Journal of Applied Analysis and Computation
Volume 10, Number 3, June 2020, 972–985
Website:http://jaac-online.com/
DOI:10.11948/20190178

FINITE ITERATIVE ALGORITHM FOR THE COMPLEX GENERALIZED SYLVESTER TENSOR EQUATIONS*

Yifen Ke

Abstract A finite iterative algorithm is proposed to solve a class of complex generalized Sylvester tensor equations. The properties of this proposed algorithm are discussed based on a real inner product of two complex tensors and the finite convergence of this algorithm is obtained. Two numerical examples are offered to illustrate the effectiveness of the proposed algorithm.

Keywords Sylvester tensor equation, finite iterative algorithm, convergence.

MSC(2010) 65H10, 12A24.

1. Introduction

A tensor is a multidimensional array of data whose elements are referred by using multiple indices. We use

$$\mathcal{A} = (a_{i_1 i_2 \cdots i_N}) \in \mathbb{C}^{I_1 \times I_2 \times \cdots \times I_N}$$

to denote a complex tensor $\mathcal{A}$, where N is called the order of tensor $\mathcal{A}$ and $(I_1, I_2, \ldots, I_N)$ is the dimension of $\mathcal{A}$. As a special case, the vector is a 1-order tensor and the matrix is a 2-order tensor. Let

$$\overline{\mathcal{A}} = (\overline{a_{i_1 i_2 \cdots i_N}}) \in \mathbb{C}^{I_1 \times I_2 \times \cdots \times I_N}$$

be a conjugate of tensor $\mathcal{A}$, where $\overline{a_{i_1 i_2 \cdots i_N}}$ is the complex conjugate of $a_{i_1 i_2 \cdots i_N}$.

In this paper, we consider the iterative algorithm to solve the following complex generalized Sylvester tensor equation

$$\mathcal{X} \times_1 A_1 + \cdots + \mathcal{X} \times_N A_N + \overline{\mathcal{X}} \times_1 B_1 + \cdots + \overline{\mathcal{X}} \times_N B_N = \mathcal{D}, \tag{1.1}$$

where the matrices $A_n, B_n \in \mathbb{C}^{I_n \times I_n} (n = 1, 2, \ldots, N)$ and the right-side tensor $\mathcal{D} \in \mathbb{C}^{I_1 \times I_2 \times \cdots \times I_N}$ are known, and $\mathcal{X} \in \mathbb{C}^{I_1 \times I_2 \times \cdots \times I_N}$ is the unknown tensor to be solved. For the tensor $\mathcal{X} \in \mathbb{C}^{I_1 \times I_2 \times \cdots \times I_N}$ and the matrix $U \in \mathbb{C}^{I_n \times I_n}$, $\mathcal{X} \times_n U \in \mathbb{C}^{I_1 \times I_2 \times \cdots \times I_N}$ is defined by

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

Email address: keyifen@fjnu.edu.cn
College of Mathematics and Informatics & FJKLMAA, Fujian Normal University, Fuzhou 350117, China

*This work was supported by National Key Research and Development Program of China (Nos. 2018YFC1504200 and 2018YFC0603500) and National Natural Science Foundation of China (Nos. 11901098 and U1839207).

When the complex generalized tensor equation (1.1) over the real field $\mathbb{R}$ and $B_n = 0$ for all $n = 1, 2, \ldots, N$, the tensor equation (1.1) reduces to the real Sylvester tensor equation

$$\mathcal{X} \times_1 A_1 + \mathcal{X} \times_2 A_2 + \cdots + \mathcal{X} \times_N A_N = \mathcal{D}. \tag{1.2}$$

By using the Kronecker product [1,13], the tensor equation (1.2) can be reformulated as follows:

$$Ax = b \tag{1.3}$$

with

$$A = I^{(I_N)} \otimes \cdots \otimes I^{(I_2)} \otimes A_1 + \cdots + A_N \otimes I^{(I_{N-1})} \otimes \cdots \otimes I^{(I_1)}, \tag{1.4}$$

and $x = \mathrm{vec}(\mathcal{X})$, $b = \mathrm{vec}(\mathcal{D})$. Here, $I^{(n)}$ stands for the identity matrix of order n and the operator $\mathrm{vec}(\cdot)$ stacks the columns of a tensor to form a vector. In the case that $\mathcal{X}$ is a 3-order tensor, the tensor equation (1.2) reduces to

$$\mathcal{X} \times_1 A + \mathcal{X} \times_2 B + \mathcal{X} \times_3 C = \mathcal{D}, \tag{1.5}$$

which comes from the finite element [7] or spectral method [14,16–18] discretization of a linear partial differential equation in high dimension.

When $\mathcal{X}$ is a simple 2-order tensor, the complex generalized tensor equation (1.1) reduces to the extended Sylvester-conjugate matrix equation

$$A_1 X + X A_2^{\mathrm{T}} + B_1 \overline{X} + \overline{X} B_2^{\mathrm{T}} = D \tag{1.6}$$

and the Sylvester tensor equation (1.2) reduces to the Sylvester matrix equation

$$A_1 X + X A_2^{\mathrm{T}} = D, \tag{1.7}$$

which arises frequently from the areas of systems and control theory [6] and has received much attention, e.g., see [5,8–10].

Nowadays, tensor equations have attracted much attention [1–4,7,11,13,15]. For example, Kressner and Tobler in [13] proposed a so-called tensor Krylov subspace method for solving (1.3) with the coefficient matrix A as (1.4) and the right-hand side $b = b_1 \otimes b_2 \otimes \cdots \otimes b_N$, which is based on a tensorized Krylov subspace associated with A and b, i.e., the Kronecker product of the usual Krylov subspaces. The tensorized Krylov subspace method transforms the system (1.3) to a system of smaller size. In [1], Ballani and Grasedyck presented an iterative scheme similarly to Krylov subspace method to solve the linear system (1.3) with

$$A = \sum_{j=1}^{k_A} \otimes_{n=1}^{N} A_{jn}, \quad A_{jn} \in \mathbb{R}^{I_n \times I_n}$$

and

$$b = \sum_{j=1}^{k_b} \otimes_{n=1}^{N} b_{jn}, \quad b_{jn} \in \mathbb{R}^{I_n},$$

which is based on the projection of the residual to a low dimensional subspace and all calculations are performed in hierarchical Tucker format. In particular, Beik, Movahed and Ahmadi-Asl in [2] proposed the conjugate gradient algorithm of tensor form (CG-BTF) to solving the Sylvester tensor equation (1.2) with $A_n \in$

$\mathbb{R}^{I_n\times I_n}(n=1,2,\ldots,N)$ being symmetric positive definite and the nested conjugate gradient algorithm of tensor form (NCG-BTF) for the Sylvester tensor equation (1.2) with $A_n\in\mathbb{R}^{I_n\times I_n}(n=1,2,\ldots,N)$ being nonsymmetric positive definite.

