal{X}}) \neq 0$. Here the frontal slices of $(N+1)$ -mode tensor $\tilde{\mathcal{Y}}_k$ are given by $\mathcal{Y}_1, \mathcal{Y}_2, \dots, \mathcal{Y}_k$ such that

$$\langle \mathcal{R}_k, \mathcal{Y}_i \rangle = 0, \quad \text{for } i = 1, 2, \dots, k,$$

in which $\mathcal{R}_k = \mathcal{G} - \mathcal{F}(\mathcal{X}_k)$.

Proof. Let $\hat{\mathcal{X}} \in \mathcal{X}_0 + \mathcal{K}_k(\mathcal{F}, \mathcal{R}_0)$ and $\tilde{\mathcal{Y}}_k \boxtimes^{(N+1)} \mathcal{F}(\mathcal{X}_k - \hat{\mathcal{X}}) \neq 0$. It is not difficult to verify that

$$\begin{aligned} \|\mathcal{G} - \mathcal{F}(\hat{\mathcal{X}})\|_{\tilde{\mathcal{Y}}_k, \epsilon}^2 &= \left\langle \mathcal{G} - \mathcal{F}(\hat{\mathcal{X}}), \mathcal{G} - \mathcal{F}(\hat{\mathcal{X}}) \right\rangle_{\tilde{\mathcal{Y}}_k, \epsilon} \\ &= \left\langle \mathcal{R}_k + \mathcal{F}(\mathcal{X}_k - \hat{\mathcal{X}}), \mathcal{R}_k + \mathcal{F}(\mathcal{X}_k - \hat{\mathcal{X}}) \right\rangle_{\tilde{\mathcal{Y}}_k, \epsilon} \\ &= \|\mathcal{R}_k\|_{\tilde{\mathcal{Y}}_k, \epsilon}^2 + 2 \left\langle \mathcal{R}_k, \mathcal{F}(\mathcal{X}_k - \hat{\mathcal{X}}) \right\rangle_{\tilde{\mathcal{Y}}_k, \epsilon} + \|\mathcal{F}(\mathcal{X}_k - \hat{\mathcal{X}})\|_{\tilde{\mathcal{Y}}_k, \epsilon}^2. \end{aligned} \quad (13)$$

Notice that $\mathcal{F}(\hat{\mathcal{X}} - \mathcal{X}_k) \neq 0$, hence, the invertibility of $\mathcal{F}(\cdot)$ implies that $\mathcal{F}(\hat{\mathcal{X}}) \neq \mathcal{F}(\mathcal{X}_k)$. From (13), one can see that

$$\begin{aligned} \left\| \mathcal{G} - \mathcal{F}(\hat{\mathcal{X}}) \right\|_{\tilde{\mathcal{Y}}_{k,\epsilon}}^2 &= \|\mathcal{R}_k\|_{\tilde{\mathcal{Y}}_{k,\epsilon}}^2 + 2 \left\langle \tilde{\mathcal{Y}}_k \boxtimes^{(N+1)} \mathcal{R}_k, \tilde{\mathcal{Y}}_k \boxtimes^{(N+1)} \mathcal{F}(\mathcal{X}_k - \hat{\mathcal{X}}) \right\rangle \\ &\quad + \epsilon \left\langle \mathcal{R}_k, \mathcal{F}(\mathcal{X}_k - \hat{\mathcal{X}}) \right\rangle + \left\| \mathcal{F}(\mathcal{X}_k - \hat{\mathcal{X}}) \right\|_{\tilde{\mathcal{Y}}_{k,\epsilon}}^2. \end{aligned}$$

Invoking the fact that $\tilde{\mathcal{Y}}_k \boxtimes^{(N+1)} \mathcal{R}_k$ is a zero vector, we obtain

$$\begin{aligned} \left\| \mathcal{G} - \mathcal{F}(\hat{\mathcal{X}}) \right\|_{\tilde{\mathcal{Y}}_{k,\epsilon}}^2 &= \|\mathcal{R}_k\|_{\tilde{\mathcal{Y}}_{k,\epsilon}}^2 + \left\| \mathcal{F}(\mathcal{X}_k - \hat{\mathcal{X}}) \right\|_{\tilde{\mathcal{Y}}_{k,\epsilon}}^2 + 2\epsilon \left\langle \mathcal{R}_k, \mathcal{F}(\mathcal{X}_k - \hat{\mathcal{X}}) \right\rangle \\ &> \|\mathcal{R}_k\|_{\tilde{\mathcal{Y}}_{k,\epsilon}}^2 + \left\| \mathcal{F}(\mathcal{X}_k - \hat{\mathcal{X}}) \right\|_{\tilde{\mathcal{Y}}_{k,\epsilon}}^2 - 2\epsilon \left| \left\langle \mathcal{R}_k, \mathcal{F}(\mathcal{X}_k - \hat{\mathcal{X}}) \right\rangle \right|. \end{aligned} \quad (14)$$

