Research Article

Zhengge Huang* and Jingjing Cui

Improved modified gradient-based iterative algorithm and its relaxed version for the complex conjugate and transpose Sylvester matrix equations

https://doi.org/10.1515/dema-2024-0083
received December 17, 2022; accepted October 18, 2024

Abstract: In this article, we present two new algorithms referred to as the improved modified gradient-based iterative (IMGI) algorithm and its relaxed version (IMRGI) for solving the complex conjugate and transpose (CCT) Sylvester matrix equations, which often arise from control theory, system theory, and so forth. Compared with the gradient-based iterative (GI) (A.-G. Wu, L.-L. Lv, and G.-R. Duan, *Iterative algorithms for solving a class of complex conjugate and transpose matrix equations*, Appl. Math. Comput. **217** (2011), 8343–8353) and the relaxed GI (RGI) (W.-L. Wang, C.-Q. Song, and S.-P. Ji, *Iterative solution to a class of complex matrix equations and its application in time-varying linear system*, J. Appl. Math. Comput. **67** (2021), 317–341) algorithms, the proposed ones can make full use of the latest information and need less computations, which leads to higher computational efficiency. With the real representation of a complex matrix as a tool, we establish sufficient and necessary conditions for the convergence of the IMGI and the IMRGI algorithms. Finally, some numerical examples are given to illustrate the effectiveness and advantages of the proposed algorithms.

Keywords: complex conjugate and transpose matrix equation, improved modified gradient-based iterative algorithm, improved modified relaxed gradient-based iterative algorithm, convergence condition, real representation

MSC 2020: 65F10, 65H10

1 Introduction

Sylvester matrix equations are often encountered in many scientific and engineering fields, such as signal processes, stability theory, observes design, control theory, matrix computation, pole assignment, optimal control, prediction and stability, system theory, eigenstructure assignment, and so on, we can refer to the related references [1–7] for details. This indicates that studying the computational methods of different kinds of Sylvester matrix equations has become an important subject in the field of computational mathematics and control. This motivates us to establish some effective algorithms for some types of Sylvester matrix equations in this work.

Due to the fact that the Sylvester matrix equations widely appear in practical problems and have important applications in many fields, in the past few decades, many researchers have devoted themselves to deriving a great deal of different methods to solve the Sylvester matrix equations, such as the Krylov subspace

*** Corresponding author: Zhengge Huang,** School of Mathematical Sciences, Center for Applied Mathematics of Guangxi, Guangxi Minzu University, 530006, Nanning, P. R. China, e-mail: ZhenggeHuang@163.com
Jingjing Cui: School of Mathematical Sciences, Center for Applied Mathematics of Guangxi, Guangxi Minzu University, 530006, Nanning, P. R. China, e-mail: jingjingcui1990@163.com

methods, neural network methods, direct methods, and so forth. The Krylov subspace methods are one of the classical iterative methods for the matrix equations, including the conjugate gradient method, the conjugate direction method, and so on. Besides, unlike numerical methods, artificial neural networks have the property of high-speed parallel operations and potential hardware implementations. Many effective neural network methods have been developed for matrix equation problems, for example, recurrent neural network method, zeroing neural network method. Also, by making use of matrix straightening operator and Kronecker product, Sylvester matrix equations can be transformed into linear systems, and the exact solutions of them can be calculated by applying the direct methods. However, it may lead to high-scale problems, and will consume much more computer time and memory space with the dimensions increasing. Hence, the research of iterative methods for the matrix equations have attracted considerable attention from many researchers.

A large number of efficient iterative methods have been developed for solving different kinds of Sylvester matrix equations. For instance, Ding and Chen [8] proposed the gradient-based iterative (GI) algorithm for solving the Sylvester matrix equation $AX + XB = C$. Ding et al. [9] derived the iterative method for the generalized Sylvester matrix equation $AXB + CXD = E$. These methods are constructed by applying the hierarchical identification principle, which decomposes a system into some subsystems and then identifies the unknown parameters of each subsystem successively. Based on this idea, there are a great deal of work involved in this direction. For example, Xie et al. [10] constructed the gradient-based and least squares-based iterative algorithms for the Sylvester-transpose matrix equation $AXB + CX^TD = F$. Song et al. [11] developed the GI algorithm for the coupled Sylvester-transpose matrix equation $\sum_{\eta=1}^{p}(A_{i\eta}X_\eta B_{i\eta} + C_{i\eta}X_\eta^T D_{i\eta}) = F_i$ $(i = 1, 2, \ldots, N)$. By using the hierarchical identification principle, Wu et al. [12] designed an efficient algorithm for solving the extended Sylvester-conjugate matrix equation $AXB + C\bar{X}D = F$. Besides, Wu et al. [13] extended the GI algorithm to solve the coupled Sylvester-conjugate matrix equation $\sum_{\eta=1}^{p}(A_{i\eta}X_\eta B_{i\eta} + C_{i\eta}\bar{X}_\eta D_{i\eta}) = F_i$ $(i = 1, 2, \ldots, N)$ and derived the sufficient condition for the convergence of the GI algorithm. Subsequently, Wu et al. [14] put forward the GI algorithm for solving the complex conjugate and transpose (CCT) matrix equation $\sum_{l=1}^{s_1}A_lXB_l + \sum_{l=1}^{s_2}C_l\bar{X}D_l + \sum_{l=1}^{s_3}G_lX^TH_l + \sum_{l=1}^{s_4}M_lX^HN_l = F$. Afterward, Beik et al. [1] proposed the GI algorithm for the generalized coupled Sylvester-transpose and conjugate matrix equations over a group of reflexive (anti-reflexive) matrices. For more iteration methods based on the hierarchical identification principle, we can refer to [6,7,15–24] and the references therein. Moreover, the finite iterative methods are a kind of effective algorithms for the Sylvester matrix equations, which can calculate the solutions of many kinds of matrix equations within finite steps in the absence of round-off errors. Then many finite iterative methods have been established. For instance, Li [25] developed a finite iterative method for solving the coupled Sylvester matrix equations with conjugate transpose. By introducing the real linear operator, Zhang [26] proposed a finite iterative algorithm for solving the complex generalized coupled Sylvester matrix equations. In addition, Yan and Ma [27] designed the biconjugate residual algorithm for solving the reflexive or antireflexive solutions of generalized coupled Sylvester matrix equations, and they also proposed a finite iterative algorithm to solve a class of generalized coupled Sylvester-conjugate matrix equations over the generalized Hamiltonian matrices in [28]. Quite recently, Ma and Yan [29] derived a modified conjugate gradient method to solve the general discrete-time periodic Sylvester matrix equations.

