COMMUN. MATH. SCI.
Vol. 19, No. 8, pp. 2275–2290

GRADIENT-BASED ITERATIVE ALGORITHMS FOR THE TENSOR NEARNESS PROBLEMS ASSOCIATED WITH SYLVESTER TENSOR EQUATIONS*

MAOLIN LIANG† AND BING ZHENG‡

Abstract. This paper is concerned with the solution of the tensor nearness problem associated with the Sylvester tensor equation represented by the Einstein product. We first proposed a gradient-based iterative algorithm for the Sylvester tensor equation mentioned above, and then the solution to the tensor nearness problem under consideration can be obtained by finding the least F-norm solution of another Sylvester tensor equation with special initial iteration tensors. It is shown that the solution to the above tensor nearness problem can be derived within finite iteration steps for any initial iteration tensors in the absence of roundoff errors. The performed numerical experiments show that the algorithm we propose here is efficient.

Keywords. Sylvester tensor equation; least F-norm solution; tensor nearness problem.

AMS subject classifications. 15A69; 65F10.

1. Introduction

Tensors are multi-dimensional arrays [1]. An Nth-order and $I_1\times I_2\times\cdots\times I_N$-dimensional tensor over the real field $\mathbb{R}$, consisting of $I_1I_2\cdots I_N$ entries, can be represented as

$$\mathscr{A}=(\mathscr{A}_{i_1\ldots i_N}) \text{ with } \mathscr{A}_{i_1\ldots i_N}\in\mathbb{R},\ 1\le i_k\le I_k,\ k=1,2,\ldots,N.$$

The set of this kind of tensors is denoted by $\mathbb{R}^{I_1\times I_2\times\cdots\times I_N}$. For $\mathscr{A}\in\mathbb{R}^{I_1\times\cdots\times I_M\times J_1\times\cdots\times J_N}$ and $\mathscr{B}\in\mathbb{R}^{J_1\times\cdots\times J_N\times K_1\times\cdots\times K_L}$, the *Einstein product* [2] of $\mathscr{A}$ and $\mathscr{B}$, denoted by $\mathscr{A}*_N\mathscr{B}$, is defined by the operation $*_N$ via

$$(\mathscr{A}*_N\mathscr{B})_{i_1\ldots i_M k_1\ldots k_L}=\sum_{j_1,\ldots,j_N} a_{i_1\ldots i_M j_1\ldots j_N} b_{j_1\ldots j_N k_1\ldots\ k_L}.$$

Tensor models are employed in numerous disciplines addressing the problem of finding multilinear structure in multiway data-sets. In particular, tensor equations with Einstein product model many phenomena in engineering and science, including continuum physics and engineering, isotropic and anisotropic elasticity [3–5]. For example, by using the central difference approximation, the three-dimensional Poisson equations can be discretized as the following multilinear system [7]

$$\mathscr{A}*_3\mathscr{X}=\mathscr{B},\ \mathscr{X}\in\mathbb{R}^{N\times N\times N},$$

where tensors $\mathscr{A}\in\mathbb{R}^{N\times N\times N\times N\times N\times N}$, $\mathscr{B}\in\mathbb{R}^{N\times N\times N}$. The general form of the above tensor equation is as follows:

$$\mathscr{A}*_M\mathscr{X}=\mathscr{B}, \tag{1.1}$$

where $\mathscr{A}\in\mathbb{R}^{K_1\times\cdots\times K_P\times I_1\times\cdots\times I_M}$ and $\mathscr{B}\in\mathbb{R}^{K_1\times\cdots\times K_P\times J_1\times\cdots\times J_N}$ are given tensors, and $\mathscr{X}\in\mathbb{R}^{I_1\times\cdots\times I_M\times J_1\times\cdots\times J_N}$ is unknown. Brazell et. al [7] researched the tensor Equation

*Received: May 13, 2020; Accepted (in revised form): May 23, 2021. Communicated by Shi Jin.
†School of Mathematics and Statistics, Tianshui Normal University, Tianshui 741001, China (liangml2005@163.com).
‡School of Mathematics and Statistics, Lanzhou University, Lanzhou 730000, China (bzheng@lzu.edu.cn).

(1.1) and the associated least-square problem by introducing the notion of inverse or pseudo-inverse of a tensor. Recently, Sun et. al. [9] extended the inverse in [7] and put forward the concept of Moore-Penrose inverses of tensors which provides the way to represent the general solution of the tensor Equation (1.1) in the sense that it is consistent (namely, there exists a tensor $\mathscr{X}^*$ satisfying (1.1)). Besides, the authors also considered the Sylvester tensor equation

$$\mathscr{A} *_M \mathscr{X} + \mathscr{X} *_N \mathscr{C} = \mathscr{D}, \tag{1.2}$$

in which the tensors $\mathscr{A} \in \mathbb{R}^{I_1\times\cdots\times I_M\times I_1\times\cdots\times I_M}$, $\mathscr{C} \in \mathbb{R}^{J_1\times\cdots\times J_N\times J_1\times\cdots\times J_N}$ and $\mathscr{D} \in \mathbb{R}^{I_1\times\cdots\times I_M\times J_1\times\cdots\times J_N}$ are known, and $\mathscr{X} \in \mathbb{R}^{I_1\times\cdots\times I_M\times J_1\times\cdots\times J_N}$ is the one to be determined. Obviously, when $M = N = 1$, this tensor equation reduces to the well-known Sylvester matrix equation

$$AX + XC = D,$$

which arises from the finite element, finite difference or spectral method [3, 4], while the former also play an important role in discretization of linear partial differential equations in high dimension [5–7, 9].

The purpose of this paper is to solve the following constrained minimization problem related to the Sylvester tensor Equation (1.2):

$$\min_{\mathscr{A} *_M \mathscr{X} + \mathscr{X} *_N \mathscr{C} = \mathscr{D}} \|\mathscr{X} - \mathscr{X}_0\|, \tag{1.3}$$

where $\mathscr{X}_0 \in \mathbb{R}^{I_1\times\cdots\times I_M\times J_1\times\cdots\times J_N}$ is a given tensor, the symbol $\|\cdot\|$ denotes the Frobenius norm (F-norm for short) of a tensor. This problem is a natural generalization of the matrix nearness problem [13–16], low rank approximation problem [17–19] and tensor completion problem [20–22] equipped with F-norm and multilinear constraints. We call (1.3) the tensor nearness problem.

Under certain conditions, it turns out that the solution to the tensor nearness problem (1.3) is unique. When the tensor $\mathscr{C}$ in (1.3) vanishes, we have proved that the unique solution to the corresponding tensor nearness problem can be represented by means of the Moore-Penrose inverses of the known tensors [23]. Nevertheless, it is well-known that it is not easy to find the Moore-Penrose inverse of a tensor. In order to address the aforementioned problem efficiently, we first consider to solve the Sylevester tensor Equation (1.2) iteratively, and design a gradient-type algorithm for it. This approach is derived from the ones presented in [7, 8, 10–12]. Furthermore, the solution to the tensor nearness problem (1.3) can be gained by applying the proposed method to another Sylvester tensor equation, see Section 4 for details. Therefore, the method presented in this paper avoids the curse mentioned above, and thus can be regarded as a continuation of [23]. One can also refer to [27–30] for the latest developments on tensor equations. As far as we know, this is the first time that iterative method is considered to solve the tensor nearness problems.

The remainder of this paper is organized as follows. Section 2 reviews some notation and definitions related to tensors. Section 3 contains the gradient-based iterative algorithm for solving the tensor Equation (1.2) as well as its convergence analysis. It is theoretically shown that the proposed approach is capable of finding the solution of (1.2) within finite iteration steps for any initial iteration tensors. Especially, the least F-norm solution of which can also be derived by choosing appropriate initial iteration tensors. Section 4 is devoted to addressing the tensor nearness problem (1.3). Section 5 provides some numerical examples to illustrate the efficiency of the proposed iterative algorithms. Finally, a conclusion is appended to end this paper.

2. Preliminaries

Throughout this paper, tensors are denoted by calligraphic letters, e.g., $\mathscr{A}$, $\mathscr{B}$, $\mathscr{C}$; matrices are denoted by boldface capital letters, e.g., **A**, **B**, **C**; vectors are denoted by boldface lowercase letters, e.g., **a**, **b**, **c**; scalars are denoted by lowercase letters, e.g., a, b, c. For a higher-order tensor, subtensors are formed when a subset of the indices is fixed, and a colon is used to indicate all elements of a mode. For example, for a tensor $\mathscr{A}\in\mathbb{R}^{I\times J\times K}$, its column, row, and tube fibers are denoted by $\mathscr{A}(:,j,k)$, $\mathscr{A}(i,:,k)$ and $\mathscr{A}(i,j,:)$, respectively. Moreover, the horizontal, lateral, and frontal slices are represented by $\mathscr{A}(i,:,:)$, $\mathscr{A}(:,j,:)$ and $\mathscr{A}(:,:,k)$, respectively.

The following definitions and conclusions will be used later.

Definition 2.1 ([7]). *For $\mathscr{A}=(\mathscr{A}_{i_1\dots i_M j_1\dots j_N})\in\mathbb{R}^{I_1\times\cdots\times I_M\times J_1\times\cdots\times J_N}$, its transpose, denoted by $\mathscr{A}^T$, is a $J_1\times\cdots\times J_N\times I_1\times\cdots\times I_M$ tensor with the entries $\widehat{\mathscr{A}_{i_1\dots i_N j_1\dots j_M}}=\mathscr{A}_{j_1\dots j_M i_1\dots i_N}$.*

Particularly, if $\mathscr{A}=(\mathscr{A}_{i_1\dots i_M j_1\dots j_M})\in\mathbb{R}^{I_1\times\cdots\times I_M\times I_1\times\cdots\times I_M}$, the trace of $\mathscr{A}$, denoted by $tr(\mathscr{A})$, is defined as $tr(\mathscr{A})=\sum\limits_{i_1,\dots,i_M}\mathscr{A}_{i_1\dots i_M i_1\dots i_M}$.