As a general form of the matrix equation (1.6), that is

$$\sum_{i_1=1}^{p}A_{i_1}XB_{i_1}+\sum_{i_2=1}^{q}C_{i_2}\overline{X}D_{i_2}=F, \tag{1.8}$$

where $A_{i_1},C_{i_2}\in\mathbb{C}^{m\times r}$, $B_{i_2},D_{i_2}\in\mathbb{C}^{s\times n}(i_1=1,2,\ldots,p;i_2=1,2,\ldots,q)$, $F\in\mathbb{C}^{m\times n}$ are the given known matrices and $X\in\mathbb{C}^{r\times s}$ is the matrix to be determined, Wu, Lv and Hou [19] presented an iterative algorithm for solving (1.8). After that, Zhang in [21] offered the finite iterative algorithm for solving the complex generalized coupled Sylvester matrix equations, that is

$$\begin{cases}\sum_{j=1}^{q}(A_{1j}X_jB_{1j}+C_{1j}X_j^{\mathrm{T}}D_{1j}+E_{1j}\overline{X}_jF_{1j}+G_{1j}X_j^{\mathrm{H}}H_{1j})=M_1,\\ \sum_{j=1}^{q}(A_{2j}X_jB_{2j}+C_{2j}X_j^{\mathrm{T}}D_{2j}+E_{2j}\overline{X}_jF_{2j}+G_{2j}X_j^{\mathrm{H}}H_{2j})=M_2,\\ \quad\vdots\\ \sum_{j=1}^{q}(A_{pj}X_jB_{pj}+C_{pj}X_j^{\mathrm{T}}D_{pj}+E_{1j}\overline{X}_jF_{pj}+G_{pj}X_j^{\mathrm{H}}H_{pj})=M_p,\end{cases} \tag{1.9}$$

where $A_{ij},E_{ij}\in\mathbb{C}^{m_i\times r_j}$, $B_{ij},F_{ij}\in\mathbb{C}^{s_j\times n_i}$, $C_{ij},G_{ij}\in\mathbb{C}^{m_i\times s_j}$, $D_{ij},H_{ij}\in\mathbb{C}^{r_j\times n_i}$, $M_i\in\mathbb{C}^{m_i\times n_i}$ $(i=1,2,\ldots,p;j=1,2,\ldots,q)$ are the given known matrices and $X_j\in\mathbb{C}^{r_j\times s_j}(j=1,2,\ldots,q)$ are the unknown matrices. In this paper, the finite iterative algorithm for solving the complex generalized Sylvester tensor equation (1.1) will be discussed.

The remainder of this paper is organized as follows. In Section 2, we offers some symbols and preliminaries. In Section 3, we present a finite iterative algorithm for solving the complex generalized Sylvester tensor equation (1.1) and constructs several results which be used in the convergence proof. Numerical examples are offered in Section 4 to illustrate the effectiveness of the proposed algorithm. Finally, some concluding remarks are given in Section 5.

2. Preliminaries

For a matrix $A=(a_{ij})\in\mathbb{C}^{I\times I}$, the symbols A^{T} and A^{H} denote the transpose and conjugate transpose of matrix A, that is

$$A^{\mathrm{T}}=\begin{pmatrix}a_{11}&a_{21}&\cdots&a_{I1}\\ a_{12}&a_{22}&\cdots&a_{I2}\\ \vdots&\vdots&\ddots&\vdots\\ a_{1I}&a_{2I}&\cdots&a_{II}\end{pmatrix}\quad\text{and}\quad A^{\mathrm{H}}=\begin{pmatrix}\overline{a_{11}}&\overline{a_{21}}&\cdots&\overline{a_{I1}}\\ \overline{a_{12}}&\overline{a_{22}}&\cdots&\overline{a_{I2}}\\ \vdots&\vdots&\ddots&\vdots\\ \overline{a_{1I}}&\overline{a_{2I}}&\cdots&\overline{a_{II}}\end{pmatrix}.$$

Now, we recall the real inner product, which is given for complex spaces over the real field $\mathbb{R}$. This inner product will play a very important role in this paper.

Definition 2.1 ([20]). A real inner product space is a vector space $\mathbb{V}$ over the real field $\mathbb{R}$ together with an inner product, i.e., with a mapping

$$\langle \cdot, \cdot \rangle_r : \mathbb{V} \times \mathbb{V} \to \mathbb{R}$$

satisfying the following three axioms for all vectors $x, y, z \in \mathbb{V}$ and all scalars $\eta \in \mathbb{R}$.

(1) Symmetry: $\langle x, y \rangle_r = \langle y, x \rangle_r$.

(2) Linearity in the first argument:

$\langle \eta x, y \rangle_r = \eta \langle x, y \rangle_r$;

$\langle x + y, z \rangle_r = \langle x, z \rangle_r + \langle y, z \rangle_r$.

(3) Positive-definiteness: $\langle x, x \rangle_r > 0$ for all $x \neq 0$.

Moreover, two vectors $x, y \in \mathbb{V}$ are said to be orthogonal if $\langle y, x \rangle_r = 0$.

For the tensors $\mathcal{X}, \mathcal{Y} \in \mathbb{C}^{I_1 \times I_2 \times \cdots \times I_N}$, the inner product is defined as

$$\langle \mathcal{X}, \mathcal{Y} \rangle := \sum_{i_1, i_2, \cdots, i_N} x_{i_1 i_2 \cdots i_N} \overline{y_{i_1 i_2 \cdots i_N}}$$

and a real inner product can be defined as

$$\langle \mathcal{X}, \mathcal{Y} \rangle_r := \mathrm{Re}\big(\langle \mathcal{X}, \mathcal{Y} \rangle\big),$$

where $\mathrm{Re}(\lambda)$ denotes the real part of the complex number λ. In fact, for the tensors $\mathcal{X}, \mathcal{Y}, \mathcal{Z} \in \mathbb{C}^{I_1 \times I_2 \times \cdots \times I_N}$, it is easy to verify that

(1) $\langle \mathcal{X}, \mathcal{Y} \rangle_r = \langle \mathcal{Y}, \mathcal{X} \rangle_r$;

(2) $\langle \eta \mathcal{X}, \mathcal{Y} \rangle_r = \eta \langle \mathcal{X}, \mathcal{Y} \rangle_r$ for any $\eta \in \mathbb{R}$;

(3) $\langle \mathcal{X} + \mathcal{Y}, \mathcal{Z} \rangle_r = \langle \mathcal{X}, \mathcal{Z} \rangle_r + \langle \mathcal{Y}, \mathcal{Z} \rangle_r$;

(4) $\langle \mathcal{X}, \mathcal{X} \rangle_r > 0$ for all $\mathcal{X} \neq \mathcal{O}$, where $\mathcal{O}$ is the zero tensor in $\mathbb{C}^{I_1 \times I_2 \times \cdots \times I_N}$, i.e., all entries of $\mathcal{O}$ are 0.

The induced-norm of a tensor $\mathcal{X}$ is defined by the following formula

$$\|\mathcal{X}\| := \sqrt{\langle \mathcal{X}, \mathcal{X} \rangle} = \sqrt{\langle \mathcal{X}, \mathcal{X} \rangle_r}.$$

For the linear operator $\mathbf{L}$ from $\mathbb{C}^{I_1 \times I_2 \times \cdots \times I_N}$ to $\mathbb{C}^{I_1 \times I_2 \times \cdots \times I_N}$, the conjugate linear operator $\mathbf{L}^*$ from $\mathbb{C}^{I_1 \times I_2 \times \cdots \times I_N}$ to $\mathbb{C}^{I_1 \times I_2 \times \cdots \times I_N}$ satisfies

$$\langle \mathbf{L}(\mathcal{X}), \mathcal{Y} \rangle_r = \langle \mathcal{X}, \mathbf{L}^*(\mathcal{Y}) \rangle_r$$

for any $\mathcal{X}$ and $\mathcal{Y}$ in $\mathbb{C}^{I_1 \times I_2 \times \cdots \times I_N}$.