Note that the CCT Sylvester matrix equations are quite general and include several kinds of Sylvester matrix equations, such as the Sylvester matrix equations, the complex conjugate Sylvester matrix equations, the Sylvester transpose matrix equations, and the conjugate transpose Sylvester matrix equations. This shows that the CCT Sylvester matrix equations appear in many practical problems and applications, and the solving problem of the CCT Sylvester matrix equations is worth research. For example, according to [21], the continuous zeroing dynamics design of time-varying linear system can be defined as $\dot{L}(t) = -\lambda JMP \circ (L(t))$, and $JMP(\cdot) : \mathbb{C}^{n\times n} \to \mathbb{C}^{n\times n}$ is defined as an array of jmp functions:

$$JMP(L(t)) = \begin{pmatrix} jmp(l_{11}(t)) & jmp(l_{12}(t)) & \cdots & jmp(l_{1n}(t)) \\ jmp(l_{21}(t)) & jmp(l_{22}(t)) & \cdots & jmp(l_{2n}(t)) \\ \vdots & \vdots & \ddots & \vdots \\ jmp(l_{n1}(t)) & jmp(l_{n2}(t)) & \cdots & jmp(l_{nn}(t)) \end{pmatrix},$$

where $jmp(l_{ij}(t)) = 1$ if $l_{ij}(t) > 0$ and $jmp(l_{ij}(t)) = 0$ if $l_{ij}(t) \le 0$, and the time derivative is

$$\begin{aligned} L(t) = {} & \dot{A}_1(t)Y(t)B_1(t) + A_1(t)\dot{Y}(t)B_1(t) + A_1(t)Y(t)\dot{B}_1(t) + \dot{A}_2(t)\bar{Y}(t)B_2(t) \\ & + A_2(t)\dot{\bar{Y}}(t)B_2(t) + A_2(t)\bar{Y}(t)\dot{B}_2(t) + \dot{A}_3(t)Y^T(t)B_3(t) + A_3(t)\dot{Y}^T(t)B_3(t) \\ & + A_3(t)Y^T(t)\dot{B}_3(t) + \dot{A}_4(t)Y^H(t)B_4(t) + A_4(t)\dot{Y}^H(t)B_4(t) + A_4(t)Y^H(t)\dot{B}_4(t). \end{aligned}$$

$\dot{L}(t) = -\lambda JMP \circ (L(t))$ is substituted into the aforementioned equation, and it can be simplified as the following time-varying linear system:

$$A_1(t)\dot{Y}(t)B_1(t) + A_2(t)\dot{\bar{Y}}(t)B_2(t) + A_3(t)\dot{Y}^T(t)B_3(t) + A_4(t)\dot{Y}^H(t)B_4(t) = G(t), \tag{1}$$

with $A_i(t), B_i(t), G_i(t) \in \mathbb{C}^{n\times n}$ being the smoothly time-varying matrices, and $Y(t) \in \mathbb{C}^{n\times n}$ being the unknown time-varying matrix needs to be determined. In addition, $t \in [0, t_f] \subseteq [0, +\infty)$, and t_f stands for the final time. Then equation (1) can be rewritten as the CCT Sylvester matrix equation. Based on this fact, in this work, we investigate the effective and feasible iterative algorithms to solve the CCT Sylvester matrix equations, whose form is as follows:

$$A_1ZB_1 + A_2\bar{Z}B_2 + A_3Z^TB_3 + A_4Z^HB_4 = H, \tag{2}$$

where $A_i, B_i, H \in \mathbb{C}^{n\times n}$ $(i = 1, 2, 3, 4)$ are given matrices and the unknown matrix $Z \in \mathbb{C}^{n\times n}$ needs to be computed.

Until now, some different algorithms for the CCT Sylvester matrix equations have been proposed. Apart from the ones in [1,14,26] mentioned earlier, Zhang and Yin [30] offered a new proof of the GI algorithm for the CCT Sylvester matrix equations by making use of the properties of the real representation of a complex matrix, and deduced the necessary and sufficient conditions for the convergence, optimal parameter, and corresponding optimal convergence factor. Then the optimal GI (OGI) algorithm is obtained. Furthermore, on the basis of the relaxation technique used in [15,17], Wang et al. [21] introduced a relaxed factor into the GI algorithm and developed the relaxed GI (RGI) algorithm, and necessary and sufficient conditions for guaranteeing the convergence of the RGI algorithm are provided. Numerical results in [21] revealed that the RGI algorithm performs better than the GI and the OGI ones.

To improve the efficiency of the GI algorithm for the Sylvester matrix equations, many improved and modified versions of the GI algorithm have been established. For instance, Wang et al. [20] proposed the modified GI (MGI) algorithm for solving the Sylvester matrix equations, which uses the latest information to compute the next result. This idea has been extended to the generalized Sylvester-transpose matrix equations and the extended Sylvester-conjugate matrix equations in [19] and [18], respectively. In addition, to reduce the computations and the storage spaces of the GI algorithm, Tian et al. [23] and Hu and Wang [31] derived the Jacobi GI (JGI) algorithm for solving the Sylvester matrix equations by replacing the coefficient matrices by their diagonal parts.

It can be seen from the iterative scheme of the GI algorithm in [14,30] that it does not use the latest calculation result to compute the next result. Then the convergence speed of the GI algorithm is slow in many cases, and even the optimal parameter is adopted in GI algorithm. In addition, when the coefficient matrices of the CCT Sylvester matrix equation are large and dense, the matrix multiplication in the GI algorithm may require a large amount of computation and storage spaces, which consumes much time and reduces the computational efficiency of the GI algorithm. To overcome the aforementioned shortcomings and improve the convergence rate of the GI algorithm, we first use the hierarchical identification principle to turn CCT Sylvester matrix equation into four subsystems. Enlightened by the ideas of the JGI and the MGI algorithms, we apply the updated technique to the GI algorithm, replace the related coefficient matrices by their diagonal parts, and then construct the improved modified GI (IMGI) algorithm for the CCT Sylvester matrix equation (2). Moreover, we apply the relaxation technique to the IMGI algorithm and develop the improved modified relaxed GI (IMRGI) algorithm. We investigate some convergence properties of the IMGI and the IMRGI algorithms, which indicate that the proposed algorithms are convergent under proper restrictions. Numerical experiments are provided to demonstrate the effectiveness and superiorities of the IMGI and the IMRGI algorithms.

The rest of this article is organized as follows. In Section 2, some notations, definitions, and results, which will be used in the latter parts of this article are listed. In Section 3, we put forward two new algorithms referred to as the IMGI and the IMRGI algorithms for the CCT Sylvester matrix equation (2), and establish the convergence theorems of the two proposed algorithms. Several numerical experiments are provided to validate the effectiveness and advantages of the proposed algorithms in Section 4. Finally, some conclusions are drawn in Section 5 to end this article.