By Definition 2.1, the inner product of two tensors $\mathscr{A},\mathscr{B}\in\mathbb{R}^{I_1\times\cdots\times I_M\times J_1\times\cdots\times J_N}$ is defined by $<\mathscr{A},\,\mathscr{B}>=tr(\mathscr{B}^T *_M \mathscr{A})$, which induces the F-norm of a tensor, i.e., $\|\mathscr{A}\|=\sqrt{<\mathscr{A},\,\mathscr{A}>}$. Especially, if $<\mathscr{A},\,\mathscr{B}>=0$, we say that the two tensors are orthogonal to each other. Furthermore, since

$$\begin{aligned}tr(\mathscr{B}^T *_M \mathscr{A}) &= \sum_{i_1\dots i_N}(\mathscr{B}^T *_M \mathscr{A})_{i_1\dots i_N i_1\dots i_N}\\ &= \sum_{i_1\dots i_N}\left(\sum_{j_1\dots j_M}\mathscr{B}^T_{i_1\dots i_N j_1\dots j_M}\mathscr{A}_{j_1\dots j_M i_1\dots i_N}\right)\\ &= \sum_{i_1\dots i_N}\sum_{j_1\dots j_M}\mathscr{B}_{j_1\dots j_M i_1\dots i_N}\mathscr{A}_{j_1\dots j_M i_1\dots i_N},\end{aligned}$$

and

$$\begin{aligned}tr(\mathscr{A}^T *_M \mathscr{B}) &= \sum_{k_1\dots k_N}(\mathscr{A}^T *_M \mathscr{B})_{k_1\dots k_N k_1\dots k_N}\\ &= \sum_{k_1\dots k_N}\left(\sum_{l_1\dots l_M}\mathscr{A}^T_{k_1\dots k_N l_1\dots l_M}\mathscr{B}_{l_1\dots l_M k_1\dots k_N}\right)\\ &= \sum_{k_1\dots k_N}\sum_{l_1\dots l_M}\mathscr{A}_{l_1\dots l_M k_1\dots k_N}\mathscr{B}_{l_1\dots l_M k_1\dots k_N},\end{aligned}$$

which deduce that

$$<\mathscr{A},\,\mathscr{B}>=<\mathscr{B},\,\mathscr{A}>. \tag{2.1}$$

Moreover, by simple algebra we can obtain the following results.

Lemma 2.1. *Let $\mathscr{A},\mathscr{B},\mathscr{C}\in\mathbb{R}^{I_1\times\cdots\times I_M\times I_1\times\cdots\times I_M}$ and $\alpha,\beta\in\mathbb{R}$, then*

(I) $tr(\alpha\cdot\mathscr{A}+\beta\cdot\mathscr{B})=\alpha\cdot tr(\mathscr{A})+\beta\cdot tr(\mathscr{B})$;

(II) $tr(\mathscr{A} *_M \mathscr{B} *_M \mathscr{C})=tr(\mathscr{B} *_M \mathscr{C} *_M \mathscr{A})=tr(\mathscr{C} *_M \mathscr{A} *_M \mathscr{B})$.

These formulas are important to simplify the proof of the relevant conclusions in this paper.

DEFINITION 2.2. *For tensor* $\mathscr{A}\in\mathbb{R}^{I_1\times\cdots\times I_M\times J_1\times\cdots\times J_N}$, *Vec*$(\mathscr{A})\in\mathbb{R}^{(I_1\cdot\ldots\cdot I_M)\times J_1\times\cdots\times J_N}$ *is obtained by lining up all the subtensors,* $\mathscr{A}(i_1,\ldots,i_M,:,\ldots,:)$ *with* $1\le i_j\le I_j$ *and* $j=1,2,\ldots,M$, *in a column; e.g., the* k*-th subblock of* $\mathscr{A}$ *is the subtensor* $\mathscr{A}(i_1,\ldots,i_M,:,\ldots,:)$ *satisfying* $k=\mathrm{ivec}(\mathbf{i},\mathbb{I})$, *where* $\mathrm{ivec}(\cdot)$ *is the index mapping function [24], i.e.,*

$$\mathrm{ivec}(\mathbf{i},\mathbb{I}) := i_1+\sum_{j=2}^{M}(i_j-1)\prod_{s=1}^{j-1}I_s \text{ and } \mathbb{I}=\{I_1,\ldots,I_M\}.$$

Specifically, if $\mathscr{A}\in\mathbb{R}^{2\times2\times2\times J_1\times\cdots\times J_N}$, then

$$\mathrm{Vec}(\mathscr{A})=\begin{bmatrix}\mathscr{A}(1,1,1,:,\ldots,:)\\ \mathscr{A}(2,1,1,:,\ldots,:)\\ \mathscr{A}(1,2,1,:,\ldots,:)\\ \mathscr{A}(2,2,1,:,\ldots,:)\\ \mathscr{A}(1,1,2,:,\ldots,:)\\ \mathscr{A}(2,1,2,:,\ldots,:)\\ \mathscr{A}(1,2,2,:,\ldots,:)\\ \mathscr{A}(2,2,2,:,\ldots,:)\end{bmatrix}.$$

We should mention that the definition of Vec is slightly different from that given in [9].

DEFINITION 2.3 ([9]). *The Kronecker product of* $\mathscr{A}=(\mathscr{A}_{i_1\ldots i_M j_1\ldots j_N})\in\mathbb{R}^{I_1\times\cdots\times I_M\times J_1\times\cdots\times J_N}$ *and* $\mathscr{B}\in\mathbb{R}^{K_1\times\cdots\times K_P\times L_1\times\cdots\times L_Q}$, *denoted by* $\mathscr{A}\otimes\mathscr{B}$, *is a 'Kr-block tensor', whose* (r,s)*-subblock is* $(\mathscr{A}_{i_1\ldots i_M j_1\ldots j_N}\mathscr{B})$ *in which* $r=\mathrm{ivec}(\mathbf{i},\mathbb{I})$ *and* $s=\mathrm{ivec}(\mathbf{j},\mathbb{J})$ *for* $\mathbb{J}=\{J_1,\ldots,J_N\}$.

From the definition of the Kronecker product of tensors, it has the following basic properties:

LEMMA 2.2 ([9, 25]). *Let* $\mathscr{A}\in\mathbb{R}^{I_1\times\cdots\times I_M\times I_1\times\cdots\times I_M}$, $\mathscr{B}\in\mathbb{R}^{I_1\times\cdots\times I_M\times J_1\times\cdots\times J_N}$, $\mathscr{C}\in\mathbb{R}^{J_1\times\cdots\times J_N\times J_1\times\cdots\times J_N}$ *and* $\mathscr{D}\in\mathbb{R}^{I_1\times\cdots\times I_M\times J_1\times\cdots\times J_N}$. *Then*

(I) $(\mathscr{B}+\mathscr{D})^T=\mathscr{B}^T+\mathscr{D}^T$.

(II) $(\mathscr{A}\otimes\mathscr{B})\otimes\mathscr{C}=\mathscr{A}\otimes(\mathscr{B}\otimes\mathscr{C})$.

(III) $(\mathscr{A}\otimes\mathscr{B})*_N(\mathscr{D}\otimes\mathscr{C})=(\mathscr{A}*_M\mathscr{D})\otimes(\mathscr{B}*_N\mathscr{C})$.

(IV) $\mathrm{Vec}(\mathscr{A}*_M\mathscr{B}*_N\mathscr{C})=(\mathscr{C}^T\otimes\mathscr{A})*_N\mathrm{Vec}(\mathscr{B})$.

DEFINITION 2.4 ([7]). *Define the transformation* ψ *from the tensor space* $\mathbb{R}^{I_1\times\cdots\times I_M\times J_1\times\cdots\times J_N}$ *to the matrix space* $\mathbb{R}^{(I_1\cdot\ldots\cdot I_M)\times(J_1\cdot\ldots\cdot J_N)}$ *as*

$$\begin{aligned}\psi:\ \mathbb{C}^{I_1\times\cdots\times I_M\times J_1\times\cdots\times J_N}&\longrightarrow\mathbb{C}^{(I_1\cdot\ldots\cdot I_M)\times(J_1\cdot\ldots\cdot J_N)}\\ \mathscr{A}_{i_1\ldots i_M j_1\ldots j_N}&\longrightarrow\boldsymbol{A}_{\mathrm{ivec}(\mathbf{i},\mathbb{I}),\ \mathrm{ivec}(\mathbf{j},\mathbb{J})}.\end{aligned}$$

Obviously, the transformation ψ is a bijection, which provides a way to unfold one tensor. For example, if $\mathscr{A}\in\mathbb{R}^{3\times3\times3\times3}$, each frontal slice $\mathscr{A}(:,:,k,l)$ with $k,l=1,2,3$ is a 3×3 matrix. If we partition the modes of the tensor $\mathscr{A}$ from the middle, then the vector $\mathrm{vec}(\mathscr{A}(:,:,k,l))$ corresponds to the $[k+3(l-1)]$-th column of the unfolding matrix