If $\left\langle \mathcal{R}_k, \mathcal{F}(\mathcal{X}_k - \hat{\mathcal{X}}) \right\rangle = 0$, the assertion follows from the above inequality for any $\epsilon > 0$. Consequently, without loss of generality, we assume that $\left\langle \mathcal{R}_k, \mathcal{F}(\mathcal{X}_k - \hat{\mathcal{X}}) \right\rangle \neq 0$ and define

$$\hat{\epsilon} := \frac{\left\| \tilde{\mathcal{Y}}_k \boxtimes^{(N+1)} \mathcal{F}(\mathcal{X}_k - \hat{\mathcal{X}}) \right\|^2}{2 \left| \left\langle \mathcal{R}_k, \mathcal{F}(\mathcal{X}_k - \hat{\mathcal{X}}) \right\rangle \right|}. \quad (15)$$

In view of (14), for any $\epsilon < \hat{\epsilon}$, we have

$$\begin{aligned} \left\| \mathcal{G} - \mathcal{F}(\hat{\mathcal{X}}) \right\|_{\tilde{\mathcal{Y}}_{k,\epsilon}}^2 &> \|\mathcal{R}_k\|_{\tilde{\mathcal{Y}}_{k,\epsilon}}^2 + \left\| \mathcal{F}(\mathcal{X}_k - \hat{\mathcal{X}}) \right\|_{\tilde{\mathcal{Y}}_{k,\epsilon}}^2 - 2\hat{\epsilon} \left| \left\langle \mathcal{R}_k, \mathcal{F}(\mathcal{X}_k - \hat{\mathcal{X}}) \right\rangle \right| \\ &= \|\mathcal{R}_k\|_{\tilde{\mathcal{Y}}_{k,\epsilon}}^2 + \epsilon \left\| \mathcal{F}(\mathcal{X}_k - \hat{\mathcal{X}}) \right\|_{\tilde{\mathcal{Y}}_{k,\epsilon}}^2 \\ &> \|\mathcal{R}_k\|_{\tilde{\mathcal{Y}}_{k,\epsilon}}^2 = \|\mathcal{G} - \mathcal{F}(\mathcal{X}_k)\|_{\tilde{\mathcal{Y}}_{k,\epsilon}}^2, \end{aligned}$$

which completes the proof. \square

We end this part with the following remark on choosing ϵ .

Remark 2. Assume that the assumptions of Proposition 1 hold. In view of (15) and Cauchy-Schwarz inequality, one may set

$$\hat{\epsilon} = \frac{1}{2} \cdot \min_{0 \neq \mathcal{Z} \in \mathcal{K}_k(\mathcal{F}, \mathcal{R}_0)} \left\{ \frac{\|\mathcal{W}(\mathcal{Z})\|}{\|\mathcal{R}_k\|} \mid \mathcal{R}_k \neq 0 \quad \text{and} \quad \mathcal{W}(\mathcal{Z}) \neq 0 \right\}$$

to eliminate the dependency of $\hat{\epsilon}$ on $\hat{\mathcal{X}}$ in the statement of Proposition 1 where $\mathcal{W}(\mathcal{Z}) := \tilde{\mathcal{Y}}_k \boxtimes^{(N+1)} \mathcal{F}(\mathcal{Z})$. In fact, the k -th approximate solution obtained via Hessian method satisfies the optimality property (12) for any $0 < \epsilon < \hat{\epsilon}$.

4. An application from image processing

Developing efficient image deblurring methods is an active area of research. For iterative methods based on the Krylov subspace, one can refer to [4, 7, 12, 26, 30,

31, 32] and the reference therein. For example, Guo et al. [16] recently developed a three-dimensional fractional total variation based-model for three-dimensional image deblurring problem.

In this section, we aim to experimentally illustrate the performance of the Hessenberg method in the context of image deblurring. For this application, the implementation of the Hessenberg method has been already considered in [26] when Eq. (1) is reduced to the Sylvester tensor equation. Here, we implement the method for solving an alternative tensor equation. To this end, before presenting numerical results, we need to review some preliminaries.