2 Preliminaries

In this section, we review and recall some notations, definitions, and lemmas, which come from References [11,21,30,32,33] and will be used throughout this article.

For a complex number p, $\mathrm{Re}(p)$ and $\mathrm{Im}(p)$ denote the real and complex parts of p, respectively. Let A be a complex matrix. $\lambda(A)$, A^T, $\bar{A}$, and A^H represent the spectrum, transpose, conjugate, and conjugate transpose of A, respectively. And $|A| = (|a_{ij}|)$ denotes the absolute value of the matrix A. If A is a square matrix, then $\mathrm{tr}(A)$ stands for the trace of A. The spectral radius, Euclid norm, and Frobenius norm of A are denoted by $\rho(A)$, $\|A\|_2 = \sqrt{\lambda_{\max}(A^H A)}$, and $\|A\| = \sqrt{\mathrm{tr}(A^H A)}$, respectively. Let $A = D - C$, where D and C are the diagonal and non-diagonal parts of A, respectively.

In addition, we present several useful definitions below.

Definition 2.1. [32] Let $A = [a_1, a_2, \ldots, a_n] \in \mathbb{C}^{m\times n}$ with a_i being the ith column of A. The vector stretching function of A is defined as follows:

$$\mathrm{vec}(A) = [a_1^T, a_2^T, \ldots, a_n^T]^T \in \mathbb{C}^{mn}.$$

Definition 2.2. [21] The vec-permutation matrix $P(m, n)$ is a square $mn \times mn$ matrix and can be defined as follows:

$$P(m,n) = \sum_{i=1}^{m}\sum_{j=1}^{n} E_{ij} \otimes E_{ij}^T,$$

where $E_{ij} = e_i e_j^T$ is an elementary matrix of order $m \times n$.

Definition 2.3. [32] For two matrices $D = (d_{ij}) \in \mathbb{C}^{m\times k}$ and $F = (f_{ij}) \in \mathbb{C}^{n\times l}$, the Kronecker product of D and F is defined as follows:

$$D \otimes F = \begin{pmatrix} d_{11}F & d_{12}F & \cdots & d_{1k}F \\ d_{21}F & d_{22}F & \cdots & d_{2k}F \\ \vdots & \vdots & & \vdots \\ d_{m1}F & d_{m2}F & \cdots & d_{mk}F \end{pmatrix} = [d_{ij}F]_{m\times k} \in C^{mn\times kl}.$$

Definition 2.4. [32] Let $L = L_1 + iL_2 \in \mathbb{C}^{p\times q}$ with L_1 and L_2 being the real and imaginary parts of L, respectively. Then the real representation of L is defined as follows:

$$L_\sigma = \begin{pmatrix} L_1 & L_2 \\ L_2 & -L_1 \end{pmatrix} \in \mathbb{R}^{2p\times 2q}.$$

In the following, we list several useful lemmas.

Lemma 2.1. [13] (The properties of the real representation)
(1) *For $G, H \in \mathbb{C}^{m\times n}$, $t \in \mathbb{R}$, it has*

$$(G + H)_\sigma = G_\sigma + H_\sigma, \quad (tG)_\sigma = tG_\sigma, \quad P_m G_\sigma P_n = (\bar{G})_\sigma;$$

(2) *If* $S \in \mathbb{C}^{m\times n}$, $T \in \mathbb{C}^{n\times r}$, *and* $U \in \mathbb{C}^{r\times p}$, *then*

$$(ST)_\sigma = S_\sigma P_n T_\sigma = S_\sigma(\bar{T})_\sigma P_r, \quad (STU)_\sigma = S_\sigma(\bar{T})_\sigma U_\sigma;$$

(3) *If* $F \in \mathbb{C}^{m\times n}$, *then* $Q_m F_\sigma Q_n = F_\sigma$;

(4) *For* $K \in \mathbb{C}^{m\times n}$, *it has* $((K^T)_\sigma)^T = K_\sigma$, *where* P_i *and* Q_j *are defined as follows*:

$$P_i = \begin{pmatrix} I_i & 0 \\ 0 & -I_i \end{pmatrix}, \quad Q_j = \begin{pmatrix} 0 & I_j \\ -I_j & 0 \end{pmatrix},$$

with I_i *and* I_j *being the* $i \times i$ *and* $j \times j$ *unit matrices, respectively.*

Lemma 2.2. [21] *Let* $Y \in \mathbb{C}^{m\times n}$, *then*

$$\mathrm{vec}(Y^T) = P(m,n)\mathrm{vec}(Y).$$

Lemma 2.3. [33] *Let* $A \in \mathbb{C}^{m\times n}$, $B \in \mathbb{C}^{n\times s}$, *and* $C \in \mathbb{C}^{s\times t}$, *then*

$$\mathrm{vec}(ABC) = (C^T \otimes A)\mathrm{vec}(B).$$

Lemma 2.4. [5] *Let* $C = [C_{ij}]$ *be a square block matrix and* $D = [||C_{ij}||]$. *Then* $\rho(C) \le \rho(|C|) \le \rho(D)$.

Lemma 2.5. [34] *Let* T *be a normal matrix, then* $\rho(T) = ||T||_2$.

3 The IMGI and the IMRGI algorithms

In this section, we will establish two new algorithms for solving the CCT Sylvester matrix equations. The proposed methods outperform the GI one in [14] and its relaxed version (RGI) [21] from the point of view of computational efficiency, which will be confirmed by the numerical results in Section 5. To this end, we first introduce a lemma as follows.

Lemma 3.1. [9] *Consider the matrix equation* $AXB = F$, *where* $A \in \mathbb{R}^{m\times r}$, $B \in \mathbb{R}^{s\times n}$, *and* $F \in \mathbb{R}^{m\times n}$ *are known matrices, and* $X \in \mathbb{R}^{r\times s}$ *is the matrix to be determined. For this matrix equation, an iterative algorithm is constructed as follows*:

$$X(k+1) = X(k) + \mu A^T(F - AX(k)B)B^T, \tag{3}$$

with

$$0 < \mu < \frac{2}{||A||_2^2||B||_2^2}. \tag{4}$$

If this matrix equation has a unique solution X^*, *then the iterative solution* $X(k)$ *converges to the unique solution* X^*, *that is*, $\lim_{k\to\infty} X(k) = X^*$.

Let $\Theta(Z) = A_1ZB_1 + A_2\bar{Z}B_2 + A_3Z^TB_3 + A_4Z^HB_4$. We will review the GI algorithm in [14], whose framework is as follows.