$\mathbf{A} = \psi(\mathscr{A})$, that is,

$$\mathbf{A} = \left[\begin{array}{c|c|c|c|c|c|c|c|c} \mathscr{A}_{1111} & \mathscr{A}_{1121} & \mathscr{A}_{1131} & \mathscr{A}_{1112} & \mathscr{A}_{1122} & \mathscr{A}_{1132} & \mathscr{A}_{1113} & \mathscr{A}_{1123} & \mathscr{A}_{1133} \\ \mathscr{A}_{2111} & \mathscr{A}_{2121} & \mathscr{A}_{2131} & \mathscr{A}_{2112} & \mathscr{A}_{2122} & \mathscr{A}_{2132} & \mathscr{A}_{2113} & \mathscr{A}_{2123} & \mathscr{A}_{2133} \\ \mathscr{A}_{3111} & \mathscr{A}_{3121} & \mathscr{A}_{3131} & \mathscr{A}_{3112} & \mathscr{A}_{3122} & \mathscr{A}_{3132} & \mathscr{A}_{3113} & \mathscr{A}_{3123} & \mathscr{A}_{3133} \\ \mathscr{A}_{1211} & \mathscr{A}_{1221} & \mathscr{A}_{1231} & \mathscr{A}_{1212} & \mathscr{A}_{1222} & \mathscr{A}_{1232} & \mathscr{A}_{1213} & \mathscr{A}_{1223} & \mathscr{A}_{1233} \\ \mathscr{A}_{2211} & \mathscr{A}_{2221} & \mathscr{A}_{2231} & \mathscr{A}_{2212} & \mathscr{A}_{2222} & \mathscr{A}_{2232} & \mathscr{A}_{2213} & \mathscr{A}_{2223} & \mathscr{A}_{2233} \\ \mathscr{A}_{3211} & \mathscr{A}_{3221} & \mathscr{A}_{3231} & \mathscr{A}_{3212} & \mathscr{A}_{3222} & \mathscr{A}_{3232} & \mathscr{A}_{3213} & \mathscr{A}_{3223} & \mathscr{A}_{3233} \\ \mathscr{A}_{1311} & \mathscr{A}_{1321} & \mathscr{A}_{1331} & \mathscr{A}_{1312} & \mathscr{A}_{1322} & \mathscr{A}_{1332} & \mathscr{A}_{1313} & \mathscr{A}_{1323} & \mathscr{A}_{1333} \\ \mathscr{A}_{2311} & \mathscr{A}_{2321} & \mathscr{A}_{2331} & \mathscr{A}_{2312} & \mathscr{A}_{2322} & \mathscr{A}_{2332} & \mathscr{A}_{2313} & \mathscr{A}_{2323} & \mathscr{A}_{2333} \\ \mathscr{A}_{3311} & \mathscr{A}_{3321} & \mathscr{A}_{3331} & \mathscr{A}_{3312} & \mathscr{A}_{3322} & \mathscr{A}_{3332} & \mathscr{A}_{3313} & \mathscr{A}_{3323} & \mathscr{A}_{3333} \end{array}\right].$$

From the definition of ψ, one can observe that the entry $\mathscr{A}_{i_1\ldots i_M j_1\ldots j_N}$ of the tensor $\mathscr{A} \in \mathbb{R}^{I_1\times\cdots\times I_M\times J_1\times\cdots\times J_N}$ is exactly the $(\mathrm{ivec}(\mathbf{i},\mathbb{I}),\mathrm{ivec}(\mathbf{j},\mathbb{J}))$-element of the image matrix $\psi(\mathscr{A})$. Thus, the *identity tensor* of size $I_1\times\cdots\times I_M\times I_1\times\cdots\times I_M$, denoted by $\mathscr{I}$, consists of the entries

$$\mathscr{I}_{i_1\ldots i_M j_1\ldots j_M} = \prod_{k=1}^{M}\delta_{i_k j_k} \text{ with } \delta_{i_k j_k} = \begin{cases} 1, & \text{if } i_k = j_k, \\ 0, & \text{if } i_k \neq j_k. \end{cases}$$

For ease of reading, we recall the concept of the Moore-Penrose of tensors [9, 23].

Definition 2.5. *Let $\mathscr{A} \in \mathbb{C}^{I_1\times\cdots\times I_M\times J_1\times\cdots\times J_N}$, then the tensor $\mathscr{X} \in \mathbb{C}^{J_1\times\cdots\times J_N\times I_1\times\cdots I_M}$, satisfying the following tensor equations*

$$(1)\ \mathscr{A} *_N \mathscr{X} *_M \mathscr{A} = \mathscr{A}, \qquad (2)\ \mathscr{X} *_M \mathscr{A} *_N \mathscr{X} = \mathscr{X},$$
$$(3)\ (\mathscr{A} *_N \mathscr{X})^H = \mathscr{A} *_N \mathscr{X}, \quad (4)\ (\mathscr{X} *_M \mathscr{A})^H = \mathscr{X} *_M \mathscr{A},$$

is called the Moore-Penrose inverse of the tensor $\mathscr{A}$, denoted by $\mathscr{A}^\dagger$.

In addition, for tensor $\mathscr{A} \in \mathbb{R}^{I_1\times\cdots\times I_M\times J_1\times\cdots\times J_N}$, its range space is defined by

$$R(\mathscr{A}) = \{\ \mathscr{Y} \mid \mathscr{Y} = \mathscr{A} *_N \mathscr{X},\ \forall\ \mathscr{X} \in \mathbb{R}^{J_1\times\cdots\times J_N}\}.$$

3. Iterative algorithm for the Sylvester tensor Equation (1.2)

In order to solve the tensor nearness problem (1.3), we, in this section, present the gradient-based iterative algorithm for solving the Sylvester tensor Equation (1.2), and then analyze its convergence.

The iterative algorithm for solving the tensor Equation (1.2) is described as below:

Algorithm 3.1

Step 1: Input $\mathscr{A} \in \mathbb{R}^{I_1\times\cdots\times I_M\times I_1\times\cdots\times I_M}$, $\mathscr{C} \in \mathbb{R}^{J_1\times\cdots\times J_N\times J_1\times\cdots\times J_N}$, $\mathscr{D} \in \mathbb{R}^{I_1\times\cdots\times I_M\times J_1\times\cdots\times J_N}$, and an initial tensor $\mathscr{X}^{(1)} \in \mathbb{R}^{I_1\times\cdots\times I_M\times J_1\times\cdots\times J_N}$.

Step 2: Compute $\mathscr{R}^{(1)} = \mathscr{D} - \mathscr{A} *_M \mathscr{X}^{(1)} - \mathscr{X}^{(1)} *_N \mathscr{C}$, and
$$\mathscr{P}^{(1)} = \mathscr{A}^T *_M \mathscr{R}^{(1)} + \mathscr{R}^{(1)} *_N \mathscr{C}^T.$$

Step 3: Compute $\mathscr{X}^{(k+1)} = \mathscr{X}^{(k)} + \dfrac{\|\mathscr{R}^{(k)}\|^2}{\|\mathscr{P}^{(k)}\|^2}\ \mathscr{P}^{(k)}$.

Step 4: Compute $\mathscr{R}^{(k+1)} = \mathscr{D} - \mathscr{A} *_M \mathscr{X}^{(k+1)} - \mathscr{X}^{(k+1)} *_N \mathscr{C}$, and
$$\mathscr{P}^{(k+1)} = \mathscr{A}^T *_M \mathscr{R}^{(k+1)} + \mathscr{R}^{(k+1)} *_N \mathscr{C}^T + \frac{\|\mathscr{R}^{(k+1)}\|^2}{\|\mathscr{R}^{(k)}\|^2}\ \mathscr{P}^{(k)}.$$

If $\mathscr{R}^{(k+1)} = 0$, or $\mathscr{R}^{(k+1)} \neq 0$, $\mathscr{P}^{(k)} = 0$, stop; otherwise, goto Step 3.

In what follows, we show that the sequence $\{\mathscr{X}^{(k)}\}$ generated by Algorithm 3.1 converges to a solution of (1.2) within finite iteration steps in the absence of roundoff errors for any initial iteration tensor $\mathscr{X}^{(1)}$. For ease of expression, denote

$$\alpha(k) := \frac{\|\mathscr{R}^{(k)}\|^2}{\|\mathscr{P}^{(k)}\|^2}, \quad \beta(k+1) := \frac{\|\mathscr{R}^{(k+1)}\|^2}{\|\mathscr{R}^{(k)}\|^2}.$$

Lemma 3.1. *Let $\{\mathscr{R}^{(i)}\}$ and $\{\mathscr{P}^{(i)}\}$ $(i=1,2,\ldots)$ be the sequences generated by Algorithm 3.1. Then, for $j \geq 2$, it holds that*

$$\begin{aligned} tr\left(\mathscr{R}^{(i+1)^T} *_M \mathscr{R}^{(j)}\right) &= tr\left(\mathscr{R}^{(i)^T} *_M \mathscr{R}^{(j)}\right) - \alpha(i)\cdot tr\left(\mathscr{P}^{(i)^T} *_M \mathscr{P}^{(j)}\right) \\ &\quad + \alpha(i)\cdot\beta(j)\cdot tr\left(\mathscr{P}^{(i)^T} *_M \mathscr{P}^{(j-1)}\right). \end{aligned} \tag{3.1}$$