Lemma 2.1. *For the tensors $\mathcal{X}, \mathcal{Y} \in \mathbb{C}^{I_1 \times I_2 \times \cdots \times I_N}$ and $U \in \mathbb{C}^{I_n \times I_n}$, let*

$$\mathbf{L}_1(\mathcal{X}) = \mathcal{X} \times_n U \quad \textit{and} \quad \mathbf{L}_2(\mathcal{X}) = \overline{\mathcal{X}} \times_n U,$$

according to the conjugate operator, we have

$$\mathbf{L}_1^*(\mathcal{Y}) = \mathcal{Y} \times_n U^{\mathrm{H}} \quad \textit{and} \quad \mathbf{L}_2^*(\mathcal{Y}) = \overline{\mathcal{Y}} \times_n U^{\mathrm{T}}.$$

Proof. By the computation, we have

$$\begin{aligned}
\langle \mathbf{L}_1(\mathcal{X}), \mathcal{Y}\rangle_r &= \langle \mathcal{X} \times_n U, \mathcal{Y}\rangle_r \\
&= \mathrm{Re}\Big(\sum_{i_1,\cdots,i_{n-1},j,i_{n+1},\cdots,i_N} \big(\sum_{i_n=1}^{I_n} x_{i_1\cdots i_n\cdots i_N} u_{ji_n}\big) \overline{y_{i_1\cdots i_{n-1}ji_{n+1}\cdots i_N}} \Big) \\
&= \mathrm{Re}\Big(\sum_{i_1,\cdots,i_{n-1},i_n,i_{n+1},\cdots,i_N} x_{i_1\cdots i_n\cdots i_N} \big(\sum_{j=1}^{I_n} \overline{y_{i_1\cdots i_{n-1}ji_{n+1}\cdots i_N}} u_{ji_n}\big) \Big) \\
&= \mathrm{Re}\Big(\sum_{i_1,\cdots,i_{n-1},i_n,i_{n+1},\cdots,i_N} x_{i_1\cdots i_n\cdots i_N} \overline{\big(\sum_{j=1}^{I_n} y_{i_1\cdots i_{n-1}ji_{n+1}\cdots i_N} \overline{u_{ji_n}}\big)} \Big) \\
&= \langle \mathcal{X}, \mathcal{Y} \times_n U^{\mathrm{H}}\rangle_r = \langle \mathcal{X}, \mathbf{L}_1^*(\mathcal{Y})\rangle_r
\end{aligned}$$

and

$$\begin{aligned}
\langle \mathbf{L}_2(\mathcal{X}), \mathcal{Y}\rangle_r &= \langle \overline{\mathcal{X}} \times_n U, \mathcal{Y}\rangle_r \\
&= \mathrm{Re}\Big(\sum_{i_1,\cdots,i_{n-1},j,i_{n+1},\cdots,i_N} \big(\sum_{i_n=1}^{I_n} \overline{x_{i_1\cdots i_n\cdots i_N}} u_{ji_n}\big) \overline{y_{i_1\cdots i_{n-1}ji_{n+1}\cdots i_N}} \Big) \\
&= \overline{\mathrm{Re}\Big(\sum_{i_1,\cdots,i_{n-1},j,i_{n+1},\cdots,i_N} \big(\sum_{i_n=1}^{I_n} \overline{x_{i_1\cdots i_n\cdots i_N}} u_{ji_n}\big) \overline{y_{i_1\cdots i_{n-1}ji_{n+1}\cdots i_N}} \Big)} \\
&= \mathrm{Re}\Big(\sum_{i_1,\cdots,i_{n-1},j,i_{n+1},\cdots,i_N} \big(\sum_{i_n=1}^{I_n} x_{i_1\cdots i_n\cdots i_N} \overline{u_{ji_n}}\big) y_{i_1\cdots i_{n-1}ji_{n+1}\cdots i_N} \Big) \\
&= \mathrm{Re}\Big(\sum_{i_1,\cdots,i_{n-1},i_n,i_{n+1},\cdots,i_N} x_{i_1\cdots i_n\cdots i_N} \overline{\big(\sum_{j=1}^{I_n} \overline{y_{i_1\cdots i_{n-1}ji_{n+1}\cdots i_N}} u_{ji_n}\big)} \Big) \\
&= \langle \mathcal{X}, \overline{\mathcal{Y}} \times_n U^{\mathrm{T}}\rangle_r = \langle \mathcal{X}, \mathbf{L}_2^*(\mathcal{Y})\rangle_r.
\end{aligned}$$

This completes the proof. □

For simplicity, in the sequel, we use the following linear operator:

$$\mathbf{L}(\mathcal{X}) := \mathcal{X} \times_1 A_1 + \cdots + \mathcal{X} \times_N A_N + \overline{\mathcal{X}} \times_1 B_1 + \cdots + \overline{\mathcal{X}} \times_N B_N. \tag{2.1}$$

Then, the complex generalized Sylvester tensor equation (1.1) can be reformulated by

$$\mathbf{L}(\mathcal{X}) = \mathcal{D}. \tag{2.2}$$

According to Lemma 2.1, it is not difficult to verify that $\mathbf{L}^*$ is specified as follows:

$$\mathbf{L}^*(\mathcal{Z}) = \mathcal{Z} \times_1 A_1^{\mathrm{H}} + \cdots + \mathcal{Z} \times_N A_N^{\mathrm{H}} + \overline{\mathcal{Z}} \times_1 B_1^{\mathrm{T}} + \cdots + \overline{\mathcal{Z}} \times_N B_N^{\mathrm{T}}. \tag{2.3}$$

With these results, we will present a finite iterative algorithm to solve the complex generalized Sylvester tensor equation (1.1) in the following section.

3. Algorithm and convergence

In this section, we will propose a finite iterative algorithm for solving the complex generalized Sylvester tensor equation (1.1) and establish the convergence result for the proposed iterative algorithm.

Algorithm 3.1. (Finite iterative algorithm for solving (1.1))

Step 1. Let the matrices $A_n, B_n \in \mathbb{C}^{I_n \times I_n}$ for $n = 1, 2, \ldots, N$ and the right-hand side $\mathcal{D}$. Choose any initial tensor $\mathcal{X}_0 \in \mathbb{C}^{I_1 \times I_2 \times \cdots \times I_N}$, compute

$$\mathcal{R}_0 = \mathcal{D} - \mathbf{L}(\mathcal{X}_0),$$
$$\mathcal{Q}_0 = \mathbf{L}^*(\mathcal{R}_0).$$

Set $k := 0$.

Step 2. If $\|\mathcal{R}_k\| = 0$ or $\|\mathcal{Q}_k\| = 0$, then stop; else compute

$$\alpha_k = \frac{\|\mathcal{R}_k\|^2}{\|\mathcal{Q}_k\|^2},$$
$$\mathcal{X}_{k+1} = \mathcal{X}_k + \alpha_k \mathcal{Q}_k,$$
$$\mathcal{R}_{k+1} = \mathcal{D} - \mathbf{L}(\mathcal{X}_{k+1}) = \mathcal{R}_k - \alpha_k \mathbf{L}(\mathcal{Q}_k),$$
$$\mathcal{Q}_{k+1} = \mathbf{L}^*(\mathcal{R}_{k+1}) + \frac{\|\mathcal{R}_{k+1}\|^2}{\|\mathcal{R}_k\|^2}\mathcal{Q}_k.$$

Step 3. Set $k := k + 1$, go to Step 2.

Now, we give some results for Algorithm 3.1.