The GI algorithm [14]:

Step 1: Given matrices $A_i, B_i, H \in \mathbb{C}^{n\times n}$ $(i = 1, 2, 3, 4)$, and two constants $\varepsilon > 0$ and $\mu > 0$. Choose the initial matrix $Z(0)$, and set $k = 0$;

Step 2: If $\delta_k = \frac{||H - A_1Z(k)B_1 - A_2\bar{Z}(k)B_2 - A_3Z(k)^TB_3 - A_4Z(k)^HB_4||}{||H||} < \varepsilon$, stop; otherwise, go to Step 3;

Step 3: Update the sequences

$$\begin{aligned}
Z_1(k+1) &= Z(k) + \mu A_1^H(H - \Theta(Z(k)))B_1^H,\\
Z_2(k+1) &= Z(k) + \mu A_2^T(\bar{H} - \overline{\Theta(Z(k))})B_2^T,\\
Z_3(k+1) &= Z(k) + \mu \overline{B_3}(H^T - \Theta^T(Z(k)))\overline{A_3},\\
Z_4(k+1) &= Z(k) + \mu B_4(E^H - \Theta^H(Z(k)))A_4,\\
Z(k+1) &= \frac{Z_1(k+1) + Z_2(k+1) + Z_3(k+1) + Z_4(k+1)}{4};
\end{aligned}$$

Step 4: Set $k := k + 1$ and return to Step 2.

Numerical experiments in [14] showed the effectiveness of the GI algorithm for the CCT Sylvester matrix equations.

According to the framework of the aforementioned GI algorithm, it can be seen that Step 3 of this algorithm may consume much time because the matrices A_i and B_i $(i = 1, 2, 3, 4)$ may be large and dense. To reduce the computation of the GI algorithm, according to Lemma 3.1, and inspired by the ideas of [23,24,31], we replace the matrices A_i and B_i $(i = 1, 2, 3, 4)$ by their diagonal parts. This may reduce the computing time of each iteration of the GI method, and the total calculation time decreases.

According to the hierarchical identification principle, we first transform the CCT Sylvester matrix equation (2) into the following four equations:

$$A_1ZB_1 = H_1, \quad \bar{A}_2Z\bar{B}_2 = H_2, \quad B_3^TZA_3^T = H_3, \quad B_4^HZA_4^H = H_4, \tag{5}$$

where $H_1 = H - \Theta(Z) + A_1ZB_1$, $H_2 = \overline{H - \Theta(Z) + A_2\bar{Z}B_2}$, $H_3 = (H - \Theta(Z) + A_3Z^TB_3)^T$, and $H_4 = (H - \Theta(Z) + A_4Z^HB_4)^H$. And then we split the matrices A_i, B_i $(i = 1, 2, 3, 4)$ into $A_i = D_{i1} + C_{i1}$ and $B_i = D_{i2} + C_{i2}$ $(i = 1, 2, 3, 4)$, with D_{i1} and D_{i2} being the diagonal parts of A_i and B_i, respectively. Then it follows from (5) that

$$\begin{aligned}
A_1ZB_1 = H_1 &\Rightarrow (D_{11} + C_{11})Z(D_{12} + C_{12}) = H_1,\\
\bar{A}_2Z\bar{B}_2 = H_2 &\Rightarrow \overline{(D_{21} + C_{21})Z(D_{22} + C_{22})} = H_2,\\
B_3^TZA_3^T = H_3 &\Rightarrow (D_{32} + C_{32})^TZ(D_{31} + C_{31})^T = H_3,\\
B_4^HZA_4^H = H_4 &\Rightarrow (D_{42} + C_{42})^HZ(D_{41} + C_{41})^H = H_4,
\end{aligned} \tag{6}$$

which yields that

$$\begin{aligned}
D_{11}ZD_{12} &= H_1 - D_{11}ZC_{12} - C_{11}ZD_{12} - C_{11}ZC_{12} := \tilde{H}_1,\\
\bar{D}_{21}Z\bar{D}_{22} &= H_2 - \bar{D}_{21}Z\bar{C}_{22} - \bar{C}_{21}Z\bar{D}_{22} - \bar{C}_{21}Z\bar{C}_{22} := \tilde{H}_2,\\
D_{32}^TZD_{31}^T &= H_3 - D_{32}^TZC_{31}^T - C_{32}^TZD_{31}^T - C_{32}^TZC_{31}^T := \tilde{H}_3,\\
D_{42}^HZD_{41}^H &= H_4 - D_{42}^HZC_{41}^H - C_{42}^HZD_{41}^H - C_{42}^HZC_{41}^H := \tilde{H}_4.
\end{aligned} \tag{7}$$

Besides, Wang et al. [20] developed the modified GI (MGI) algorithm for the Sylvester equations. The information generated in the first half-iterative step is fully exploited and used to compute the result of the second half-iterative step, which leads to faster convergence rate. This motivates us to apply the updated technique in [20] to (7) and construct the following IMGI algorithm based on Lemma 3.1.

The IMGI algorithm:

Step 1: Given matrices $A_i, B_i, H \in \mathbb{C}^{n\times n}$ $(i = 1, 2, 3, 4)$, and two constants $\varepsilon > 0$ and $\mu > 0$. Choose the initial matrices $Z(0)$ and $Z_i(0)(i = 1, 2, 3, 4)$, and set $k = 0$;

Step 2: If $\delta_k = \frac{\|H - A_1Z(k)B_1 - A_2\bar{Z}(k)B_2 - A_3Z(k)^TB_3 - A_4Z(k)^HB_4\|}{\|H\|} < \varepsilon$, stop; otherwise, go to Step 3;

Step 3: Update the sequences

$$\begin{aligned}
Z^{(1)}(k+1) &= Z(k) + \mu D_{11}^H(H - \Theta(Z(k)))D_{12}^H,\\
\hat{Z}(k) &= (Z^{(1)}(k+1) + Z^{(2)}(k) + Z^{(3)}(k) + Z^{(4)}(k))/4,\\
Z^{(2)}(k+1) &= \hat{Z}(k) + \mu D_{21}^T(\bar{H} - \overline{\Theta(\hat{Z}(k))})D_{22}^T,\\
\check{Z}(k) &= (Z^{(1)}(k+1) + Z^{(2)}(k+1) + Z^{(3)}(k) + Z^{(4)}(k))/4,
\end{aligned}$$

$$
\begin{aligned}
&Z^{(3)}(k+1) = \check{Z}(k) + \mu\bar{D}_{32}(H^T - \Theta^T(\check{Z}(k)))\bar{D}_{31},\\
&\acute{Z}(k) = (Z^{(1)}(k+1) + Z^{(2)}(k+1) + Z^{(3)}(k+1) + Z^{(4)}(k))/4,\\
&Z^{(4)}(k+1) = \acute{Z}(k) + \mu D_{42}(H^H - \Theta^H(\acute{Z}(k)))D_{41},\\
&Z(k+1) = (Z^{(1)}(k+1) + Z^{(2)}(k+1) + Z^{(3)}(k+1) + Z^{(4)}(k+1))/4;
\end{aligned}
$$

Step 4: Set $k \coloneqq k+1$ and return to Step 2.