Proof. By the Steps 3 and 4 of Algorithm 3.1, we have

$$\mathscr{R}^{(k+1)} = \mathscr{R}^{(k)} - \alpha(k)\cdot(\mathscr{A} *_M \mathscr{P}^{(k)} + \mathscr{P}^{(k)} *_N \mathscr{C}), \tag{3.2}$$

and then

$$\begin{aligned} & tr\left(\mathscr{R}^{(i+1)^T} *_M \mathscr{R}^{(j)}\right) \\ &= tr\left((\mathscr{R}^{(i)} - \alpha(i)\cdot(\mathscr{A} *_M \mathscr{P}^{(i)} + \mathscr{P}^{(i)} *_N \mathscr{C}))^T *_M \mathscr{R}^{(j)}\right) \\ &= tr\left(\mathscr{R}^{(i)^T} *_M \mathscr{R}^{(j)}\right) - \alpha(i)\cdot tr\left((\mathscr{A} *_M \mathscr{P}^{(i)} + \mathscr{P}^{(i)} *_N \mathscr{C})^T *_M \mathscr{R}^{(j)}\right) \\ &= tr\left(\mathscr{R}^{(i)^T} *_M \mathscr{R}^{(j)}\right) - \alpha(i)\cdot tr\left(\mathscr{P}^{(i)^T} *_M (\mathscr{A}^T *_M \mathscr{R}^{(j)} + \mathscr{R}^{(j)} *_N \mathscr{C}^T)\right) \\ &= tr\left(\mathscr{R}^{(i)^T} *_M \mathscr{R}^{(j)}\right) - \alpha(i)\cdot tr\left(\mathscr{P}^{(i)^T} *_M (\mathscr{P}^{(j)} - \beta(j)\cdot\mathscr{P}^{(j-1)})\right), \end{aligned}$$

which implies that the equality (3.1) holds true. □

The next lemma reveals the orthogonality of the sequences $\{\mathscr{R}^{(i)}\}$ and $\{\mathscr{P}^{(i)}\}$ generated by Algorithm 3.1, which is similar to the classical conjugate gradient method [26].

Lemma 3.2. *Let $\{\mathscr{R}^{(i)}\}$ and $\{\mathscr{P}^{(i)}\}$ $(i=1,2,\ldots)$ be the sequences generated by Algorithm 3.1. Then*

$$tr\left(\mathscr{R}^{(i)^T} *_M \mathscr{R}^{(j)}\right) = 0, \quad tr\left(\mathscr{P}^{(i)^T} *_M \mathscr{P}^{(j)}\right) = 0, \quad i,j = 1,2,\ldots,t\,(t \geq 2), i \neq j. \tag{3.3}$$

Proof. We prove (3.3) by induction.

By (2.1), $tr\left(\mathscr{R}^{(i)^T} *_M \mathscr{R}^{(j)}\right) = tr\left(\mathscr{R}^{(j)^T} *_M \mathscr{R}^{(i)}\right)$, so we only consider the case: $i \geq j$.

When $t=2$, from Algorithm 3.1, we obtain

$$\begin{aligned} & tr\left(\mathscr{R}^{(2)^T} *_M \mathscr{R}^{(1)}\right) \\ &= tr\left(\mathscr{R}^{(1)^T} *_M \mathscr{R}^{(1)}\right) - \alpha(1)\cdot tr\left((\mathscr{A} *_M \mathscr{P}^{(1)} + \mathscr{P}^{(1)} *_N \mathscr{C})^T *_M \mathscr{R}^{(1)}\right) \\ &= tr\left(\mathscr{R}^{(1)^T} *_M \mathscr{R}^{(1)}\right) - \alpha(1)\cdot tr\left(\mathscr{P}^{(1)^T} *_M (\mathscr{A}^T *_M \mathscr{R}^{(1)} + \mathscr{R}^{(1)} *_N \mathscr{C}^T)\right) \end{aligned}$$

$$= tr\left(\mathscr{R}^{(1)^T} *_M \mathscr{R}^{(1)}\right) - \alpha(1)\cdot tr\left(\mathscr{P}^{(1)^T} *_M \mathscr{P}^{(1)}\right)$$
$$= 0, \tag{3.4}$$

and

$$tr\left(\mathscr{P}^{(2)^T} *_M \mathscr{P}^{(1)}\right)$$
$$= tr\left((\mathscr{A}^T *_M \mathscr{R}^{(2)} + \mathscr{R}^{(2)} *_N \mathscr{C}^T + \beta(2)\cdot \mathscr{P}^{(1)})^T *_M \mathscr{P}^{(1)}\right)$$
$$= tr\left(\mathscr{R}^{(2)^T} *_M (\mathscr{A} *_M \mathscr{P}^{(1)} + \mathscr{P}^{(1)} *_N \mathscr{C})\right) + \beta(2)\cdot tr\left(\mathscr{P}^{(1)^T} *_M \mathscr{P}^{(1)}\right)$$
$$= tr\left(\mathscr{R}^{(2)^T} *_M (\mathscr{R}^{(1)} - \mathscr{R}^{(2)})\cdot \frac{1}{\alpha(1)}\right) + \beta(2)\cdot tr\left(\mathscr{P}^{(1)^T} *_M \mathscr{P}^{(1)}\right)$$
$$= -\frac{1}{\alpha(1)}\cdot tr\left(\mathscr{R}^{(2)^T} *_M \mathscr{R}^{(2)}\right) + \beta(2)\cdot tr\left(\mathscr{P}^{(1)^T} *_M \mathscr{P}^{(1)}\right)$$
$$= 0. \tag{3.5}$$

Suppose that (3.3) holds for $t = s$, that is,

$$tr\left(\mathscr{R}^{(s)^T} *_M \mathscr{R}^{(j)}\right) = 0, tr\left(\mathscr{P}^{(s)^T} *_M \mathscr{P}^{(j)}\right) = 0, j = 1,2,\ldots,s-1.$$

In view of Lemma 3.1, when $t = s+1$, we have

$$tr\left(\mathscr{R}^{(s+1)^T} *_M \mathscr{R}^{(s)}\right) = tr\left(\mathscr{R}^{(s)^T} *_M \mathscr{R}^{(s)}\right) - \alpha(s)\cdot tr\left(\mathscr{P}^{(s)^T} *_M \mathscr{P}^{(s)}\right)$$
$$+ \alpha(s)\cdot\beta(s)\cdot tr\left(\mathscr{P}^{(s)^T} *_M \mathscr{P}^{(s-1)}\right)$$
$$= tr\left(\mathscr{R}^{(s)^T} *_M \mathscr{R}^{(s)}\right) - \alpha(s)\cdot tr\left(\mathscr{P}^{(s)^T} *_M \mathscr{P}^{(s)}\right)$$
$$= 0, \tag{3.6}$$

and

$$tr\left(\mathscr{P}^{(s+1)^T} *_M \mathscr{P}^{(s)}\right)$$
$$= tr\left((\mathscr{A}^T *_M \mathscr{R}^{(s+1)} + \mathscr{R}^{(s+1)} *_N \mathscr{C}^T + \beta(s+1)\cdot \mathscr{P}^{(s)})^T *_M \mathscr{P}^{(s)}\right)$$
$$= tr\left(\mathscr{R}^{(s+1)^T} *_M (\mathscr{R}^{(s)} - \mathscr{R}^{(s+1)})\cdot \frac{1}{\alpha(s)}\right) + \beta(s+1)\cdot tr\left(\mathscr{P}^{(s)^T} *_M \mathscr{P}^{(s)}\right)$$
$$= -\frac{1}{\alpha(s)}\cdot tr\left(\mathscr{R}^{(s+1)^T} *_M \mathscr{R}^{(s+1)}\right) + \beta(s+1)\cdot tr\left(\mathscr{P}^{(s)^T} *_M \mathscr{P}^{(s)}\right)$$
$$= 0. \tag{3.7}$$

Now we consider the cases $j = 1,2,\ldots,s-1$. In fact, when $j = 1$, similar to the proofs of (3.4) and (3.5), we have

$$tr\left(\mathscr{R}^{(s+1)^T} *_M \mathscr{R}^{(1)}\right) = tr\left((\mathscr{R}^{(s)} - \alpha(s)\cdot(\mathscr{A} *_M \mathscr{P}^{(s)} + \mathscr{P}^{(s)} *_N \mathscr{C}))^T *_M \mathscr{R}^{(1)}\right)$$
$$= -\alpha(s)\cdot tr\left(\mathscr{P}^{(s)^T} *_M (\mathscr{A}^T *_M \mathscr{R}^{(1)} + \mathscr{R}^{(1)} *_M \mathscr{C}^T)\right)$$
$$= -\alpha(s)\cdot tr\left(\mathscr{P}^{(s)^T} *_M \mathscr{P}^{(1)}\right)$$

$$=0, \tag{3.8}$$

and

$$\begin{aligned}
&tr\left(\mathscr{P}^{(s+1)^T} *_M \mathscr{P}^{(1)}\right)\\
&= tr\left((\mathscr{A}^T *_M \mathscr{R}^{(s+1)} + \mathscr{R}^{(s+1)} *_N \mathscr{C}^T + \beta(s+1)\cdot \mathscr{P}^{(s)})^T *_M \mathscr{P}^{(1)}\right)\\
&= tr\left(\mathscr{R}^{(s+1)^T} *_M (\mathscr{A} *_M \mathscr{P}^{(1)} + \mathscr{P}^{(1)} *_N \mathscr{C})\right)\\
&= -\frac{1}{\alpha(1)}\cdot tr\left(\mathscr{R}^{(s+1)^T} *_M (\mathscr{R}^{(1)} - \mathscr{R}^{(2)})\right)\\
&= 0.
\end{aligned} \tag{3.9}$$

When $2\le j\le s-1$, similar to the proofs of (3.6) and (3.7), using Lemma 3.1 once again, we can respectively deduce that

$$tr\left(\mathscr{R}^{(s+1)^T} *_M \mathscr{R}^{(j)}\right) = 0 \text{ and } tr\left(\mathscr{P}^{(s+1)^T} *_M \mathscr{P}^{(j)}\right) = 0,$$

which, together with (3.4)-(3.9), indicates that (3.3) holds. □

LEMMA 3.3. *Suppose that $\widetilde{\mathscr{X}}$ is an arbitrary solution of the tensor Equation (1.2), then the sequences $\{\mathscr{R}^{(k)}\}$ and $\{\mathscr{P}^{(k)}\}$ satisfy*

$$tr\left((\widetilde{\mathscr{X}} - \mathscr{X}^{(k)})^T *_M \mathscr{P}^{(k)}\right) = \|\mathscr{R}^{(k)}\|^2, \quad k=1,2,\ldots. \tag{3.10}$$

Proof. We prove (3.10) by induction as well.