Theorem 3.1. *According to Algorithm 3.1, the tensor sequences $\mathcal{R}_k$ and $\mathcal{Q}_k$ for $k = 0, 1, 2, \ldots$ satisfy*

$$\langle \mathcal{R}_k, \mathcal{R}_l \rangle_r = 0, \quad k, l = 0, 1, 2, \ldots, k \neq l, \tag{3.1}$$

$$\langle \mathcal{Q}_k, \mathcal{Q}_l \rangle_r = 0, \quad k, l = 0, 1, 2, \ldots, k \neq l. \tag{3.2}$$

Proof. Using the mathematical induction. Firstly, we prove that

$$\langle \mathcal{R}_0, \mathcal{R}_1 \rangle_r = 0 \quad \text{and} \quad \langle \mathcal{Q}_0, \mathcal{Q}_1 \rangle_r = 0.$$

In fact, we have

$$\begin{aligned}
\langle \mathcal{R}_0, \mathcal{R}_1 \rangle_r &= \langle \mathcal{R}_0, \mathcal{R}_0 - \alpha_0 \mathbf{L}(\mathcal{Q}_0) \rangle_r \\
&= \langle \mathcal{R}_0, \mathcal{R}_0 \rangle_r - \langle \mathcal{R}_0, \alpha_0 \mathbf{L}(\mathcal{Q}_0) \rangle_r \\
&= \langle \mathcal{R}_0, \mathcal{R}_0 \rangle_r - \alpha_0 \langle \mathbf{L}(\mathcal{Q}_0), \mathcal{R}_0 \rangle_r \\
&= \langle \mathcal{R}_0, \mathcal{R}_0 \rangle_r - \alpha_0 \langle \mathcal{Q}_0, \mathbf{L}^*(\mathcal{R}_0) \rangle_r \\
&= \langle \mathcal{R}_0, \mathcal{R}_0 \rangle_r - \frac{\|\mathcal{R}_0\|^2}{\|\mathcal{Q}_0\|^2} \langle \mathcal{Q}_0, \mathcal{Q}_0 \rangle_r = 0
\end{aligned}$$

and

$$\begin{aligned}
\langle \mathcal{Q}_0, \mathcal{Q}_1 \rangle_r &= \langle \mathcal{Q}_0, \mathbf{L}^*(\mathcal{R}_1) + \frac{\|\mathcal{R}_1\|^2}{\|\mathcal{R}_0\|^2}\mathcal{Q}_0 \rangle_r \\
&= \langle \mathcal{Q}_0, \mathbf{L}^*(\mathcal{R}_1) \rangle_r + \frac{\|\mathcal{R}_1\|^2}{\|\mathcal{R}_0\|^2} \langle \mathcal{Q}_0, \mathcal{Q}_0 \rangle_r \\
&= \langle \mathbf{L}(\mathcal{Q}_0), \mathcal{R}_1 \rangle_r + \frac{\|\mathcal{R}_1\|^2}{\|\mathcal{R}_0\|^2} \langle \mathcal{Q}_0, \mathcal{Q}_0 \rangle_r \\
&= \langle \frac{\mathcal{R}_0 - \mathcal{R}_1}{\alpha_0}, \mathcal{R}_1 \rangle_r + \frac{\|\mathcal{R}_1\|^2}{\|\mathcal{R}_0\|^2} \langle \mathcal{Q}_0, \mathcal{Q}_0 \rangle_r \\
&= \frac{1}{\alpha_0} \langle \mathcal{R}_0, \mathcal{R}_1 \rangle_r - \frac{1}{\alpha_0} \langle \mathcal{R}_1, \mathcal{R}_1 \rangle_r + \frac{\|\mathcal{R}_1\|^2}{\|\mathcal{R}_0\|^2} \langle \mathcal{Q}_0, \mathcal{Q}_0 \rangle_r \\
&= -\frac{\|\mathcal{Q}_0\|^2}{\|\mathcal{R}_0\|^2} \langle \mathcal{R}_1, \mathcal{R}_1 \rangle_r + \frac{\|\mathcal{R}_1\|^2}{\|\mathcal{R}_0\|^2} \langle \mathcal{Q}_0, \mathcal{Q}_0 \rangle_r = 0.
\end{aligned}$$

Now, we assume that the results of Theorem 3.1 are correct for $k, l \leq s (s > 1)$. For $l = s + 1$ and $k = s$, we have

$$\begin{aligned}
\langle \mathcal{R}_s, \mathcal{R}_{s+1} \rangle_r &= \langle \mathcal{R}_s, \mathcal{R}_s - \alpha_s \mathbf{L}(\mathcal{Q}_s) \rangle_r \\
&= \langle \mathcal{R}_s, \mathcal{R}_s \rangle_r - \alpha_s \langle \mathbf{L}(\mathcal{Q}_s), \mathcal{R}_s \rangle_r \\
&= \langle \mathcal{R}_s, \mathcal{R}_s \rangle_r - \alpha_s \langle \mathcal{Q}_s, \mathbf{L}^*(\mathcal{R}_s) \rangle_r \\
&= \langle \mathcal{R}_s, \mathcal{R}_s \rangle_r - \alpha_s \langle \mathcal{Q}_s, \mathcal{Q}_s - \frac{\|\mathcal{R}_s\|^2}{\|\mathcal{R}_{s-1}\|^2} \mathcal{Q}_{s-1} \rangle_r \\
&= \langle \mathcal{R}_s, \mathcal{R}_s \rangle_r - \alpha_s \langle \mathcal{Q}_s, \mathcal{Q}_s \rangle_r + \frac{\alpha_s \|\mathcal{R}_s\|^2}{\|\mathcal{R}_{s-1}\|^2} \langle \mathcal{Q}_s, \mathcal{Q}_{s-1} \rangle_r \\
&= \langle \mathcal{R}_s, \mathcal{R}_s \rangle_r - \frac{\|\mathcal{R}_s\|^2}{\|\mathcal{Q}_s\|^2} \langle \mathcal{Q}_s, \mathcal{Q}_s \rangle_r = 0
\end{aligned}$$

and

$$\begin{aligned}
\langle \mathcal{Q}_s, \mathcal{Q}_{s+1} \rangle_r &= \langle \mathcal{Q}_s, \mathbf{L}^*(\mathcal{R}_{s+1}) + \frac{\|\mathcal{R}_{s+1}\|^2}{\|\mathcal{R}_s\|^2} \mathcal{Q}_s \rangle_r \\
&= \langle \mathcal{Q}_s, \mathbf{L}^*(\mathcal{R}_{s+1}) \rangle_r + \frac{\|\mathcal{R}_{s+1}\|^2}{\|\mathcal{R}_s\|^2} \langle \mathcal{Q}_s, \mathcal{Q}_s \rangle_r \\
&= \langle \mathbf{L}(\mathcal{Q}_s), \mathcal{R}_{s+1} \rangle_r + \frac{\|\mathcal{R}_{s+1}\|^2}{\|\mathcal{R}_s\|^2} \langle \mathcal{Q}_s, \mathcal{Q}_s \rangle_r \\
&= \langle \frac{\mathcal{R}_s - \mathcal{R}_{s+1}}{\alpha_s}, \mathcal{R}_{s+1} \rangle_r + \frac{\|\mathcal{R}_{s+1}\|^2}{\|\mathcal{R}_s\|^2} \langle \mathcal{Q}_s, \mathcal{Q}_s \rangle_r \\
&= \frac{1}{\alpha_s} \langle \mathcal{R}_s, \mathcal{R}_{s+1} \rangle_r - \frac{1}{\alpha_s} \langle \mathcal{R}_{s+1}, \mathcal{R}_{s+1} \rangle_r + \frac{\|\mathcal{R}_{s+1}\|^2}{\|\mathcal{R}_s\|^2} \langle \mathcal{Q}_s, \mathcal{Q}_s \rangle_r \\
&= -\frac{\|\mathcal{Q}_s\|^2}{\|\mathcal{R}_s\|^2} \langle \mathcal{R}_{s+1}, \mathcal{R}_{s+1} \rangle_r + \frac{\|\mathcal{R}_{s+1}\|^2}{\|\mathcal{R}_s\|^2} \langle \mathcal{Q}_s, \mathcal{Q}_s \rangle_r \\
&= 0.
\end{aligned}$$