Remark 3.1. Comparing the IMGI algorithm with the GI algorithm, it can be seen that the differences between the two algorithms are the Step 3 of them. The related coefficient matrices in the GI algorithm are replaced by their diagonal parts, and the next result is computed by the latest information. This leads to less computations of each iteration and faster convergence. So we can expect that the proposed IMGI algorithm performs better than the GI one, which will be verified by numerical results.

In the sequel, we will establish the convergence condition of the IMGI algorithm. Before that, we start with a lemma as follows.

Lemma 3.2. [21] *The CCT Sylvester matrix equation* (2) *has a unique solution if and only if the matrix* $\mathcal{A}$ *is nonsingular, then the unique solution is given by*

$$\mathrm{vec}(Z_\sigma) = \mathcal{A}^{-1}\mathrm{vec}(H_\sigma), \tag{8}$$

where

$$\mathcal{A} = (P_n(B_1)_\sigma)^T \otimes ((A_1)_\sigma P_n) + (B_2)_\sigma^T \otimes (A_2)_\sigma + [(P_n(B_3)_\sigma)^T \otimes ((A_3)_\sigma P_n) + (B_4)_\sigma^T \otimes (A_4)_\sigma]P(2n, 2n). \tag{9}$$

And the corresponding homogeneous matrix equation $A_1ZB_1 + A_2\bar{Z}B_2 + A_3Z^TB_3 + A_4Z^HB_4 = 0$ *has a unique solution* $Z = 0$.

In the following, by applying the properties of the real representation of a complex matrix and the vector stretching operator, we study the necessary and sufficient condition for the convergence of the IMGI algorithm.

Theorem 3.1. *Suppose that the CCT Sylvester matrix equation* (2) *has a unique solution* Z^*. *Then the IMGI algorithm is convergent if and only if the parameter* μ *is selected to satisfy* $\rho(ST) < 1$, *where*

$$
S = \begin{pmatrix}
I & 0 & 0 & 0\\
\frac{1}{4}M_2 & I & 0 & 0\\
\frac{1}{16}M_3M_2 + \frac{1}{4}M_3 & \frac{1}{4}M_3 & I & 0\\
\frac{1}{4}M_4 + \frac{1}{16}M_4M_2 + \frac{1}{16}M_4M_3 + \frac{1}{64}M_4M_3M_2 & \frac{1}{16}M_4M_3 + \frac{1}{4}M_4 & \frac{1}{4}M_4 & I
\end{pmatrix},
\qquad
T = \frac{1}{4}\begin{pmatrix}
M_1 & M_1 & M_1 & M_1\\
0 & M_2 & M_2 & M_2\\
0 & 0 & M_3 & M_3\\
0 & 0 & 0 & M_4
\end{pmatrix},
\tag{10}
$$

with

$$
\begin{aligned}
&M_1 = I_{4n^2} - \mu W_1, \quad M_2 = I_{4n^2} - \mu W_2,\\
&M_3 = I_{4n^2} - \mu W_3, \quad M_4 = I_{4n^2} - \mu W_4,\\
&W_1 = (B_1D_{12}^H)_\sigma^T P_n \otimes (D_{11}^HA_1)_\sigma P_n + (B_2D_{12}^H)_\sigma^T \otimes (D_{11}^HA_2)_\sigma + [(B_3D_{12}^H)_\sigma^T P_n \otimes (D_{11}^HA_3)_\sigma P_n + (B_4D_{12}^H)_\sigma^T\\
&\qquad \otimes (D_{11}^HA_4)_\sigma]P(2n, 2n),
\end{aligned}
$$

$$W_2 = (\bar{B}_1 D_{22}^T)_\sigma^T \otimes (D_{21}^T \bar{A}_1)_\sigma + [(\bar{B}_3 D_{22}^T)_\sigma^T \otimes (D_{21}^T \bar{A}_3)_\sigma + (\bar{B}_4 D_{22}^T)_\sigma^T P_n \otimes (D_{21}^T \bar{A}_4)_\sigma P_n] P(2n, 2n) + (\bar{B}_2 D_{22}^T)_\sigma^T P_n \otimes (D_{21}^T \bar{A}_2)_\sigma P_n,$$

$$W_3 = [(A_1^T \bar{D}_{31})_\sigma^T P_n \otimes (\bar{D}_{32} B_1^T)_\sigma P_n + (A_2^T \bar{D}_{31})_\sigma^T \otimes (\bar{D}_{32} B_2^T)_\sigma] P(2n, 2n) + (A_3^T \bar{D}_{31})_\sigma^T P_n \otimes (\bar{D}_{32} B_3^T)_\sigma P_n + (A_4^T \bar{D}_{31})_\sigma^T \otimes (\bar{D}_{32} B_4^T)_\sigma,$$

$$W_4 = [(A_1^H D_{41})_\sigma^T \otimes (D_{42} B_1^H)_\sigma + (A_2^H D_{41})_\sigma^T P_n \otimes (D_{42} B_2^H)_\sigma P_n] P(2n, 2n) + (A_3^H D_{41})_\sigma^T \otimes (D_{42} B_3^H)_\sigma + (A_4^H D_{41})_\sigma^T P_n \otimes (D_{42} B_4^H)_\sigma P_n.$$

Proof. Define the following error matrices:

$$\begin{aligned} &\tilde{Z}(k) = Z(k) - Z^*, \quad \tilde{\hat{Z}}(k) = \hat{Z}(k) - Z^*, \quad \tilde{\check{Z}}(k) = \check{Z}(k) - X^*, \quad \tilde{\acute{Z}}(k) = \acute{Z}(k) - Z^*, \\ &\tilde{Z}^{(1)}(k) = Z^{(1)}(k) - Z^*, \quad \tilde{Z}^{(2)}(k) = Z^{(2)}(k) - Z^*, \quad \tilde{Z}^{(3)}(k) = Z^{(3)}(k) - Z^*, \quad \tilde{Z}^{(4)}(k) = Z^{(4)}(k) - Z^*. \end{aligned} \tag{11}$$