4.1. Basic concepts

To present our mentioned tensor equation in reported numerical experiments, we need to recall the definition of $*_c$ -product from [20].

Definition 4 ($*_c$ -product). *Let $\mathcal{A} \in \mathbb{R}^{m \times \ell \times n}$ and $\mathcal{B} \in \mathbb{R}^{\ell \times p \times n}$. The tensor-tensor product $\mathcal{C} = \mathcal{A} *_c \mathcal{B}$ is of size $m \times p \times n$ such that*

$$\hat{\mathcal{C}}^{(i)} := \hat{\mathcal{A}}^{(i)} \hat{\mathcal{B}}^{(i)}, \quad \text{for } i = 1, \dots, n,$$

where $\hat{\mathcal{A}} = \mathcal{A} \times_3 M$, $\hat{\mathcal{B}} = \mathcal{B} \times_3 M$ and $\mathcal{C} = \hat{\mathcal{C}} \times_3 M^{-1}$. The matrix $M = W^{-1}C(I+Z)$ can be computed in MATLAB using

$$C = \text{dct}(\text{eye}(n)), \quad W = \text{diag}(C(:, 1)), \quad Z = \text{diag}(\text{ones}(n-1, 1), 1).$$

We consider the following tensor equation

$$\mathcal{A} *_c \mathcal{X} = \mathcal{G}, \tag{16}$$

where tensor $\mathcal{A} \in \mathbb{R}^{\ell \times \ell \times m}$ and the right-hand side $\mathcal{G} \in \mathbb{R}^{\ell \times p \times m}$ are given and $\mathcal{X} \in \mathbb{R}^{\ell \times p \times m}$ is the unknown tensor. The tensor problem (16) may appear in engineering, signal processing, image and video data processing problems, see [12, 20, 21, 27, 28].

Notice that the linear operator $\mathcal{F}(\cdot)$ takes the following form

$$\begin{aligned} \mathcal{F} : \mathbb{R}^{\ell \times p \times m} &\rightarrow \mathbb{R}^{\ell \times p \times m} \\ \mathcal{X} &\mapsto \mathcal{F}(\mathcal{X}) := \mathcal{A} *_c \mathcal{X}. \end{aligned}$$

In the sequel, we briefly explain the strategy for exploiting tensors to reformat a typical discrete model for image blurring; see [20] for further details. Consider the linear system of equations

$$Bx = g, \tag{17}$$

where B denotes the discrete blurring matrix of order n^2 and x is the vectorized form of the image X . The right-hand side of the above equation contains an error e which is called “noise”, i.e., $g = \hat{g} + e$ where \hat{g} is the unknown noise-free unavailable right-hand side. Following the discussions in [20], one can reformulate Eq. (17) by

$$\mathcal{A} *_c \mathcal{X} = \mathcal{G}, \quad \mathcal{G} = \hat{\mathcal{G}} + \mathcal{N}, \tag{18}$$

where the image X is obtained by “reshaping” the first column of $\mathbf{mat}(\mathcal{X})$, and the right-hand side tensor \mathcal{G} is obtained by $\mathbf{ten}(g)$, see [20] for the definition of $\mathbf{mat}(\cdot)$ and $\mathbf{ten}(\cdot)$. The tensor \mathcal{G} in (18) is contaminated by a noise tensor \mathcal{N} with normally distributed random entries with zero mean and scaled to correspond to a specific noise level $\eta = \|\mathcal{N}\|/\|\mathcal{G}\|$.

Let $B = T \otimes T$ and T is a Toeplitz matrix representing a Gaussian blur. We created $T \in \mathbb{R}^{n \times n}$ in MATLAB as follows:

$$T = \frac{1}{\sqrt{2\pi\sigma^2}} \cdot \mathbf{toeplitz}(\mathbf{z}) \quad (19)$$

with

$$\mathbf{z} = [\exp(-(0:\mathbf{band}-1)^2/(2\sigma^2)), \mathbf{zeros}(1, n - \mathbf{band})]$$

where \mathbf{band} and σ are given for each of our test problems, and “ $\mathbf{toeplitz}(\cdot)$ ” is a command in MATLAB. To generate coefficient tensor \mathcal{A} in (18), we set $\mathcal{A}^{(i)} = T(i, 1)T$ for $i = 1, 2, \dots, n$, recalling that $\mathcal{A}^{(i)}$ denotes the i -th frontal slice of \mathcal{A} .