Finally, we prove that the results of Theorem 3.1 are correct for $l = s + 1$ and $k \leq s - 1$. In fact, we have

$$\begin{aligned}
\langle \mathcal{R}_k, \mathcal{R}_{s+1} \rangle_r &= \langle \mathcal{R}_k, \mathcal{R}_s - \alpha_s \mathbf{L}(\mathcal{Q}_s) \rangle_r \\
&= \langle \mathcal{R}_k, \mathcal{R}_s \rangle_r - \alpha_s \langle \mathbf{L}(\mathcal{Q}_s), \mathcal{R}_k \rangle_r \\
&= -\alpha_s \langle \mathcal{Q}_s, \mathbf{L}^*(\mathcal{R}_k) \rangle_r \\
&= -\alpha_s \langle \mathcal{Q}_s, \mathcal{Q}_k - \frac{\|\mathcal{R}_k\|^2}{\|\mathcal{R}_{k-1}\|^2} \mathcal{Q}_{k-1} \rangle_r \\
&= -\alpha_s \langle \mathcal{Q}_s, \mathcal{Q}_k \rangle_r + \frac{\alpha_s \|\mathcal{R}_k\|^2}{\|\mathcal{R}_{k-1}\|^2} \langle \mathcal{Q}_s, \mathcal{Q}_{k-1} \rangle_r = 0
\end{aligned}$$

and

$$\begin{aligned}
\langle \mathcal{Q}_k, \mathcal{Q}_{s+1} \rangle_r &= \langle \mathcal{Q}_k, \mathbf{L}^*(\mathcal{R}_{s+1}) + \frac{\|\mathcal{R}_{s+1}\|^2}{\|\mathcal{R}_s\|^2} \mathcal{Q}_s \rangle_r \\
&= \langle \mathcal{Q}_k, \mathbf{L}^*(\mathcal{R}_{s+1}) \rangle_r + \frac{\|\mathcal{R}_{s+1}\|^2}{\|\mathcal{R}_s\|^2} \langle \mathcal{Q}_k, \mathcal{Q}_s \rangle_r
\end{aligned}$$

$$
\begin{aligned}
&= \langle \mathbf{L}(\mathcal{Q}_k), \mathcal{R}_{s+1} \rangle_r \\
&= \langle \frac{\mathcal{R}_k - \mathcal{R}_{k+1}}{\alpha_k}, \mathcal{R}_{s+1} \rangle_r \\
&= \frac{1}{\alpha_k} \big(\langle \mathcal{R}_k, \mathcal{R}_{s+1} \rangle_r - \langle \mathcal{R}_{k+1}, \mathcal{R}_{s+1} \rangle_r \big) \\
&= 0.
\end{aligned}
$$

According to the mathematical induction, the proof is completed. □

Theorem 3.2. *If the tensor $\mathcal{X}_*$ is a solution of the complex generalized Sylvester tensor equation* (1.1)*, for any initial iterative tensor $\mathcal{X}_0 \in \mathbb{C}^{I_1 \times I_2 \times \cdots \times I_N}$, the tensor sequences $\mathcal{R}_k$ and $\mathcal{Q}_k$ generated by Algorithm 3.1 satisfy*

$$
\langle \mathcal{Q}_k, \mathcal{X}_* - \mathcal{X}_k \rangle_r = \|\mathcal{R}_k\|^2, \quad k = 0, 1, 2, \ldots. \tag{3.3}
$$

Proof. Using the mathematical induction. Firstly, we verify that

$$
\langle \mathcal{Q}_0, \mathcal{X}_* - \mathcal{X}_0 \rangle_r = \|\mathcal{R}_0\|^2.
$$

In fact, we have

$$
\begin{aligned}
\langle \mathcal{Q}_0, \mathcal{X}_* - \mathcal{X}_0 \rangle_r &= \langle \mathbf{L}^*(\mathcal{R}_0), \mathcal{X}_* - \mathcal{X}_0 \rangle_r = \langle \mathcal{R}_0, \mathbf{L}(\mathcal{X}_* - \mathcal{X}_0) \rangle_r \\
&= \langle \mathcal{R}_0, \mathbf{L}(\mathcal{X}_*) - \mathbf{L}(\mathcal{X}_0) \rangle_r = \langle \mathcal{R}_0, \mathcal{D} - \mathbf{L}(\mathcal{X}_0) \rangle_r \\
&= \langle \mathcal{R}_0, \mathcal{R}_0 \rangle_r = \|\mathcal{R}_0\|^2.
\end{aligned}
$$

Now, we assume that the result of Theorem 3.2 is correct for $k = s (s \geq 1)$. For $k = s + 1$, we have

$$
\begin{aligned}
&\langle \mathcal{Q}_{s+1}, \mathcal{X}_* - \mathcal{X}_{s+1} \rangle_r \\
&= \langle \mathbf{L}^*(\mathcal{R}_{s+1}) + \frac{\|\mathcal{R}_{s+1}\|^2}{\|\mathcal{R}_s\|^2} \mathcal{Q}_s, \mathcal{X}_* - \mathcal{X}_{s+1} \rangle_r \\
&= \langle \mathbf{L}^*(\mathcal{R}_{s+1}), \mathcal{X}_* - \mathcal{X}_{s+1} \rangle_r + \frac{\|\mathcal{R}_{s+1}\|^2}{\|\mathcal{R}_s\|^2} \langle \mathcal{Q}_s, \mathcal{X}_* - \mathcal{X}_{s+1} \rangle_r \\
&= \langle \mathcal{R}_{s+1}, \mathbf{L}(\mathcal{X}_* - \mathcal{X}_{s+1}) \rangle_r + \frac{\|\mathcal{R}_{s+1}\|^2}{\|\mathcal{R}_s\|^2} \langle \mathcal{Q}_s, \mathcal{X}_* - \mathcal{X}_s - \alpha_s \mathcal{Q}_s \rangle_r \\
&= \langle \mathcal{R}_{s+1}, \mathcal{D} - \mathbf{L}(\mathcal{X}_{s+1}) \rangle_r + \frac{\|\mathcal{R}_{s+1}\|^2}{\|\mathcal{R}_s\|^2} \langle \mathcal{Q}_s, \mathcal{X}_* - \mathcal{X}_s \rangle_r - \frac{\alpha_s \|\mathcal{R}_{s+1}\|^2}{\|\mathcal{R}_s\|^2} \langle \mathcal{Q}_s, \mathcal{Q}_s \rangle_r \\
&= \langle \mathcal{R}_{s+1}, \mathcal{R}_{s+1} \rangle_r + \frac{\|\mathcal{R}_{s+1}\|^2}{\|\mathcal{R}_s\|^2} \|\mathcal{R}_s\|^2 - \frac{\|\mathcal{R}_s\|^2}{\|\mathcal{Q}_s\|^2} \cdot \frac{\|\mathcal{R}_{s+1}\|^2}{\|\mathcal{R}_s\|^2} \langle \mathcal{Q}_s, \mathcal{Q}_s \rangle_r \\
&= \|\mathcal{R}_{s+1}\|^2.
\end{aligned}
$$

According to the mathematical induction, the proof is completed. □

With Theorems 3.1 and 3.2, we can obtain the following conclusion.