It follows from Line 1 of Step 3 of the IMGI algorithm that

$$\tilde{Z}^{(1)}(k+1) = \tilde{Z}(k) - \mu D_{11}^H (A_1 \tilde{Z}(k) B_1 + A_2 \tilde{\hat{Z}}(k) B_2 + A_3 \tilde{Z}(k)^T B_3 + A_4 \tilde{Z}(k)^H B_4) D_{12}^H,$$

which together with the real representation and Lemma 2.1 results in

$$\begin{aligned} (\tilde{Z}^{(1)}(k+1))_\sigma &= (\tilde{Z}(k))_\sigma - \mu (D_{11}^H A_1 \tilde{Z}(k) B_1 D_{12}^H + D_{11}^H A_2 \tilde{\hat{Z}}(k) B_2 D_{12}^H + D_{11}^H A_3 \tilde{Z}(k)^T B_3 D_{12}^H + D_{11}^H A_4 \tilde{Z}(k)^H B_4 D_{12}^H)_\sigma \\ &= (\tilde{Z}(k))_\sigma - \mu [(D_{11}^H A_1)_\sigma P_n (\tilde{Z}(k))_\sigma P_n (B_1 D_{12}^H)_\sigma + (D_{11}^H A_2)_\sigma (\tilde{Z}(k))_\sigma (B_2 D_{12}^H)_\sigma \\ &\quad + (D_{11}^H A_3)_\sigma P_n (\tilde{Z}(k)^T)_\sigma P_n (B_3 D_{12}^H)_\sigma + (D_{11}^H A_4)_\sigma (\tilde{Z}(k)^T)_\sigma (B_4 D_{12}^H)_\sigma]. \end{aligned} \tag{12}$$

Using straightening operator on both sides of relation (12) yields that

$$\begin{aligned} \mathrm{vec}[(\tilde{Z}^{(1)}(k+1))_\sigma] &= \mathrm{vec}[(\tilde{Z}(k))_\sigma] - \mu \mathrm{vec}[(D_{11}^H A_1)_\sigma P_n (\tilde{Z}(k))_\sigma P_n (B_1 D_{12}^H)_\sigma + (D_{11}^H A_2)_\sigma (\tilde{Z}(k))_\sigma (B_2 D_{12}^H)_\sigma \\ &\quad + (D_{11}^H A_3)_\sigma P_n (\tilde{Z}(k)^T)_\sigma P_n (B_3 D_{12}^H)_\sigma + (D_{11}^H A_4)_\sigma (\tilde{Z}(k)^T)_\sigma (B_4 D_{12}^H)_\sigma] \\ &= \mathrm{vec}[(\tilde{Z}(k))_\sigma] - \mu \{(B_1 D_{12}^H)_\sigma^T P_n \otimes (D_{11}^H A_1)_\sigma P_n + (B_2 D_{12}^H)_\sigma^T \otimes (D_{11}^H A_2)_\sigma \\ &\quad + [(B_3 D_{12}^H)_\sigma^T P_n \otimes (D_{11}^H A_3)_\sigma P_n + (B_4 D_{12}^H)_\sigma^T \otimes (D_{11}^H A_4)_\sigma] P(2n, 2n)\} \mathrm{vec}[(\tilde{Z}(k))_\sigma] \\ &= [I_{4n^2} - \mu W_1] \mathrm{vec}((\tilde{Z}(k))_\sigma) \coloneqq M_1 \mathrm{vec}[(\tilde{Z}(k))_\sigma], \end{aligned} \tag{13}$$

where

$$\begin{aligned} W_1 &= (B_1 D_{12}^H)_\sigma^T P_n \otimes (D_{11}^H A_1)_\sigma P_n + (B_2 D_{12}^H)_\sigma^T \otimes (D_{11}^H A_2)_\sigma + [(B_3 D_{12}^H)_\sigma^T P_n \otimes (D_{11}^H A_3)_\sigma P_n + (B_4 D_{12}^H)_\sigma^T \\ &\quad \otimes (D_{11}^H A_4)_\sigma] P(2n, 2n), \\ M_1 &= I_{4n^2} - \mu W_1. \end{aligned}$$

From Line 3 of Step 3 of the IMGI algorithm, we have

$$\begin{aligned} \tilde{Z}^{(2)}(k+1) &= \tilde{\hat{Z}}(k) - \mu D_{21}^T \overline{(A_1 \tilde{\hat{Z}}(k) B_1 + A_2 \tilde{\hat{Z}}(k) B_2 + A_3 \tilde{\hat{Z}}(k)^T B_3 + A_4 \tilde{\hat{Z}}(k)^H B_4)} D_{22}^T \\ &= \tilde{\hat{Z}}(k) - \mu (D_{21}^T \bar{A}_1 \tilde{\hat{Z}}(k) \bar{B}_1 D_{22}^T + D_{21}^T \bar{A}_2 \tilde{\hat{Z}}(k) \bar{B}_2 D_{22}^T + D_{21}^T \bar{A}_3 \tilde{\hat{Z}}(k)^T \bar{B}_3 D_{22}^T + D_{21}^T \bar{A}_4 \tilde{\hat{Z}}(k)^H \bar{B}_4 D_{22}^T). \end{aligned} \tag{14}$$

Taking the real representation of the both sides of (14) and using Lemma 2.1 result in

$$\begin{aligned} (\tilde{Z}^{(2)}(k+1))_\sigma &= (\tilde{\hat{Z}}(k))_\sigma - \mu (D_{21}^T \bar{A}_1 \tilde{\hat{Z}}(k) \bar{B}_1 D_{22}^T + D_{21}^T \bar{A}_2 \tilde{\hat{Z}}(k) \bar{B}_2 D_{22}^T + D_{21}^T \bar{A}_3 \tilde{\hat{Z}}(k)^T \bar{B}_3 D_{22}^T + D_{21}^T \bar{A}_4 \tilde{\hat{Z}}(k)^H \bar{B}_4 D_{22}^T)_\sigma \\ &= (\tilde{\hat{Z}}(k))_\sigma - \mu [(D_{21}^T \bar{A}_1)_\sigma (\tilde{\hat{Z}}(k))_\sigma (\bar{B}_1 D_{22}^T)_\sigma + (D_{21}^T \bar{A}_2)_\sigma P_n (\tilde{\hat{Z}}(k))_\sigma P_n (\bar{B}_2 D_{22}^T)_\sigma \\ &\quad + (D_{21}^T \bar{A}_3)_\sigma (\tilde{\hat{Z}}(k)^T)_\sigma (\bar{B}_3 D_{22}^T)_\sigma + (D_{21}^T \bar{A}_4)_\sigma P_n \tilde{\hat{Z}}(k)^T P_n (\bar{B}_4 D_{22}^T)_\sigma]. \end{aligned} \tag{15}$$