4.2. Numerical experiments

All numerical experiments were computed using MATLAB version 9.9 (R2020b) running on an Intel Core i5 CPU at 2.50 GHz with 8 GB of memory using Tensor Toolbox [3].

The Tikhonov regularization technique consists of replacing the solution of (18) by the following minimization problem

$$\min_{\mathcal{X} \in \mathbb{R}^{\ell \times p \times m}} \{ \|\mathcal{A} *_c \mathcal{X} - \mathcal{G}\|^2 + \mu \|\mathcal{X}\|^2 \},$$

where $\mu > 0$ is the regularization parameter. In the following, we compare the performance of Hessenberg and Arnoldi methods in conjunction with the Tikhonov regularization method. The corresponding methods are respectively called by Hessenberg-Tikhonov and Arnoldi-Tikhonov methods. For more details on the implementation of the Hessenberg and Arnoldi methods in conjunction with the Tikhonov regularization method, we refer the readers to [6, 7, 26].

In Table 1, we report the total required number of iterations and consumed CPU-time (in seconds) under “Iter” and “CPU(s)”, respectively. For more detail, we also disclose the relative error

$$\text{Err} := \frac{\|\mathcal{X}_{\mu_k, k} - \hat{\mathcal{X}}\|}{\|\hat{\mathcal{X}}\|},$$

where $\hat{\mathcal{X}}$ denotes the exact solution of the problem with error-free right-hand side tensor $\hat{\mathcal{G}}$ associated with \mathcal{G} , and $\mathcal{X}_{\mu_k, k}$ denotes the k -th computed approximation determined by the algorithms. We note that the deblurred images based on computed regularized solutions are obtained by reshaping the first column of $\mathbf{mat}(\mathcal{X}_{\mu_k, k})$. The regularization parameter μ_k is determined by the discrepancy principle, see [17, Chapter 7] for more details.

The initial approximate solution in all experiments is the zero tensor and the iterations were terminated once a maximum 60 number of iterations is reached or

the following condition holds

$$\frac{\|\mathcal{X}_{\mu_k, k} - \mathcal{X}_{\mu_{k-1}, k-1}\|}{\|\mathcal{X}_{\mu_{k-1}, k-1}\|} \leq \tau, \quad (20)$$

for a user-specified value of the parameter $\tau > 0$.

In Tables 1 and 2, the MATLAB function PSNR denotes the peak signal-to-noise ratio between the original and a blurred (or restored) image in decibels. The higher the PSNR is, the better is the quality of deblurred image.

We examine the following two test problems to compare the performance of the Hessenberg-Tikhonov and Arnoldi-Tikhonov methods to restore an image contaminated by blur and noise.

Example 1. We use the blur operator obtained by (19) and set $\tau = 5 \cdot 10^{-3}$ in (20). The results are reported for the following two cases:

Case I. `band` = 11 and $\sigma = 4$. The exact solution is the `rice` image from MATLAB.

Case II. `band` = 16 and $\sigma = 6$. The exact solution is the `airplane`[‡] image.

Both gray-scale images is represented by an array of 256×256 pixels. The original and blurred-noisy images are plotted in Figure 1 for further details. The blurred-noisy image G is obtained by reshaping the first column of `mat(9)`. The obtained regularized solutions are shown in Figure 1 for the noise level $\eta = 0.01$.

We report the numerical results for Example 1 in Table 1. As observed, here, both methods works well and determine suitable approximations for the exact solution for both noise levels. Overall, using Hessenberg-Tikhonov method leads to better results than Arnoldi-Tikhonov method in Case I. For the second case, the Hessenberg-Tikhonov method surpasses Arnoldi-Tikhonov method for the level of noise $\eta = 0.001$. For the noise of level $\eta = 0.01$, Hessenberg-Tikhonov method provides slightly more accurate solution than Arnoldi-Tikhonov method.

[‡]This image is available at <http://sipi.usc.edu/database/download.php?vol=misc&img=5.1.11>

Table 1: Comparison results for Example 1.