When $k=1$, it follows from Algorithm 3.1 and Lemma 3.2 that

$$\begin{aligned}
tr\left((\widetilde{\mathscr{X}} - \mathscr{X}^{(1)})^T *_M \mathscr{P}^{(1)}\right) &= tr\left((\widetilde{\mathscr{X}} - \mathscr{X}^{(1)})^T *_M (\mathscr{A}^T *_M \mathscr{R}^{(1)} + \mathscr{R}^{(1)} *_M \mathscr{C}^T)\right)\\
&= tr\left((\mathscr{D} - \mathscr{A} *_M \mathscr{X}^{(1)} - \mathscr{X}^{(1)} *_N \mathscr{C})^T *_M \mathscr{R}^{(1)}\right)\\
&= tr\left(\mathscr{R}^{(1)^T} *_M \mathscr{R}^{(1)}\right)\\
&= \|\mathscr{R}^{(1)}\|^2.
\end{aligned} \tag{3.11}$$

Assume that (3.10) holds for $k=s$, then

$$\begin{aligned}
tr\left((\widetilde{\mathscr{X}} - \mathscr{X}^{(s+1)})^T *_M \mathscr{P}^{(s)}\right) &= tr\left((\widetilde{\mathscr{X}} - \mathscr{X}^{(s)} - \alpha(s)\cdot \mathscr{P}^{(s)})^T *_M \mathscr{P}^{(s)}\right)\\
&= tr\left((\widetilde{\mathscr{X}} - \mathscr{X}^{(s)})^T *_M \mathscr{P}^{(s)}\right) - \alpha(s)\cdot tr\left(\mathscr{P}^{(s)^T} *_M \mathscr{P}^{(s)}\right)\\
&= \|\mathscr{P}^{(s)}\|^2 - \alpha(s)\cdot tr\left(\mathscr{P}^{(s)^T} *_M \mathscr{P}^{(s)}\right)\\
&= 0.
\end{aligned}$$

Furthermore, we have

$$\begin{aligned}
&tr\left((\widetilde{\mathscr{X}} - \mathscr{X}^{(s+1)})^T *_M \mathscr{P}^{(s+1)}\right)\\
&= tr\left((\widetilde{\mathscr{X}} - \mathscr{X}^{(s+1)})^T *_M (\mathscr{A}^T *_M \mathscr{R}^{(s+1)} + \mathscr{R}^{(s+1)} *_N \mathscr{C}^T + \beta(s+1)\cdot \mathscr{P}^{(s)})\right)\\
&= tr\left((\mathscr{A} *_M (\widetilde{\mathscr{X}} - \mathscr{X}^{(s+1)}) + (\widetilde{\mathscr{X}} - \mathscr{X}^{(s+1)}) *_N \mathscr{C})^T *_M \mathscr{R}^{(s+1)}\right)
\end{aligned}$$

$$= tr\left((\mathscr{D} - \mathscr{A} *_M \mathscr{X}^{(s+1)} - \mathscr{X}^{(s+1)} *_N \mathscr{C})^T *_M \mathscr{R}^{(s+1)} \right)$$
$$= \|\mathscr{R}^{(s+1)}\|^2. \tag{3.12}$$

The proof is complete. □

Making use of Lemmas 3.2 and 3.3, we can prove the main result of this paper.

THEOREM 3.1. *If the tensor Equation (1.2) is consistent, then for any initial iteration tensor $\mathscr{X}^{(1)}$, its solution can be derived by Algorithm 3.1 within finite iteration steps.*

Proof. For simplicity, denote

$$m := I_1 \cdot \ldots \cdot I_M,\ n := J_1 \cdot \ldots \cdot J_N.$$

If $\mathscr{R}^{(k)} \neq 0$, $k = 1,2,\ldots,mn$, it follows from Lemma 3.3 that $\mathscr{P}^{(k)} \neq 0$, then one can compute $\mathscr{X}^{(mn+1)}$ and $\mathscr{R}^{(mn+1)}$ by Algorithm 3.1. Furthermore, from Lemma 3.2 we know that

$$tr\left(\mathscr{R}^{(mn+1)^T} *_M \mathscr{R}^{(k)} \right) = 0 \text{ and } tr\left(\mathscr{P}^{(mn+1)^T} *_M \mathscr{P}^{(l)} \right) = 0,$$

where $k,l = 1,2,\ldots,mn, k \neq l$. Since the sequence $\{\mathscr{R}^{(k)}\}$ is an orthogonal basis of tensor space $\mathbb{R}^{I_1\times\cdots\times I_M\times J_1\times\cdots\times J_N}$, which implies that $\mathscr{R}^{(mn+1)} = 0$, i.e., $\mathscr{X}^{(mn+1)}$ is a solution of (1.2). □

Moreover, according to the basic properties of Algorithm 3.1 mentioned above, we can show that the solvability of the tensor Equation (1.2) can be determined automatically during the iteration process.

THEOREM 3.2. *The tensor Equation (1.2) is inconsistent if and only if there exists a positive integer k_0 such that $\mathscr{R}^{(k_0)} \neq 0$ and $\mathscr{P}^{(k_0)} = 0$.*

Proof. If the tensor Equation (1.2) is inconsistent, it follows that $\mathscr{R}^{(k)} \neq 0$ for any k. Provided that $\mathscr{P}^{(k)} \neq 0$ for all positive integers k, then, from the proof of Theorem 3.1 we know that there must exist $\mathscr{X}^{(k)}$ satisfying (1.2), which contradicts the inconsistency. Conversely, if there is a positive integer k_0, such that $R_{k_0} \neq 0$ but $P_{k_0} = 0$, which contradicts Lemma 3.3, so the tensor Equation (1.2) is inconsistent. The proof is complete. □

In addition, since the tensor equation is always over-determined, we are often interested in the least F-norm solution. Next we can show that the least F-norm solution of the tensor Equation (1.2) can also be gained by means of Algorithm 3.1. We first prove the following lemma for this aim.

LEMMA 3.4. *Let $\mathscr{X}^*$ be a solution of the tensor Equation (1.1), then $\mathscr{X}^*$ is the unique least F-norm solution if $\mathscr{X}^* \in R(\mathscr{A}^T)$.*

Proof. For convenience of expression, we use the same symbol ψ to represent the unfoldings of different tensors, e.g., $\mathbf{A} = \psi(\mathscr{A})$, $\mathbf{B} = \psi(\mathscr{B})$ and $\mathbf{X} = \psi(\mathscr{X})$. We prove the conclusion by two steps:

Step (1) The tensor Equation (1.1) is equivalent to the matrix equation

$$\mathbf{AX} = \mathbf{B} \text{ with } \mathbf{X} \in \mathbb{R}^{m\times n}. \tag{3.13}$$

In fact, from the definition of Einstein product, we can respectively rewrite (1.1) and (3.13) in terms of the components as

$$(\mathscr{A} *_M \mathscr{X})_{k_1 \ldots k_P j_1 \ldots j_N} = \sum_{i_1,\ldots,i_M} \mathscr{A}_{k_1 \ldots k_P i_1 \ldots i_M} \mathscr{X}_{i_1 \ldots i_M j_1 \ldots j_N} = \mathscr{B}_{k_1 \ldots k_P j_1 \ldots j_N},$$

and

$$\sum_t \mathbf{A}_{pt} \mathbf{X}_{ts} = \mathbf{B}_{ps}.$$

Since ψ is a bijection, then there must exist, respectively, the unique index $\{k_1, \ldots, k_P\}$, $\{i_1, \ldots, i_M\}$ and $\{j_1, \ldots, j_N\}$ such that ivec$([k_1, \ldots, k_P], \mathbb{K}) = p$, ivec$([i_1, \ldots, i_M], \mathbb{I}) = t$, and ivec$([j_1, \ldots, j_N], \mathbb{J}) = s$ in which $\mathbb{K} = \{K_1, \ldots, K_P\}$. Therefore, the above two systems are equivalent.

Step (2) As is well-known [21], the least F-norm solution of matrix Equation (3.13) is $\widetilde{\mathbf{X}} = \mathbf{A}^{\dagger}\mathbf{B} \in R(\mathbf{A}^T)$, where the superscript $\dagger$ denotes the Moore-Penrose inverse of a matrix. In view of the uniqueness of the least F-norm solution, and together with the fact that ψ is a bijection, we complete the proof. ∎

Depending on the above lemma, we can prove the following theorem.