Applying Kronecker product and vec operator in (15) leads to

$$\begin{aligned} \mathrm{vec}[(\tilde{Z}^{(2)}(k+1))_\sigma] &= \mathrm{vec}[(\tilde{\hat{Z}}(k))_\sigma] - \mu \mathrm{vec}[(D_{21}^T \bar{A}_1)_\sigma (\tilde{\hat{Z}}(k))_\sigma (\bar{B}_1 D_{22}^T)_\sigma + (D_{21}^T \bar{A}_2)_\sigma P_n (\tilde{\hat{Z}}(k))_\sigma P_n (\bar{B}_2 D_{22}^T)_\sigma \\ &\quad + (D_{21}^T \bar{A}_3)_\sigma (\tilde{\hat{Z}}(k)^T)_\sigma (\bar{B}_3 D_{22}^T)_\sigma + (D_{21}^T \bar{A}_4)_\sigma P_n \tilde{\hat{Z}}(k)^T P_n (\bar{B}_4 D_{22}^T)_\sigma] \\ &= \mathrm{vec}[(\tilde{\hat{Z}}(k))_\sigma] - \mu \{(\bar{B}_1 D_{22}^T)_\sigma^T \otimes (D_{21}^T \bar{A}_1)_\sigma + (\bar{B}_2 D_{22}^T)_\sigma^T P_n \otimes (D_{21}^T \bar{A}_2)_\sigma P_n \end{aligned} \tag{16}$$

$$
\begin{aligned}
&+ [(\bar{B}_3 D_{22}^T)_\sigma^T \otimes (D_{21}^T \bar{A}_3)_\sigma + (\bar{B}_4 D_{22}^T)_\sigma^T P_n \otimes (D_{21}^T \bar{A}_4)_\sigma P_n] P(2n, 2n)\}\mathrm{vec}[(\tilde{\tilde{Z}}(k))_\sigma] \\
&= [I_{4n^2} - \mu W_2]\mathrm{vec}[(\tilde{\tilde{Z}}(k))_\sigma] \coloneqq M_2 \mathrm{vec}[(\tilde{\tilde{Z}}(k))_\sigma],
\end{aligned}
$$

with

$$
\begin{aligned}
W_2 &= (\bar{B}_1 D_{22}^T)_\sigma^T \otimes (D_{21}^T \bar{A}_1)_\sigma + [(\bar{B}_3 D_{22}^T)_\sigma^T \otimes (D_{21}^T \bar{A}_3)_\sigma + (\bar{B}_4 D_{22}^T)_\sigma^T P_n \otimes (D_{21}^T \bar{A}_4)_\sigma P_n] P(2n, 2n) + (\bar{B}_2 D_{22}^T)_\sigma^T P_n \\
&\quad \otimes (D_{21}^T \bar{A}_2)_\sigma P_n, \\
M_2 &= I_{4n^2} - \mu W_2.
\end{aligned}
$$

According to Line 5 of Step 3 of the IMGI algorithm, it has

$$
\begin{aligned}
\tilde{Z}^{(3)}(k+1) &= \tilde{\tilde{Z}}(k) - \mu \bar{D}_{32}(A_1 \tilde{\tilde{Z}}(k) B_1 + A_2 \tilde{\tilde{\tilde{Z}}}(k) B_2 + A_3 \tilde{\tilde{Z}}(k)^T B_3 + A_4 \tilde{\tilde{Z}}(k)^H B_4)^T \bar{D}_{31} \\
&= \tilde{\tilde{Z}}(k) - \mu \bar{D}_{32}(B_1^T \tilde{\tilde{Z}}(k)^T A_1^T + B_2^T \tilde{\tilde{\tilde{Z}}}(k)^T A_2^T + B_3^T \tilde{\tilde{Z}}(k) A_3^T + B_4^T \tilde{\tilde{\tilde{Z}}}(k) A_4^T) \bar{D}_{31} \\
&= \tilde{\tilde{Z}}(k) - \mu(\bar{D}_{32} B_1^T \tilde{\tilde{Z}}(k)^T A_1^T \bar{D}_{31} + \bar{D}_{32} B_2^T \tilde{\tilde{\tilde{Z}}}(k)^T A_2^T \bar{D}_{31} + \bar{D}_{32} B_3^T \tilde{\tilde{Z}}(k) A_3^T \bar{D}_{31} + \bar{D}_{32} B_4^T \tilde{\tilde{\tilde{Z}}}(k) A_4^T \bar{D}_{31}).
\end{aligned} \tag{17}
$$

By making use of the real representation of the both sides of (17) and Lemma 2.1, we derive

$$
\begin{aligned}
(\tilde{Z}^{(3)}(k+1))_\sigma &= (\tilde{\tilde{Z}}(k))_\sigma - \mu(\bar{D}_{32} B_1^T \tilde{\tilde{Z}}(k)^T A_1^T \bar{D}_{31} + \bar{D}_{32} B_2^T \tilde{\tilde{\tilde{Z}}}(k)^T A_2^T \bar{D}_{31} + \bar{D}_{32} B_3^T \tilde{\tilde{Z}}(k) A_3^T \bar{D}_{31} + \bar{D}_{32} B_4^T \tilde{\tilde{\tilde{Z}}}(k) A_4^T \bar{D}_{31})_\sigma \\
&= (\tilde{\tilde{Z}}(k))_\sigma - \mu[(\bar{D}_{32} B_1^T)_\sigma P_n (\tilde{\tilde{Z}}(k)^T)_\sigma P_n (A_1^T \bar{D}_{31})_\sigma + (\bar{D}_{32} B_2^T)_\sigma (\tilde{\tilde{\tilde{Z}}}(k)^T)_\sigma (A_2^T \bar{D}_{31})_\sigma \\
&\quad + (\bar{D}_{32} B_3^T)_\sigma P_n (\tilde{\tilde{Z}}(k))_\sigma P_n (A_3^T \bar{D}_{31})_\sigma + (\bar{D}_{32} B_4^T)_\sigma (\tilde{\tilde{\tilde{Z}}}(k))_\sigma (A_4^T \bar{D}_{31})_\sigma].
\end{aligned} \tag{18}
$$