Case I					
Noise level (η)	Method	Iter	CPU(s)	Err	PSNR
0.01	Hessenberg-Tikhonov	6	0.6176	$1.3911 \cdot 10^{-1}$	23.4779
	Arnoldi-Tikhonov	9	1.3402	$1.8424 \cdot 10^{-1}$	21.3066
0.001	Hessenberg-Tikhonov	8	1.0822	$1.0984 \cdot 10^{-1}$	25.7960
	Arnoldi-Tikhonov	12	2.1456	$1.0965 \cdot 10^{-1}$	25.8106
Case II					
Noise level (η)	Method	Iter	CPU(s)	Err	PSNR
0.01	Hessenberg-Tikhonov	12	2.1506	$8.1811 \cdot 10^{-2}$	24.0145
	Arnoldi-Tikhonov	10	1.5554	$8.5115 \cdot 10^{-2}$	23.6695
0.001	Hessenberg-Tikhonov	6	0.5611	$6.8164 \cdot 10^{-2}$	25.5971
	Arnoldi-Tikhonov	10	1.4410	$6.9903 \cdot 10^{-2}$	25.3772

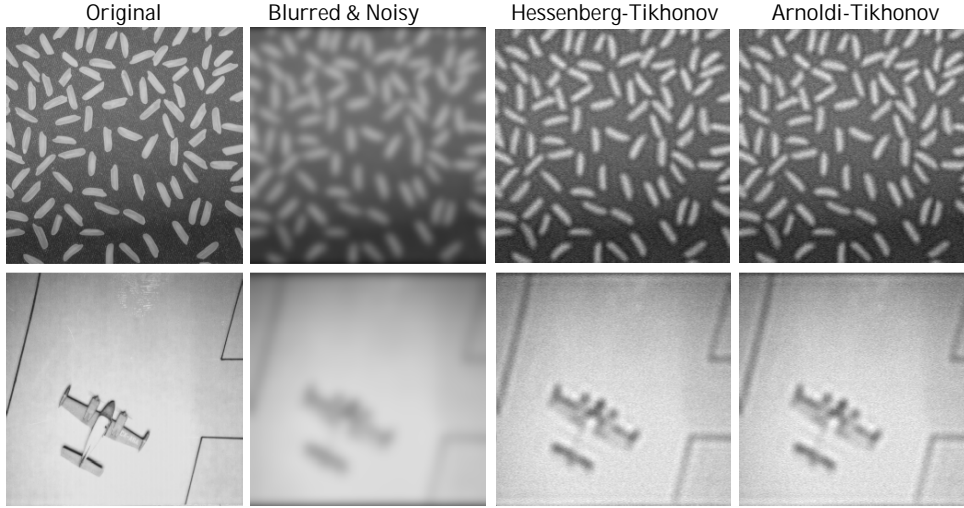


Figure 1: Original, Noisy and restored images using the Tikhonov regularization in conjunction with the Hessenberg and Arnoldi process.

Example 2. The exact solution of this test example is the boat[§] image, which is represented by an array of 512×512 pixels and displayed in Figure 2. We use the blur operator obtained by (19) with $\mathbf{band} = 3$, $\sigma = 4$. The iterations are terminated as soon as the stopping criterion (20) is satisfied where $\tau = 4 \cdot 10^{-2}$.

Results for this example are reported in Table 2. Both methods work well for both noise levels. The original, noisy and restored images are respectively plotted in Figures 2 and 3 for further details. As seen, here, Hessenberg-Tikhonov method consumes slightly less CPU-time (in seconds) in comparison with Arnoldi-Tikhonov

[§]This image is available at <https://sipi.usc.edu/database/download.php?vol=misc&img=boat.512>

method. The provided approximate solutions by Hessenberg-Tikhonov are also a bit more accurate.

Table 2: Comparison results for Example 2.

Noise level (η)	Method	Iter	CPU(s)	Err	PSNR
0.01	Hessenberg-Tikhonov	4	5.5581	$1.2472 \cdot 10^{-1}$	23.5330
	Arnoldi-Tikhonov	4	5.6118	$1.2915 \cdot 10^{-1}$	23.2042
0.001	Hessenberg-Tikhonov	4	5.5583	$1.0742 \cdot 10^{-1}$	24.8391
	Arnoldi-Tikhonov	4	5.6160	$1.1840 \cdot 10^{-1}$	23.9633



Figure 2: Exact image (left) and contaminated image (right).

Figure 3: Restored images using the Tikhonov regularization in conjunction with the Hessenberg and Arnoldi process for noise of level 0.01.