THEOREM 3.3. *Assume that the tensor Equation (1.2) is consistent, and let the initial iteration tensor $\mathscr{X}^{(1)} = \mathscr{A}^T *_M \mathscr{W} + \mathscr{W} *_N \mathscr{C}^T$ with arbitrary $\mathscr{W} \in \mathbb{R}^{I_1 \times \cdots \times I_M \times J_1 \times \cdots \times J_N}$, or especially, $\mathscr{X}^{(1)} = \mathscr{O} \in \mathbb{R}^{I_1 \times \cdots \times I_M \times J_1 \times \cdots \times J_N}$, then the solution generated by Algorithm 3.1 is the unique least F-norm solution.*

Proof. From Algorithm 3.1 and Theorem 3.1, it is known that if we choose the initial iteration tensor $\mathscr{X}^{(1)} = \mathscr{A}^T *_M \mathscr{W} + \mathscr{W} *_N \mathscr{C}^T$ for some tensor $\mathscr{W}$, then the approximate solution $\mathscr{X}^{(k)}$ of the tensor Equation (1.2) possesses the form $\mathscr{X}^{(k)} = \mathscr{A}^T *_M \mathscr{H} + \mathscr{H} *_N \mathscr{C}^T$ for some $\mathscr{H} \in \mathbb{R}^{I_1 \times \cdots \times I_M \times J_1 \times \cdots \times J_N}$. Using the definition of Vec and Lemma 2.2, we deduce that

$$\begin{aligned} \operatorname{Vec}\left(\mathscr{X}^{(k)}\right) &= \operatorname{Vec}\left(\mathscr{A}^T *_M \mathscr{H} + \mathscr{H} *_N \mathscr{C}^T\right) \\ &= \left(\mathscr{I}_1 \otimes \mathscr{A}^T + \mathscr{C} \otimes \mathscr{I}_2\right) *_N \operatorname{Vec}(\mathscr{H}) \\ &\in R\left(\left(\mathscr{I}_1 \otimes \mathscr{A} + \mathscr{C}^T \otimes \mathscr{I}_2\right)^T\right), \end{aligned} \tag{3.14}$$

where $\mathscr{I}_1$ and $\mathscr{I}_2$ are the identity tensors of size $J_1 \times \cdots \times J_N \times J_1 \times \cdots \times J_N$ and $I_1 \times \cdots \times I_M \times I_1 \times \cdots \times I_M$, respectively. On the other hand, by using the properties of the Kronecker product, one can demonstrate that (1.2) is equivalent to the tensor equation

$$(\mathscr{I}_1 \otimes \mathscr{A} + \mathscr{C}^T \otimes \mathscr{I}_2) *_N \operatorname{Vec}(\mathscr{X}) = \operatorname{Vec}(\mathscr{D}),$$

which, together with (3.14) and Lemma 3.4, implies that $\mathscr{X}^{(k)}$ is the least F-norm solution of the Sylvester tensor Equation (1.2). The proof is complete. ∎

4. Solving the tensor nearness problem

In this section, we apply Algorithm 3.1 to the solution of the tensor nearness problem (1.3). Suppose that the tensor Equation (1.2) is consistent, i.e., its solution set, denoted by Φ, is nonempty. It is easy to verify that the set Φ is a closed and convex set in the tensor space $\mathbb{R}^{I_1 \times \cdots \times I_M \times J_1 \times \cdots \times J_N}$, which reveals that the solution to the tensor nearness problem is unique, denoted by $\widehat{\mathscr{X}}$ for convenience.

We should point out that the unique solution $\widehat{\mathscr{X}}$ can also be derived by using Algorithm 3.1. Actually, noting the fact that to solve the tensor nearness problem with the given tensor $\mathscr{X}_0$ is equivalent to find the least F-norm solution (denoted by $\widehat{\mathscr{Y}}$) of the following Sylvester tensor equation

$$\mathscr{A} *_M \mathscr{Y} + \mathscr{Y} *_N \mathscr{C} = \widetilde{\mathscr{D}}, \tag{4.1}$$

where $\mathscr{Y} = \mathscr{X} - \mathscr{X}_0$ and $\widetilde{\mathscr{D}} = \mathscr{D} - \mathscr{A} *_M \mathscr{X}_0 - \mathscr{X}_0 *_N \mathscr{C}$, then, it follows from Theorem 3.3 that $\widehat{\mathscr{X}}$ can be obtained by applying Algorithm 3.1 to (4.1) with the initial iteration tensor $\mathscr{X}^{(1)} = \mathscr{A}^T *_M \mathscr{W} + \mathscr{W} *_N \mathscr{C}^T$ for some $\mathscr{W} \in \mathbb{R}^{I_1 \times \cdots \times I_M \times J_1 \times \cdots \times J_N}$, or especially, $\mathscr{X}^{(1)} = \mathscr{O} \in \mathbb{R}^{I_1 \times \cdots \times I_M \times J_1 \times \cdots \times J_N}$, i.e., the null tensor with zero elements. In this case, the nearness solution of (1.3) can be obtained by $\widehat{\mathscr{X}} = \widehat{\mathscr{Y}} + \mathscr{X}_0$.

5. Numerical experiments

In this section, we perform some numerical examples to illustrate the feasibility and effectiveness of the proposed algorithm in present paper. All computations were written using MATLAB (version R2016a) on a personal computer with 2.50GHz central processing unit (Intel(R) Core(TM) i5-3210M) and 4GB memory. Specially, all the tensor calculations in our tests were carried out with the Tensor Toolbox Version 2.6.[1] The iterations will be terminated if the norm of the residual, i.e., RES$= \|\mathscr{D} - \mathscr{A} *_M \mathscr{X}^{(k)} - \mathscr{X}^{(k)} *_N \mathscr{C}\| < \varepsilon = 1.0e-10$, or the number of iteration steps exceeds the maximum $k_{\max} = 1000$.

EXAMPLE 5.1. *Suppose the tensors $\mathscr{A} \in \mathbb{R}^{4\times3\times4\times3}$, $\mathscr{C} \in \mathbb{R}^{3\times3\times3\times3}$, $\mathscr{D} \in \mathbb{R}^{4\times3\times3\times3}$ in (1.2) are given as follows:*

$$\mathscr{A}(:,:,1,1) = \begin{bmatrix} 11 & 7 & 7 \\ -2 & 11 & -2 \\ 11 & -2 & 7 \\ -2 & 11 & -2 \end{bmatrix}, \mathscr{A}(:,:,2,1) = \begin{bmatrix} -2 & -2 & -2 \\ 3 & -2 & 3 \\ -2 & 3 & -2 \\ 3 & -2 & 3 \end{bmatrix},$$

$$\mathscr{A}(:,:,3,1) = \begin{bmatrix} 3 & -4 & -4 \\ -1 & 3 & -1 \\ 3 & -1 & -4 \\ -1 & 3 & -1 \end{bmatrix}, \mathscr{A}(:,:,4,1) = \begin{bmatrix} 2 & -9 & -9 \\ -6 & 2 & -6 \\ 2 & -6 & -9 \\ -6 & 2 & -6 \end{bmatrix},$$

$$\mathscr{A}(:,:,2,2) = \begin{bmatrix} -16 & 3 & 3 \\ -11 & -16 & -11 \\ -16 & -11 & 3 \\ -11 & -16 & -11 \end{bmatrix}, \mathscr{A}(:,:,1,2) = \begin{bmatrix} 0 & 7 & 7 \\ 11 & 0 & 11 \\ 0 & 11 & 7 \\ 11 & 0 & 11 \end{bmatrix},$$

$$\mathscr{A}(:,:,3,2) = \begin{bmatrix} -11 & 15 & 15 \\ 0 & -11 & 0 \\ -11 & 0 & 15 \\ 0 & -11 & 0 \end{bmatrix}, \mathscr{A}(:,:,4,2) = \begin{bmatrix} -4 & -2 & -2 \\ 16 & -4 & 16 \\ -4 & 16 & -2 \\ 16 & -4 & 16 \end{bmatrix},$$

$$\mathscr{A}(:,:,1,3) = \begin{bmatrix} 3 & -3 & -3 \\ 13 & 3 & 13 \\ 3 & 13 & -3 \\ 13 & 3 & 13 \end{bmatrix}, \mathscr{A}(:,:,2,3) = \begin{bmatrix} 26 & 0 & 0 \\ -4 & 26 & -4 \\ 26 & -4 & 0 \\ -4 & 26 & -4 \end{bmatrix},$$

[1]http://www.sandia.gov/tgkolda/TensorToolbox/index-2.6.html.

$$\mathscr{A}(:,:,3,3)=\begin{bmatrix}-4&1&1\\8&-4&8\\-4&8&1\\8&-4&8\end{bmatrix},\mathscr{A}(:,:,4,3)=\begin{bmatrix}2&-8&-8\\-16&2&-16\\2&-16&-8\\-16&2&-16\end{bmatrix};$$

$$\mathscr{C}(:,:,1,1)=\begin{bmatrix}10&0&6\\15&10&10\\10&15&10\end{bmatrix},\ \mathscr{C}(:,:,2,1)=\begin{bmatrix}6&-9&17\\-9&6&6\\6&-9&6\end{bmatrix},$$

$$\mathscr{C}(:,:,3,1)=\begin{bmatrix}4&-19&-3\\-14&4&4\\4&-14&4\end{bmatrix},\ \ \mathscr{C}(:,:,1,2)=\begin{bmatrix}9&-22&-8\\0&9&9\\9&0&9\end{bmatrix},$$

$$\mathscr{C}(:,:,2,2)=\begin{bmatrix}0&-9&-3\\-13&0&0\\0&-13&0\end{bmatrix},\ \mathscr{C}(:,:,3,2)=\begin{bmatrix}-7&-17&12\\6&-7&-7\\-7&6&-7\end{bmatrix},$$

$$\mathscr{C}(:,:,2,3)=\begin{bmatrix}5&-13&1\\-5&5&5\\5&-5&5\end{bmatrix},\qquad \mathscr{C}(:,:,1,3)=\begin{bmatrix}0&-3&4\\5&0&0\\0&5&0\end{bmatrix},$$

$$\mathscr{C}(:,:,3,3)=\begin{bmatrix}0&-12&3\\-1&0&0\\0&-1&0\end{bmatrix},$$

and the tensor $\mathscr{D}$ *is chosen such that* $\mathscr{D}=\mathscr{A}*_M\mathscr{X}^*+\mathscr{X}^**_N\mathscr{C}$ *with* $\mathscr{X}^*=\mathtt{reshape}(1:108,[4,3,3,3])\in\mathbb{R}^{4\times3\times3\times3}$.