Theorem 3.3. *If the complex generalized Sylvester tensor equation* (1.1) *is consistent, then a solution can be obtained within finite iteration steps by using Algorithm 3.1 for any initial tensor $\mathcal{X}_0 \in \mathbb{C}^{I_1 \times I_2 \times \cdots \times I_N}$.*

At the end of this section, we present the NCG-BTF algorithm [2] for the Sylvester tensor equation (1.2) with $A_n \in \mathbb{R}^{I_n \times I_n} (n = 1, 2, \ldots, N)$ being real non-symmetric positive definite. Let

$$H_n = \frac{A_n + A_n^{\mathrm{T}}}{2} \quad \text{and} \quad S_n = \frac{A_n^{\mathrm{T}} - A_n}{2} \tag{3.4}$$

for $n = 1, 2, \ldots, N$. Substituting $A_n = H_n - S_n$ into (1.2), we have

$$\mathcal{X} \times_1 H_1 + \mathcal{X} \times_2 H_2 + \cdots + \mathcal{X} \times_N H_N = \mathcal{D} + \mathcal{X} \times_1 S_1 + \mathcal{X} \times_2 S_2 + \cdots + \mathcal{X} \times_N S_N.$$

We compute $\mathcal{X}_{k+1}$ as the solution of the following tensor equation:

$$\mathcal{X} \times_1 H_1 + \mathcal{X} \times_2 H_2 + \cdots + \mathcal{X} \times_N H_N = \mathcal{D} + \mathcal{X}_k \times_1 S_1 + \mathcal{X}_k \times_2 S_2 + \cdots + \mathcal{X}_k \times_N S_N,$$

where $\mathcal{X}_k$ is the k-th approximate solution to the exact solution $\mathcal{X}_*$ of (1.2). Notice that for the tensors $\mathcal{X}, \mathcal{Y} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, the inner product is reduced to

$$\langle \mathcal{X}, \mathcal{Y} \rangle := \sum_{i_1, i_2, \cdots, i_N} x_{i_1 i_2 \cdots i_N} y_{i_1 i_2 \cdots i_N}.$$

Then, the NCG-BTF algorithm for solving (1.2) is as follows.

Algorithm 3.2. (NCG_BTF algorithm for solving (1.2) [2])

Step 1. Let the matrices $A_n, B_n \in \mathbb{R}^{I_n \times I_n}$ for $n = 1, 2, \ldots, N$ and the right-hand side $\mathcal{D} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$. Given the tolerance σ. Choose any initial tensor $\mathcal{X}_0 \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, compute

$$\mathcal{R}_0 = \mathcal{D} - \mathcal{X}_0 \times_1 A_1 - \mathcal{X}_0 \times_2 A_2 - \cdots - \mathcal{X}_0 \times_N A_N.$$

Set $\mathcal{X}_{(0,0)} = \mathcal{X}_0$ and $k := 0$.

Step 2. If $\|\mathcal{R}_k\| = 0$, then stop; else compute

$$\widetilde{\mathcal{D}} = \mathcal{D} + \mathcal{X}_{(k,0)} \times_1 S_1 + \mathcal{X}_{(k,0)} \times_2 S_2 + \cdots + \mathcal{X}_{(k,0)} \times_N S_N,$$
$$\mathcal{R}_{(k,0)} = \widetilde{\mathcal{D}} - \mathcal{X}_{(k,0)} \times_1 H_1 - \mathcal{X}_{(k,0)} \times_2 H_2 - \cdots - \mathcal{X}_{(k,0)} \times_N H_N,$$
$$\mathcal{P}_0 = \mathcal{R}_{(k,0)},$$

for $\ell = 0, 1, 2, \ldots,$ do

$$\mathcal{W} = \mathcal{P}_\ell \times_1 H_1 + \mathcal{P}_\ell \times_2 H_2 + \cdots + \mathcal{P}_\ell \times_N H_N,$$
$$\alpha_\ell = \frac{\langle \mathcal{R}_{(k,\ell)}, \mathcal{R}_{(k,\ell)} \rangle}{\langle \mathcal{W}, \mathcal{P}_\ell \rangle},$$
$$\mathcal{X}_{(k,\ell+1)} = \mathcal{X}_{(k,\ell)} + \alpha_\ell \mathcal{P}_\ell,$$
$$\mathcal{R}_{(k,\ell+1)} = \mathcal{R}_{(k,\ell)} - \alpha_\ell \mathcal{W},$$

if $\|\mathcal{R}_{(k,\ell+1)}\| \le \sigma \|\mathcal{R}_{(k,0)}\|$, then compute

$$\mathcal{X}_{k+1} = \mathcal{X}_{(k,\ell+1)},$$
$$\mathcal{R}_{k+1} = \mathcal{D} - \mathcal{X}_{k+1} \times_1 A_1 - \mathcal{X}_{k+1} \times_2 A_2 - \cdots - \mathcal{X}_{k+1} \times_N A_N,$$

else compute

$$\beta_\ell = \frac{\|\mathcal{R}_{(k,\ell+1)}\|^2}{\|\mathcal{R}_{(k,\ell)}\|^2},$$
$$\mathcal{P}_{\ell+1} = \mathcal{R}_{(k,\ell+1)} + \beta_\ell \mathcal{P}_\ell,$$

end if.

end for.

Step 3. Set $k := k + 1$ and $\mathcal{X}_{(k,0)} = \mathcal{X}_{k+1}$, go to Step 2.

4. Numerical examples

In this section, we use some test problems to examine the numerical effectiveness of Algorithm 3.1 for solving the complex generalized Sylvester tensor equation (1.1).

In actual computations, all runs are started from the initial tensor $\mathcal{X}_0 = \mathcal{O}$, are terminated if the current iterations satisfy

(i) $\|\mathcal{R}_k\| \leq 10^{-6}$ or $\|\mathcal{Q}_k\| \leq 10^{-6}$ for Algorithm 3.1,

(ii) $\|\mathcal{R}_k\| \leq 10^{-6}$ for Algorithm 3.2,

or if the number of the prescribed iterative steps $k_{\max} = 4000$ is exceeded, and are performed on a personal computer with 3.60 GHz central processing unit (Intel(R) Core(TM) i7-7700), 32.0 GB memory and Windows 10 operating system.

Here, 'IT' and 'CPU' denote the number of iteration steps and elapsed CPU time in seconds, respectively. In addition, ERR := $\|\mathcal{R}_k\|$. We comment here that the Tensor Toolbox [12] is utilized for solving the succeeding discussed problems

Example 4.1. Consider the following convection-diffusion equation

$$\begin{aligned} &-\nu\Delta u + c^{\mathrm{T}}\nabla u = f \text{ in } \ \Omega = [0,1]\times[0,1]\times[0,1],\\ &u = 0, \qquad\qquad\qquad \text{on } \ \partial\Omega. \end{aligned}$$

By using a standard finite difference discretization on a uniform grid for the diffusion term and a second-order convergent scheme (Fromm's scheme) for the convection term with the mesh-size $h = 1/(p+1)$, the discrete system matrix of the form (1.2) is obtained such that

$$A_n = \frac{\nu}{h^2}\begin{pmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \\ & & & -1 & 2 \end{pmatrix}_{p\times p} + \frac{c_n}{4h}\begin{pmatrix} 3 & -5 & 1 & & \\ 1 & 3 & -5 & \ddots & \\ & \ddots & \ddots & \ddots & 1 \\ & & 1 & 3 & -5 \\ & & & 1 & 3 \end{pmatrix}_{p\times p}, \quad n = 1,2,3.$$

And the right-hand side $\mathcal{D}$ is constructed so that $\mathcal{X}_* = ones(p,p,p)$ is the exact solution of the real Sylvester tensor equation (1.2).