5. Conclusions

The Hessenberg method was considered to solve a general class of linear operator equations. It was shown that at each iterate, the Hessenberg method produces approximate solution satisfying an optimality property. The performance of the Hessenberg method had not been previously reported in the literature for solving the tensor equation $\mathcal{A} *_c \mathcal{X} = \mathcal{G}$. Therefore, two image restoration test problems in the above form were taken from [20]. Numerical comparison results were reported between the Hessenberg and Arnoldi processes in conjunction with the well-known Tikhonov regularization technique.

Acknowledgments

The authors would like to sincerely thank anonymous referee for his/her careful reading of the manuscript and helpful suggestions.

References

- [1] S. AMINI, F. TOUTOUNIAN, M. GACHPAZAN, *The block CMRH method for solving nonsymmetric linear systems with multiple right-hand sides*, J. Comput. Appl. Math. **337** (2018), 166–174.
- [2] S. AMINI, F. TOUTOUNIAN, *Weighted and flexible versions of block CMRH method for solving nonsymmetric linear systems with multiple right-hand sides*, Comput. Math. Appl. **76** (2018), 2011–2021.
- [3] B. W. BADER, T. G. KOLDA, *MATLAB Tensor Toolbox Version 2.5*, <http://www.sandia.gov/tgkolda/TensorToolbox>.
- [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] F. P. A. BEIK, F. S. MOVAHED, S. AHMADI-ASL, *On the Krylov subspace methods based on tensor format for positive definite Sylvester tensor equations*, Numer. Linear Algebra Appl. **23** (2016), 444–466.
- [6] F.P.A. BEIK, M. NAJAFI-KALYANI, *A preconditioning technique in conjunction with Krylov subspace methods for solving multilinear systems*, Appl. Math. Lett. **116** (2021) 107051.
- [7] F. P. A. BEIK, M. NAJAFI-KALYANI, L. REICHEL, *Iterative Tikhonov regularization of tensor equations based on the Arnoldi process and some of its generalizations*, Appl. Numer. Math. **151** (2020), 425–447.
- [8] M. BOLTEN, K. KARSTEN KAHL, S. SOKOLOVIC, *Multigrid methods for tensor structured Markov chains with low rank approximation*, SIAM J. Sci. Comput. **38** (2016), A649–A667.
- [9] M. BRAZELL, N. LI, NAVASCA, C. TAMON, *Solving multilinear systems via tensor inversion*, SIAM J. Matrix Anal. Appl. **34** (2013), 542–570.
- [10] Z. CHEN, L. Z. LU, *A projection method and Kronecker product preconditioner for solving Sylvester tensor equations*, Sci. China Math. **55** (2012) 1281–1292.
- [11] A. CICHOCKI, R. ZDUNEK, A.H. PHAN, S.I. AMARI, *Nonnegative Matrix and Tensor Factorizations: Applications to Exploratory Multi-way Data Analysis and Blind Source Separation*, 1st ed. John Wiley & Sons, 2009.