In this case, the tensor Equation (1.2) is consistent and $\mathscr{X}^*$ is an exact solution. Applying Algorithm 3.1 with initial iteration tensor $\mathscr{X}^{(1)}=\mathscr{O}\in\mathbb{R}^{4\times3\times3\times3}$ to (1.2), we obtain the least F-norm solution, denoted by $\widetilde{\mathscr{X}}$, and the corresponding residual RES$=9.6392\text{e}-11$ after 86 iteration steps. In addition, since the conjugate gradient least squares algorithm (CGLS) has better performance than the bi-conjugate gradient algorithm and the bi-conjugate residual algorithm [30], we only compared Algorithm 3.1 (denoted by T-CG) with CGLS method for the above initial iteration tensor. In Figure 5.1, we plotted their convergent curves over the norm of the residual versus iteration number k, which imply that our method converges faster although the latter takes less iteration steps.

$$\widetilde{\mathscr{X}}(:,:,1,1)=\begin{bmatrix}42.9784&46.3496&53.2346\\53.0555&68.2438&49.0433\\54.4996&54.9506&59.0609\\61.1020&53.8303&73.3694\end{bmatrix},\widetilde{\mathscr{X}}(:,:,2,1)=\begin{bmatrix}32.9897&36.6903&42.0641\\38.3122&47.6399&40.5920\\39.5236&41.8336&45.8862\\43.1914&41.8239&53.2235\end{bmatrix},$$

$$\widetilde{\mathscr{X}}(:,:,3,1)=\begin{bmatrix}46.9887&50.6593&56.1705\\52.7434&62.6039&54.4513\\53.9760&56.1170&60.1748\\57.9106&56.0063&68.1459\end{bmatrix},\widetilde{\mathscr{X}}(:,:,1,2)=\begin{bmatrix}37.0000&41.0000&45.0000\\38.0000&42.0000&46.0000\\39.0000&43.0000&47.0000\\40.0000&44.0000&48.0000\end{bmatrix},$$

$$\widetilde{\mathscr{X}}(:,:,2,2)=\begin{bmatrix}50.9990 & 54.9690 & 59.1064\\ 52.4312 & 56.9640 & 59.8592\\ 53.4524 & 57.2834 & 61.2886\\ 54.7191 & 58.1824 & 62.9224\end{bmatrix},\widetilde{\mathscr{X}}(:,:,3,2)=\begin{bmatrix}41.0103 & 45.3097 & 47.9359\\ 37.6878 & 36.3601 & 51.4080\\ 38.4764 & 44.1664 & 48.1138\\ 36.8086 & 46.1761 & 42.7765\end{bmatrix},$$

$$\widetilde{\mathscr{X}}(:,:,1,3)=\begin{bmatrix}73.0000 & 77.0000 & 81.0000\\ 74.0000 & 78.0000 & 82.0000\\ 75.0000 & 79.0000 & 83.0000\\ 76.0000 & 80.0000 & 84.0000\end{bmatrix},\widetilde{\mathscr{X}}(:,:,2,3)=\begin{bmatrix}57.0144 & 61.4336 & 63.5103\\ 51.9630 & 48.5042 & 67.9711\\ 52.6669 & 59.0329 & 62.9594\\ 49.9320 & 61.4465 & 55.0871\end{bmatrix},$$

$$\widetilde{\mathscr{X}}(:,:,3,3)=\begin{bmatrix}59.0196 & 63.5885 & 64.9782\\ 51.8069 & 45.6842 & 70.6751\\ 52.4051 & 59.6161 & 63.5163\\ 48.3363 & 62.5345 & 52.4753\end{bmatrix}.$$

Next we consider the tensor nearness problem (1.3).

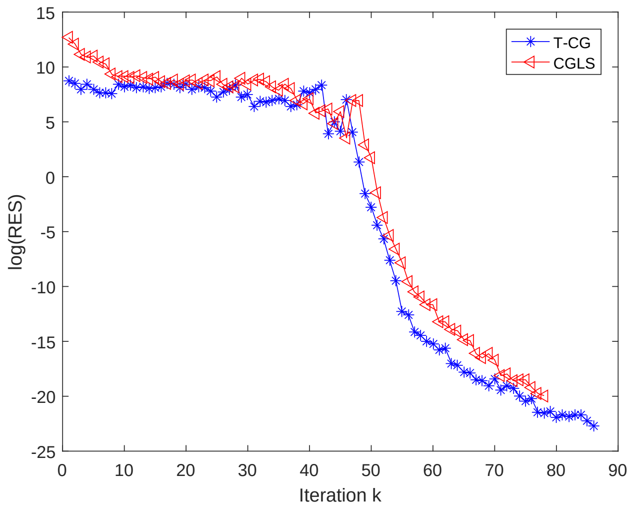

FIG. 5.1. *Convergence behavior of two algorithms for the tensor equation in Example 5.1.*

EXAMPLE 5.2. *Let the tensors $\mathscr{A}$, $\mathscr{C}$ and $\mathscr{D}$ in (1.2) be the same as in Example 5.1, and assume that the given tensor $\mathscr{X}_0$ is as follows:*

$$\mathscr{X}_0(:,:,1,1)=\begin{bmatrix}0 & 11 & -10\\ -7 & -1 & -4\\ -4 & -4 & 5\\ 7 & -6 & -5\end{bmatrix},\mathscr{X}_0(:,:,2,1)=\begin{bmatrix}4 & -11 & -12\\ -3 & 6 & -20\\ -13 & 4 & 0\\ 6 & -6 & -4\end{bmatrix},$$

$$\mathscr{X}_0(:,:,3,1)=\begin{bmatrix}-5 & 11 & 0\\ -16 & -2 & 4\\ 33 & -2 & -1\\ -16 & -8 & 9\end{bmatrix},\mathscr{X}_0(:,:,1,2)=\begin{bmatrix}7 & 6 & -4\\ 14 & -4 & -11\\ -13 & -32 & 9\\ -28 & 0 & -10\end{bmatrix},$$

$$\mathscr{X}_0(:,:,2,2)=\begin{bmatrix}-10 & 5 & 18\\ -6 & -8 & 8\\ -16 & -4 & 8\\ -12 & -4 & 9\end{bmatrix},\ \mathscr{X}_0(:,:,3,2)=\begin{bmatrix}-7 & 11 & 4\\ -4 & -1 & 0\\ -14 & -5 & -21\\ 4 & 6 & 14\end{bmatrix},$$

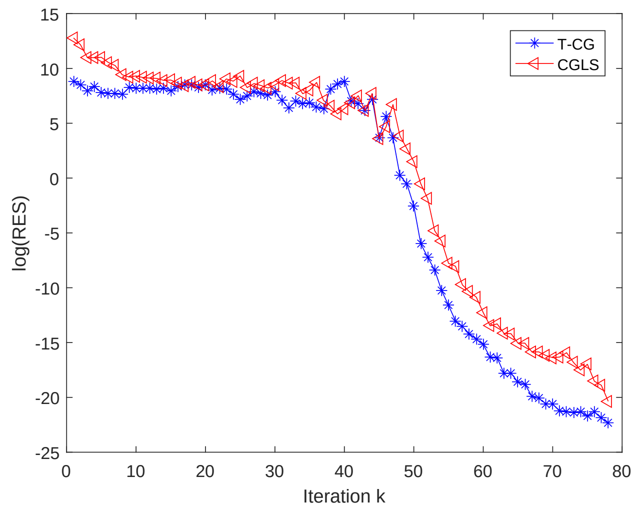

FIG. 5.2. *Convergence behavior of two algorithms for the tensor nearness problem in Example 5.2.*

$$\mathscr{X}_0(:,:,1,3)=\begin{bmatrix}1&1&4\\7&21&-5\\-4&2&0\\-16&5&-18\end{bmatrix},\mathscr{X}_0(:,:,2,3)=\begin{bmatrix}9&4&5\\2&-2&-4\\-7&-2&13\\-16&6&-4\end{bmatrix},$$

$$\mathscr{X}_0(:,:,3,3)=\begin{bmatrix}0&-9&1\\8&-16&-14\\9&-15&-12\\-19&-3&-2\end{bmatrix}.$$

Applying Algorithm 3.1 with $\mathscr{X}^{(1)}=\mathscr{O}$ to the tensor Equation (4.1), we obtain the solution to the tensor nearness problem (1.3) after 79 iteration steps, i.e., $\widehat{\mathscr{X}}$.