For Example 4.1, we compare Algorithms 3.1 and 3.2. The inner iteration of Algorithm 3.2 is terminated if the value $\sigma = 10^{-3}$ or the number of the prescribed inner iterative steps $\ell_{\max} = 100$. The notation '—' denotes the case that the algorithm is stopped after $k_{\max}$ iterations without computing a suitable approximate solution.

The obtained results are presented in Table 1 with $c_1 = c_2 = c_3 = 1$ and Table 2 with $c_1 = 2$, $c_2 = 4$ and $c_3 = 8$, respectively. From both tables, we can see that the NCG_BTF algorithm is stopped after $k_{\max}$ iterations while the proposed finite iterative algorithm performs well for the viscous parameter $\nu = 0.01, 0.1$. Notice that when $\nu = 1$ and $c_1 = c_2 = c_3 = 1$, the NCG_BTF algorithm surpasses the finite iterative algorithm in CPU time. However, the finite iterative algorithm outperforms the NCG_BTF algorithm in terms of the required CPU time when $\nu = 1$, $c_1 = 2$, $c_2 = 4$ and $c_3 = 8$.

Table 1. Numerical results for Example 4.1 with $c_1 = c_2 = c_3 = 1$

	Algorithm	p	10	20	30	40
$\nu = 0.01$	Algorithm 3.1	IT	110	342	642	993
		CPU	0.1802	0.8596	2.2244	5.4495
		ERR	8.3334e-05	9.8194e-05	9.9630e-05	9.6617e-05
		$\|\mathcal{X}_k - \mathcal{X}_*\|$	3.5909e-06	2.8993e-06	2.2330e-06	1.7522e-06
	Algorithm 3.2	IT	—	—	—	—
		CPU	—	—	—	—
		ERR	—	—	—	—
		$\|\mathcal{X}_k - \mathcal{X}_*\|$	—	—	—	—
$\nu = 0.1$	Algorithm 3.1	IT	119	429	934	1621
		CPU	0.2086	1.0964	3.2363	8.5487
		ERR	6.8776e-05	9.1699e-05	9.2786e-05	9.9407e-05
		$\|\mathcal{X}_k - \mathcal{X}_*\|$	9.1169e-06	7.9246e-07	5.2281e-07	3.5560e-07
	Algorithm 3.2	IT	—	—	—	—
		CPU	—	—	—	—
		ERR	—	—	—	—
		$\|\mathcal{X}_k - \mathcal{X}_*\|$	—	—	—	—
$\nu = 1$	Algorithm 3.1	IT	118	458	1026	1823
		CPU	0.2060	1.1286	3.6703	10.0833
		ERR	8.1771e-05	8.7127e-05	9.2088e-05	9.8007e-05
		$\|\mathcal{X}_k - \mathcal{X}_*\|$	1.3843e-07	7.9723e-08	6.2234e-08	4.7348e-08
	Algorithm 3.2	IT	7	8	8	8
		CPU	0.1894	0.5707	1.1523	2.2605
		ERR	9.3391e-05	1.8217e-05	2.1200e-05	2.3934e-05
		$\|\mathcal{X}_k - \mathcal{X}_*\|$	1.5234e-06	2.5373e-07	2.9433e-07	3.3172e-07

Table 2. Numerical results for Example 4.1 with $c_1 = 2$, $c_2 = 4$, $c_3 = 8$

	Algorithm	p	10	20	30	40
$\nu = 0.01$	Algorithm 3.1	IT	167	412	670	963
		CPU	0.2821	1.0776	2.4698	5.8402
		ERR	9.4749e-05	9.7837e-05	9.5953e-05	9.9879e-05
		$\|\mathcal{X}_k - \mathcal{X}_*\|$	1.9364e-06	8.4346e-07	9.1760e-07	7.1202e-07
	Algorithm 3.2	IT	—	—	—	—
		CPU	—	—	—	—
		ERR	—	—	—	—
		$\|\mathcal{X}_k - \mathcal{X}_*\|$	—	—	—	—
$\nu = 0.1$	Algorithm 3.1	IT	185	547	1048	1694
		CPU	0.3216	1.4728	3.5380	9.4980
		ERR	8.7556e-05	9.3382e-05	9.7626e-05	9.7968e-05
		$\|\mathcal{X}_k - \mathcal{X}_*\|$	7.7475e-07	4.7910e-07	4.0709e-07	3.3213e-07
	Algorithm 3.2	IT	—	—	—	—
		CPU	—	—	—	—
		ERR	—	—	—	—
		$\|\mathcal{X}_k - \mathcal{X}_*\|$	—	—	—	—
$\nu = 1$	Algorithm 3.1	IT	211	767	1707	3018
		CPU	0.3646	1.9861	5.8372	16.2171
		ERR	9.6083e-05	9.7733e-05	9.9412e-05	9.9650e-05
		$\|\mathcal{X}_k - \mathcal{X}_*\|$	1.9164e-07	1.4820e-07	9.6294e-08	1.7522e-06
	Algorithm 3.2	IT	91	89	89	89
		CPU	2.4500	6.5841	13.4773	26.1544
		ERR	8.7412e-05	9.7808e-05	9.2644e-05	9.1362e-05
		$\|\mathcal{X}_k - \mathcal{X}_*\|$	1.0056e-06	1.1360e-06	1.0816e-06	1.7522e-06

Example 4.2. Consider the complex generalized Sylvester tensor equation (1.1) with

$$A_1 = \begin{pmatrix} 7-3\mathrm{i} & 3-18\mathrm{i} & -5-21\mathrm{i} \\ -17-\mathrm{i} & 4+4\mathrm{i} & 6+\mathrm{i} \\ -1 & -11+13\mathrm{i} & 3 \end{pmatrix}, \quad B_1 = \begin{pmatrix} 8-5\mathrm{i} & 4-7\mathrm{i} & 13+8\mathrm{i} \\ -6+12\mathrm{i} & -1+30\mathrm{i} & -4+9\mathrm{i} \\ -3-18\mathrm{i} & 11+6\mathrm{i} & 5-3\mathrm{i} \end{pmatrix},$$

$$A_2 = \begin{pmatrix} -4 & -1-4\mathrm{i} & -6+8\mathrm{i} \\ 3-6\mathrm{i} & -19+9\mathrm{i} & 10+2\mathrm{i} \\ 3-16\mathrm{i} & 10-18\mathrm{i} & 5\mathrm{i} \end{pmatrix}, \quad B_2 = \begin{pmatrix} 6+19\mathrm{i} & -2-7\mathrm{i} & 4+8\mathrm{i} \\ -7+26\mathrm{i} & 5 & 17+16\mathrm{i} \\ -9-17\mathrm{i} & 2+15\mathrm{i} & 12+5\mathrm{i} \end{pmatrix},$$

$$A_3 = \begin{pmatrix} -7-3\mathrm{i} & -5+2\mathrm{i} & -10-4\mathrm{i} \\ -4-4\mathrm{i} & -3+7\mathrm{i} & 11-6\mathrm{i} \\ -2-2\mathrm{i} & 0 & 4-5\mathrm{i} \end{pmatrix}, \quad B_3 = \begin{pmatrix} 5+11\mathrm{i} & -7-36\mathrm{i} & -14\mathrm{i} \\ 11+3\mathrm{i} & -15-6\mathrm{i} & 12-6\mathrm{i} \\ 12-6\mathrm{i} & 13-\mathrm{i} & 7-5\mathrm{i} \end{pmatrix},$$