- [12] M. EL GUIDE, A. EL ICHI, K. JBILOU, F. P. A. BEIK, *Tensor Krylov subspace methods via the Einstein product with applications to image and video processing*, Appl. Numer. Math. **181** (2022), 347–363.
- [13] X.-M. GU, T.-Z. HUANG, G. YIN, B. CARPENTIERI, C. WEN, L. DU, *Restarted Hessenberg method for solving shifted nonsymmetric linear systems*, J. Comput. Appl. Math. **331** (2018), 166–177.
- [14] X.-M. GU, T.-Z. HUANG, B. CARPENTIERI, A. IMAKURA, K. ZHANG, L. DU, *Efficient variants of the CMRH method for solving a sequence of multi-shifted non-Hermitian linear systems simultaneously*, J. Comput. Appl. Math. **375** (2020), 112788.
- [15] X.-M. GU, S.-L. LEI, K. ZHANG, Z.-LI. SHEN, C. WEN, B. CARPENTIERI, *A Hessenberg-type algorithm for computing PageRank problems*, Numer. Algorithms, **89** (2022), 1845–1863.
- [16] L. GUO, X.-L. ZHAO, X.-M. GU, Y.-L. ZHAO, Y.-B. ZHENG, T.-Z. HUANG, *Three-dimensional fractional total variation regularized tensor optimized model for image deblurring*, Appl. Math. Comput. **404** (2021), 126224.
- [17] P.C. HANSEN, *Rank-Deficient and Discrete Ill-Posed Problems*, SIAM, Philadelphia, 1998.
- [18] M. HEYOUNI, *The global Hessenberg and CMRH methods for linear systems with multiple right-hand sides*, Numer. Algorithms, **26** (2001), 317–332.
- [19] B. HUANG, Y. XIE, C. MA, *Krylov subspace methods to solve a class of tensor equations via the Einstein product*, Numer. Linear Algebra Appl. **26** (2019), e2254.
- [20] E. KERNFELD, M. E. KILMER, S. AERON, *Tensor-tensor products with invertible linear transforms*, Linear Algebra Appl. **485** (2015), 545–570.
- [21] M. E. KILMER, C. D. MARTIN, *Factorization strategies for third-order tensors*, Linear Algebra Appl. **435** (2011), 641–658.
- [22] T. KOLDA, B.W. BADER, *Tensor decompositions and applications*, SIAM Rev. **51** (2009), 455–500.
- [23] D. KRESSNER, C. TOBLER, *Low-rank tensor Krylov subspace methods for parametrized linear systems*, SIAM J. Matrix Anal. Appl. **32** (2011), 1288–1316.
- [24] A. MALEK, Z. K. BOJDI, P. N. N. GOLBARG, *Solving fully three-dimensional microscale dual phase lag problem using mixed-collocation finite difference discretization*, J. Heat Transfer, **134** (2012), 094504.
- [25] A. MALEK, S. H. M. MASULEH, *Mixed collocation-finite difference method for 3D microscopic heat transport problems*, J. Comput. Appl. Math. **217** (2008), 137–147.
- [26] M. NAJAFI-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.
- [27] E. NEWMAN, M. E. KILMER, *Nonnegative tensor patch dictionary approaches for image compression and deblurring applications*, SIAM J. Imaging Sci. **13** (2020), 1084–1112.
- [28] L. QI, Z. LUO, *Tensor Analysis: Spectral Theory and Special Tensors*, SIAM, Philadelphia, 2017.
- [29] H. SADOK, *CMRH: A new method for solving nonsymmetric linear systems based on the Hessenberg reduction algorithm*, Numer. Algorithms, **20** (1999), 303–321.
- [30] L. REICHEL, U. O. UGWU, *The tensor Golub-Kahan-Tikhonov method applied to the solution of ill-posed problems with a t-product structure*, Numer. Linear Algebra Appl. **29** (2022), e2412.
- [31] L. REICHEL, U. O. UGWU, *Tensor Arnoldi-Tikhonov and GMRES-Type methods for ill-Posed problems with a t-product structure*, J. Sci. Comput. **90** (2022), Article number: 59.

- [32] L. REICHEL, U. O. UGWU, *Weighted tensor Golub–Kahan–Tikhonov–type methods applied to image processing using a t -product*, J. Comput. Appl. Math. **415** (2022), 114488.

Appendix A.

In this part, we summarize the Hessenberg and Arnoldi processes. Brief discussions are also included to compare the computational costs of these processes. In addition, we recall the presented algorithm in [6] for finding the set of indices for which a tensor takes its maximum value in modulus.

Notice that the well-known Arnoldi process can be regarded as a special case of Algorithm 1 setting $\mathcal{V}_i = \mathcal{Y}_i$ for $i = 1, 2, \dots, m$. In practical implementation, in the Hessenberg process the tensor \mathcal{Y}_i is chosen as a tensor having only one nonzero entry being equal to one. We comment that the iterative methods based on the Hessenberg process are applied with the pivoting strategy to avoid a possible breakdown, for further details see [18, 29]. To this end, at each step of the Hessenberg process, Algorithm 2 is used for finding the set of indices corresponding to the maximum element (in modulus) of a tensor; see [26] for more details. Basically, in this case, Algorithm 1 reduces to Algorithm 3.

We finish this part by comparing the number of required operations for the Arnoldi process and Algorithm 3 at each step of computing the new approximation for the solution of (1). Evidently, the differences between number of operations in Arnoldi process and Algorithm 3 are in the requirement of using Algorithm 2, computing the values of β and $h_{i,j}$ for $i = 1, 2, \dots, m+1$ and $j = 1, 2, \dots, m$. At each step, the total number of operations of Arnoldi process is higher than Algorithm 3 due to the more expensive computational costs of $h_{i,j}$ in Arnoldi process. Basically, evaluation of each $h_{i,j}$ corresponds to computing an inner product of the form (3). Consequently, Lines 6 and 10 of Algorithm 3 demonstrate that each step of Arnoldi process is more expensive than Algorithm 3. Notice that the cost of implementing Algorithm 2 is negligible especially in the case that $N \ll \max(I_1, I_2, \dots, I_N)$.