$$\widehat{\mathscr{X}}(:,:,1,1)=\begin{bmatrix}44.1912&54.1943&43.7075\\50.4602&68.6229&49.3393\\48.1731&53.1240&62.4182\\79.5249&53.7313&65.1154\end{bmatrix},\widehat{\mathscr{X}}(:,:,2,1)=\begin{bmatrix}39.4807&25.7036&37.5108\\39.1969&44.5352&36.9080\\41.6060&39.9043&56.2875\\45.9264&37.6248&44.2264\end{bmatrix},$$

$$\widehat{\mathscr{X}}(:,:,3,1)=\begin{bmatrix}40.5057&59.6836&56.9232\\41.8432&65.1252&55.9031\\83.6837&59.5245&59.2572\\52.0225&56.2288&71.7468\end{bmatrix},\widehat{\mathscr{X}}(:,:,1,2)=\begin{bmatrix}37.0000&41.0000&45.0000\\38.0000&42.0000&46.0000\\39.0000&43.0000&47.0000\\40.0000&44.0000&48.0000\end{bmatrix},$$

$$\widehat{\mathscr{X}}(:,:,2,2)=\begin{bmatrix}41.7494&54.6924&71.7399\\49.8603&58.7485&67.0232\\33.7992&61.6395&66.1633\\53.6123&55.2035&71.7550\end{bmatrix},\widehat{\mathscr{X}}(:,:,3,2)=\begin{bmatrix}34.5193&56.2964&52.4892\\36.8031&39.4648&55.0920\\36.3940&46.0957&37.7125\\34.0736&50.3752&51.7736\end{bmatrix},$$

$$\widehat{\mathscr{X}}(:,:,1,3)=\begin{bmatrix}73.0000&77.0000&81.0000\\74.0000&78.0000&82.0000\\75.0000&79.0000&83.0000\\76.0000&80.0000&84.0000\end{bmatrix},\widehat{\mathscr{X}}(:,:,2,3)=\begin{bmatrix}66.1178&55.6214&63.4721\\57.5053&53.0468&68.9621\\44.5545&56.3846&70.5580\\50.0412&66.5350&56.9444\end{bmatrix},$$

$$\widehat{\mathscr{X}}(:,:,3,3)=\begin{bmatrix}64.4359 & 52.8083 & 61.1574\\ 62.3311 & 36.4565 & 60.7722\\ 56.7895 & 56.3274 & 48.6033\\ 36.7992 & 60.3013 & 46.4383\end{bmatrix}.$$

At this time, $\|\widehat{\mathscr{X}}-\mathscr{X}_0\|=640.2422$.

Meanwhile, we compared our method with the conjugate gradient least squares method proposed in [30] for the same initial iteration tensor as above, and described their convergence curves in Figure 5.2, which indicate that our method has a slightly better performance.

6. Conclusions

In this paper, we first present an iterative method for solving the Sylvester tensor Equation (1.2), i.e., Algorithm 3.1. For any initial iteration tensor, it is shown that the solvability of this equation can be determined automatically (see, Theorem 3.2), and that the solution (if it exits) can be obtained within finite iteration steps in absence of roundoff errors (see, Theorem 3.1). Particularly, the least F-norm solution of (1.2) can also be derived by selecting appropriate initial iteration tensor (see, Theorem 3.3). Furthermore, applying this iterative method to another Sylvester tensor equation, i.e., (4.1), we can obtain the unique solution to the tensor nearness problem (1.3). Many other examples that we have tested in MATLAB confirm the theoretical results presented in this paper. Of course, for a problem with large and not sparse tensors $\mathscr{A},\mathscr{C}$ and $\mathscr{D}$, Algorithm 3.1 may not terminate in a finite number of iteration steps because of roundoff errors. This is an important problem which we should study in a future work. Moreover, the approach we propose in this paper can not be directly used to solve the Sylvester tensor Equation (1.2) when it is inconsistent, which will be considered in our future work as well.

Acknowledgements. The first author was supported by the National Natural Science Foundation of China (Grant No.11961057), and the Science Foundation of Tianshui Normal University (Nos.CXT2019-36,CXJ2020-11) as well as the Fuxi Scientific Research Innovation Team Foundation of Tianshui Normal University (No.FXD2020-03). The second author was supported by National Natural Science Foundation of China (Grant No.12071196).

REFERENCES

[1] T.G. Kolda and B.W. Bader. *Tensor decompositions and applications*, SIAM Rev., 51:455–500, 2009. 1

[2] A. Einstein, *The foundation of the general theory of relativity*, in A.J. Kox, M.J. Klein, and R. Schulmann (eds.), The Collected Papers of Albert Einstein, Princeton University Press, Princeton, 6:146–200, 1997. 1

[3] B.-W. Li, Y.-S. Sun, and D.-W. Zhang, *Chebyshev collocation spectral methods for coupled radiation and conduction in a concentric spherical participating medium*, ASME J. Heat Transfer., 131:062701–062709, 2009. 1, 1

[4] B.-W. Li, S. Tian, Y.-S. Sun, and Z. Mao, *Schur-decomposition for 3D matrix equations and its application in solving radiative discrete ordinates equations discretized by Chebyshev collocation spectral method*, J. Comput. Phys., 229:1198–1212, 2010. 1, 1

[5] W.-M. Lai, D. Rubin, and E. Krempl, *Introduction to Continuum Mechanics*, Butterworth-Heinemann Paperback, Oxford, 2009. 1, 1

[6] M. Itskov, *On the theory of fourth-order tensors and their applications in computational mechanics*, Comput. Meth. Appl. Mech. Eng., 189(2):419–438, 2000. 1

[7] M. Brazell, N. Li, C. Navasca, and C. Tamon, *Solving multilinear systems via tensor inversion*, SIAM J. Matrix Anal. Appl., 34(2):542–270, 2013. 1, 1, 1, 1, 2.1, 2.4

[8] Q.-W. Wang and X.-J. Xu, *Iterative algorithms for solving some tensor equations*, Linear Multilinear Algebra, 67(7):1325–1349, 2019. 1

[9] L.-Z. Sun, B.-D. Zheng, C.-J. Bu, and Y.-M. Wei, *Moore-Penrose inverse of tensors via Einstein product*, Linear Multilinear Algebra, 64(4):686–698, 2016. 1, 1, 2, 2.3, 2.2, 2

[10] Y.-X. Peng, X.-Y. Hu, and L. Zhang, *An iteration method for the symmetric solutions and the optimal approximation solution of the matrix equation $AXB=C$*, Appl. Math. Comput., 160:763–777, 2005. 1

[11] Z. Chen and L.-Z. Lu. *A projection method and Kronecker product preconditioner for solving Sylvester tensor equations*, Sci. China Math., 55:1281–1292, 2012. 1

[12] M.-L. Liang, C.-H. You, and L.-F. Dai, *An efficient algorithm for the generalized centro-symmetric solution of matrix equation $AXB=C$*, Numer. Algor., 44:173–184, 2007. 1

[13] N.J. Higham, *Matrix nearness problems and applications*, In M. J. C. Gover and S. Barnett (eds.), Applications of Matrix Theory, Oxford University Press, 1–27, 1989. 1

[14] A.-P. Liao, Z.-Z. Bai, and Y. Lei, *Best approximation of matrix equation $AXB+CYD=E$*, SIAM J. Matrix Anal. Appl., 27:675–688, 2005. 1

[15] I.S. Dhillon and J.A. Tropp, *Matrix nearness problems with Bregman divergences*, SIAM J. Matrix Anal. Appl., 29(2):1120–1146, 2007. 1

[16] S. Noschese and L. Reichel, *Some matrix nearness problems suggested by Tikhonov regularization*, Linear Algebra Appl., 502(1):366–386, 2016. 1

[17] T. Zhang and G.H. Golub, *Rank-one approximation to high order tensors*, SIAM J. Matrix Anal. Appl., 23:534–550, 2001. 1

[18] V.D. Silva and L.-H. Lim, *Tensor rank and the ill-posedness of the best low-rank approximation problem*, SIAM J. Matrix Anal. Appl., 30(3):1084–1127, 2008. 1

[19] L.-Q. Qi, *The best rank-one approximation ratio of a tensor space*, SIAM J. Matrix Anal. Appl., 32(2):430–442, 2011. 1

[20] J. Liu, P. Musialski, P. Wonka, and J. Ye, *Tensor completion for estimating missing values in visual data*, IEEE 12th International Conference on Computer Vision, 2114–2121, 2009. 1

[21] S. Gandy, B. Recht, and I. Yamada, *Tensor completion and low-n-rank tensor recovery via convex optimization*, Inverse Probl., 27(2):025010, 2011. 1

[22] J. Geng, L.-S. Wang, Y.-T. Xu, and X.-Y. Wang, *A weighted nuclear norm method for tensor completion*, Int. J. Signal Process. Image Process. Pattern Recognit., 7(1):1–12, 2014. 1

[23] M.-L. Liang and B. Zheng, *Further results on Moore-Penrose inverses of tensors with application to tensor nearness problems*, Comput. Math. Appl., 77:1282–1293, 2019. 1, 2

[24] S. Ragnarsson and C.F. Van Loan, *Block tensor unfoldings*, SIAM J. Matrix Anal. Appl., 33(1):149–169, 2012. 2.2

[25] R. Behera and D. Mishra, *Further results on generalized inverses of tensors via Einstein product*, Linear Multilinear Algebra, 65:1662–1682, 2017. 2.2

[26] G.H. Golub and C.F. Van Loan, *Matrix Computations*, Fourth Edition, Johns Hopkins University Press, 2012. 3

[27] T. Li, Q.-W. Wang, and X.-F. Duan, *Numerical algorithms for solving discrete Lyapunov tensor equation*, J. Comput. Appl. Math., 370:112676, 2020. 1

[28] X.-F. Zhang, Q.-W. Wang, and T. Li, *The accelerated overrelaxation splitting method for solving symmetric tensor equations*, Comput. Appl. Math., 39:155, 2020. 1

[29] T. Li, Q.-W. Wang, and X.-F. Zhang, *Hermitian and skew-Hermitian splitting methods for solving a tensor equation*, Int. J. Comput. Math., 1274–1290, 2020. 1

[30] B.-H. Huang and C.-F. Ma, *An iterative algorithm to solve the generalized Sylvester tensor equations*, Linear Multilinear Algebra, 68(6):1175–1200, 2020. 1, 5, 5