$$A_4 = \begin{pmatrix} 4+7\mathrm{i} & 8+6\mathrm{i} & 6-\mathrm{i} \\ -1-9\mathrm{i} & -6-12\mathrm{i} & 1-30\mathrm{i} \\ 22+4\mathrm{i} & -7-\mathrm{i} & 7-6\mathrm{i} \end{pmatrix}, \quad B_4 = \begin{pmatrix} 9-4\mathrm{i} & -3-2\mathrm{i} & 2-2\mathrm{i} \\ -17+8\mathrm{i} & -5+9\mathrm{i} & 6-24\mathrm{i} \\ 5\mathrm{i} & -16+6\mathrm{i} & 11+16\mathrm{i} \end{pmatrix},$$

where i is imaginary unit. And the right-hand side $\mathcal{D}$ is constructed so that $\mathcal{X}_* = ones(3,3,3,3) + ones(3,3,3,3)\mathrm{i}$ is the exact solution of the tensor equation (1.1).

For Example 4.2, we get the approximate solution as $\mathcal{X}_*$ after 312 iterative steps, which the residual and error are

$$\mathrm{ERR} = 6.1095\text{e-}05 \quad \text{and} \quad \|\mathcal{X}_{312} - \mathcal{X}_*\| = 2.2969\text{e-}06.$$

For more details, the convergence history of the finite iterative algorithm is plotted in Figure 1.

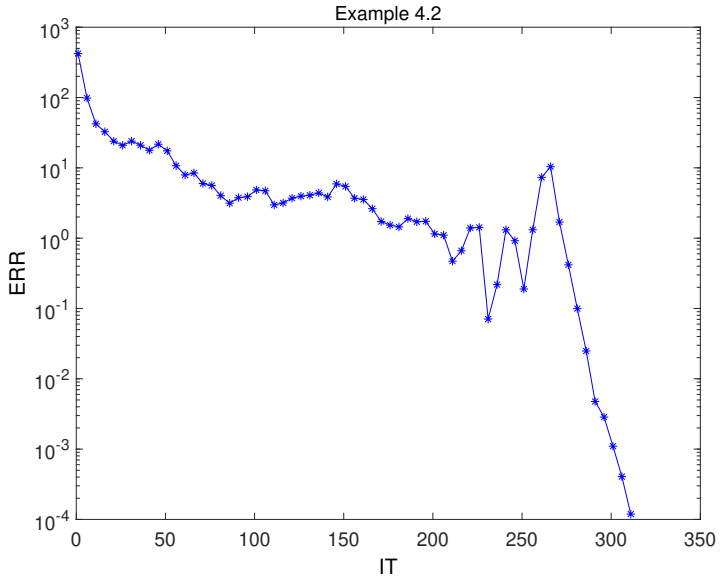

Figure 1. The convergence history of ERR for Example 4.2.

5. Conclusions

We have proposed a finite iterative algorithm to solve the complex generalized Sylvester tensor equation (1.1), which is generalized of Sylvester tensor equations (1.2) considered in [2]. It is proved that the algorithm is convergent within the finite iterative steps in the absence of the round-off error, which is based on the introducing real inner product. The applications of the proposed algorithm have been compared with the NCG-BTF algorithm when solves the Sylvester tensor equation (1.2). In addition, another example is offered to illustrate the effectiveness of the proposed algorithm for the complex generalized Sylvester tensor equation (1.1).

Acknowledgments. The author is grateful to the anonymous referees for their useful suggestions which improve the contents of this article. The author is also grateful for the hospitality and support during her research at Chern Institute of Mathematics of Nankai University from April 1 to June 30 in 2018.

References

[1] J. Ballani and L. Grasedyck, *A Projection method to solve linear systems in tensor format*, Numer. Linear. Algebra. Appl., 2013, 20, 27–43.

[2] F. P. A. Beik, F. S. Movahed and S. Ahmadi-Asl, *On the Krylov subspace methods based on tensor format for positive definite Sylvester tensor equations*, Numer. Linear. Algebra. Appl., 2016, 23, 444–466.

[3] Z. Chen and L. Lu, *A projection method and Kronecker product preconditioner for solving Sylvester tensor equations*, Sci. China Math., 2012, 55(6), 1281–1292.

[4] Z. Chen and L. Lu, *A gradient based iterative solutions for Sylvester tensor equations*, Math. Probl. Eng., Volume 2013, Article ID 819479, 7 pages.

[5] F. Ding and T. Chen, *On iterative solutions of general coupled matrix equations*, SIAM J. Control Optim., 2006, 44(6), 2269–2284.

[6] G. H. Golub, S. Nash and C. F. Van Loan, *A Hessenberg-Schur method for the problem* $AX + XB = C$, IEEE Trans. Automat. Contr., 1979, 24, 909–913.

[7] L. Grasedyck, *Existence and computation of low Kronecker-rank approximations for large linear systems of tensor product structure*, Computing, 2004, 72(3-4), 247–265.

[8] Y. Ke and C. Ma, *A preconditioned nested splitting conjugate gradient iterative method for the large sparse generalized Sylvester equation*, Comput. Math. Appl., 2014, 68, 1409–1420.

[9] Y. Ke and C. Ma, *Alternating direction method for generalized Sylvester matrix equation* $AXB + CYD = E$, Appl. Math. Comput., 2015, 260, 106–125.

[10] Y. Ke and C. Ma, *The unified frame of alternating direction method of multipliers for three classes of matrix equations arising in control theory*, Asian J. Control, 2017, 20(3), 1–18.

[11] T. G. Kolda and B. W. Bader, *Tensor decompositions and applications*, SIAM Rev., 2009, 51(3), 455–500.

[12] T. Kolda, B. Bader et al., *MATLAB Tensor Toolbox Version 2.6 (released Feb. 6, 2015)*. (Available from: http://www.sandia.gov/ tgkolda/TensorToolbox).

[13] D. Kressner and C. Tobler, *Krylov subspace methods for linear systems with tensor product structure*, SIAM J. Matrix Anal. Appl., 2010, 31, 1688–1714.

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

[15] L. Liang and B. Zheng, *Sensitivity analysis of the Lyapunov tensor equation*, Linear Multilinear A., 2018, 1–18.

[16] A. Malek and S. H. M. Masuleh, *Mixed collocation-finite difference method for 3D microscopic heat transport problems*, J. Comput. Appl. Math., 2008, 217, 137–147.

[17] A. Malek, Z. K. Bojdi and P. N. N. Golbarg, *Solving fully three-dimensional microscale dual phase lag problem using mixed-collocation finite difference discretization*, J. Heat Transfer, 2012, 134, 0945041–0945046.

[18] S. H. M. Masuleh and T. N. Phillips, *Viscoelastic flow in an undulating tube using spectral methods*, Comput. Fluids, 2004, 33, 1075–1095.

[19] A. Wu, L. Lv and M. Hou, *Finite iterative algorithms for extended Sylvester-conjugate matrix equations*, Math. Comput. Model., 2011, 54, 2363–2384.

[20] X. Zhang, *Matrix Analysis and Applications*, Tsinghua University Press, Beijing, 2004.

[21] H. Zhang, *A finite iterative algorithm for solving the complex generalized coupled Sylvester matrix equations by using the linear operators*, J. Franklin Inst., 2017, 354, 1856–1874.