Algorithm 1 Hessenberg process. [26]

Require: Input tensor \mathcal{V} and scalar $m \geq 1$ as the maximum allowed dimension of the Krylov subspace;

Ensure: The upper Hessenberg matrix $\bar{H}_m = [h_{i,j}]_{(m+1) \times m}$ and $(N+1)$ -mode tensor $\tilde{\mathcal{V}}_m$ with the column tensors $\mathcal{V}_1, \mathcal{V}_2, \dots, \mathcal{V}_m$.

- 1: Set $\beta = \langle \mathcal{V}, \mathcal{Y}_1 \rangle$ and $\mathcal{V}_1 = \mathcal{V}/\beta$.
 - 2: **for** $j = 1, 2, \dots, m$ **do**
 - 3: $\mathcal{W} = \mathcal{F}(\mathcal{V}_j)$;
 - 4: **for** $i = 1, 2, \dots, j$ **do**
 - 5: $h_{i,j} = \langle \mathcal{Y}_i, \mathcal{W} \rangle$;
 - 6: $\mathcal{W} = \mathcal{W} - h_{i,j}\mathcal{V}_i$;
 - 7: **end for**
 - 8: $h_{j+1,j} = \langle \mathcal{Y}_{j+1}, \mathcal{W} \rangle$. If $h_{j+1,j} = 0$, then stop;
 - 9: $\mathcal{V}_{j+1} = \mathcal{W}/h_{j+1,j}$;
 - 10: **end for**
-

Algorithm 2 Pivoting strategy for a τ -mode tensor [6, Algorithm 1]

Require: Input a tensor $\mathcal{R} \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_\tau}$;

Ensure: Index group $(i_1, i_2, \dots, i_\tau)$;

 1: $[\sim, j] = \max(|\text{vec}(\mathcal{R})|)$;

 2: **for** $i = 1 : (\tau - 1)$ **do**

 3: $i_{\tau-i+1} = \left\lceil \frac{j}{\prod_{\ell=1}^{\tau-i} I_\ell} \right\rceil$;

 4: $\ell = j - (i_{\tau-i+1} - 1) \prod_{\ell=1}^{\tau-i} I_\ell$;

 5: $j = \ell$;

 6: **end for**

 7: $i_1 = \ell - (i_2 - 1)I_1$;

Algorithm 3 Hessenberg_BTF process with maximum strategy. [26]

Require: Input an $I_1 \times I_2 \times \dots \times I_N$, tensor \mathcal{V} and the restart parameter m .

Ensure: The upper Hessenberg matrix $\bar{H}_m = [h_{i,j}]_{(m+1) \times m}$ and $(N + 1)$ -mode tensor $\tilde{\mathcal{V}}_m$ with the column tensors $\mathcal{V}_1, \mathcal{V}_2, \dots, \mathcal{V}_m$.

 1: Determine triple $(i_{1,0}, i_{2,0}, \dots, i_{N,0})$ using Algorithm 2 for the input \mathcal{V} ;

 2: Set $\beta = \mathcal{V}_{i_{1,0}, i_{2,0}, \dots, i_{N,0}}$; $\mathcal{V}_1 = \mathcal{V}/\beta$; and $p_{1,\eta} = i_{\eta,0}$ for $\eta = 1, 2, \dots, N$;

 3: **for** $j = 1, \dots, m$ **do**

 4: $\mathcal{U} = \mathcal{F}(\mathcal{V}_j)$;

 5: **for** $i = 1, \dots, j$ **do**

 6: $h_{i,j} = \mathcal{U}_{p_{i,1}, p_{i,2}, \dots, p_{i,N}}$;

 7: $\mathcal{U} = \mathcal{U} - h_{i,j} \mathcal{V}_i$;

 8: **end for**

 9: Determine triple $(i_{1,0}, i_{2,0}, \dots, i_{N,0})$ using Algorithm 2 for the input \mathcal{U} ;

 10: Set $h_{j+1,j} = \mathcal{U}_{i_{1,0}, i_{2,0}, \dots, i_{N,0}}$; $\mathcal{V}_{j+1} = \mathcal{U}/h_{j+1,j}$; $p_{j+1,\eta} = i_{\eta,0}$ for $\eta = 1, 2, \dots, N$;

 11: **end for**