By utilizing Kronecker product and vec operator in (18), we deduce that

$$
\begin{aligned}
\mathrm{vec}[(\tilde{Z}^{(3)}(k+1))_\sigma] &= \mathrm{vec}[(\tilde{\tilde{Z}}(k))_\sigma] - \mu \mathrm{vec}[(\bar{D}_{32} B_1^T)_\sigma P_n (\tilde{\tilde{Z}}(k)^T)_\sigma P_n (A_1^T \bar{D}_{31})_\sigma + (\bar{D}_{32} B_2^T)_\sigma (\tilde{\tilde{\tilde{Z}}}(k)^T)_\sigma (A_2^T \bar{D}_{31})_\sigma \\
&\quad + (\bar{D}_{32} B_3^T)_\sigma P_n (\tilde{\tilde{Z}}(k))_\sigma P_n (A_3^T \bar{D}_{31})_\sigma + (\bar{D}_{32} B_4^T)_\sigma (\tilde{\tilde{\tilde{Z}}}(k))_\sigma (A_4^T \bar{D}_{31})_\sigma] \\
&= \mathrm{vec}[(\tilde{\tilde{Z}}(k))_\sigma] - \mu\{[(A_1^T \bar{D}_{31})_\sigma^T P_n \otimes (\bar{D}_{32} B_1^T)_\sigma P_n + (A_2^T \bar{D}_{31})_\sigma^T \otimes (\bar{D}_{32} B_2^T)_\sigma] P(2n, 2n) \\
&\quad + (A_3^T \bar{D}_{31})_\sigma^T P_n \otimes (\bar{D}_{32} B_3^T)_\sigma P_n + (A_4^T \bar{D}_{31})_\sigma^T \otimes (\bar{D}_{32} B_4^T)_\sigma\} \mathrm{vec}[(\tilde{\tilde{\tilde{Z}}}(k))_\sigma] \\
&= [I_{4n^2} - \mu W_3]\mathrm{vec}[(\tilde{\tilde{Z}}(k))_\sigma] \coloneqq M_3 \mathrm{vec}[(\tilde{\tilde{Z}}(k))_\sigma],
\end{aligned} \tag{19}
$$

with

$$
\begin{aligned}
W_3 &= [(A_1^T \bar{D}_{31})_\sigma^T P_n \otimes (\bar{D}_{32} B_1^T)_\sigma P_n + (A_2^T \bar{D}_{31})_\sigma^T \otimes (\bar{D}_{32} B_2^T)_\sigma] P(2n, 2n) + (A_3^T \bar{D}_{31})_\sigma^T P_n \otimes (\bar{D}_{32} B_3^T)_\sigma P_n + (A_4^T \bar{D}_{31})_\sigma^T \\
&\quad \otimes (\bar{D}_{32} B_4^T)_\sigma, \\
M_3 &= I_{4n^2} - \mu W_3.
\end{aligned}
$$

In the same manner applied in (18), we can deduce that

$$
\begin{aligned}
\tilde{Z}^{(4)}(k+1) &= \tilde{\tilde{Z}}(k) - \mu D_{42}(A_1 \tilde{\tilde{Z}}(k) B_1 + A_2 \tilde{\tilde{\tilde{Z}}}(k) B_2 + A_3 \tilde{\tilde{Z}}(k)^T B_3 + A_4 \tilde{\tilde{Z}}(k)^H B_4)^H D_{41} \\
&= \tilde{\tilde{Z}}(k) - \mu D_{42}(B_1^H \tilde{\tilde{Z}}(k)^H A_1^H + B_2^H \tilde{\tilde{Z}}(k)^T A_2^H + B_3^H \tilde{\tilde{\tilde{Z}}}(k) A_3^H + B_4^H \tilde{\tilde{Z}}(k) A_4^H) D_{41} \\
&= \tilde{\tilde{Z}}(k) - \mu(D_{42} B_1^H \tilde{\tilde{Z}}(k)^H A_1^H D_{41} + D_{42} B_2^H \tilde{\tilde{Z}}(k)^T A_2^H D_{41} + D_{42} B_3^H \tilde{\tilde{\tilde{Z}}}(k) A_3^H D_{41} + D_{42} B_4^H \tilde{\tilde{Z}}(k) A_4^H D_{41}),
\end{aligned} \tag{20}
$$

by Line 7 of Step 3 of the IMGI algorithm. Using the real representation in (20) and Lemma 2.1 yields that

$$
\begin{aligned}
(\tilde{Z}^{(4)}(k+1))_\sigma &= (\tilde{\tilde{Z}}(k))_\sigma - \mu(D_{42} B_1^H \tilde{\tilde{Z}}(k)^H A_1^H D_{41} + D_{42} B_2^H \tilde{\tilde{Z}}(k)^T A_2^H D_{41} + D_{42} B_3^H \tilde{\tilde{\tilde{Z}}}(k) A_3^H D_{41} + D_{42} B_4^H \tilde{\tilde{Z}}(k) A_4^H D_{41})_\sigma \\
&= (\tilde{\tilde{Z}}(k))_\sigma - \mu[(D_{42} B_1^H)_\sigma (\tilde{\tilde{Z}}(k)^T)_\sigma (A_1^H D_{41})_\sigma + (D_{42} B_2^H)_\sigma P_n (\tilde{\tilde{Z}}(k)^T)_\sigma P_n (A_2^H D_{41})_\sigma \\
&\quad + (D_{42} B_3^H)_\sigma (\tilde{\tilde{\tilde{Z}}}(k))_\sigma (A_3^H D_{41})_\sigma + (D_{42} B_4^H)_\sigma P_n (\tilde{\tilde{Z}}(k))_\sigma P_n (A_4^H D_{41})_\sigma].
\end{aligned} \tag{21}
$$

By utilizing Kronecker product and vec operator in (21), it holds that

$$
\mathrm{vec}[(\tilde{Z}^{(4)}(k+1))_\sigma] = \mathrm{vec}[(\tilde{\tilde{Z}}(k))_\sigma] - \mu \mathrm{vec}[(D_{42} B_1^H)_\sigma (\tilde{\tilde{Z}}(k)^T)_\sigma (A_1^H D_{41})_\sigma + (D_{42} B_2^H)_\sigma P_n (\tilde{\tilde{Z}}(k)^T)_\sigma P_n (A_2^H D_{41})_\sigma \tag{22}
$$

$$
\begin{aligned}
&+ (D_{42}B_3^H)_\sigma(\tilde{\tilde{Z}}(k))_\sigma(A_3^H D_{41})_\sigma + (D_{42}B_4^H)_\sigma P_n(\tilde{\tilde{Z}}(k))_\sigma P_n(A_4^H D_{41})_\sigma]\\
&= \mathrm{vec}[(\tilde{\tilde{Z}}(k))_\sigma] - \mu\{[(A_1^H D_{41})_\sigma^T \otimes (D_{42}B_1^H)_\sigma + (A_2^H D_{41})_\sigma^T P_n \otimes (D_{42}B_2^H)_\sigma P_n]P(2n, 2n)\\
&\quad + (A_3^H D_{41})_\sigma^T \otimes (D_{42}B_3^H)_\sigma + (A_4^H D_{41})_\sigma^T P_n \otimes (D_{42}B_4^H)_\sigma P_n\}\mathrm{vec}[(\tilde{\tilde{Z}}(k))_\sigma]\\
&= [I_{4n^2} - \mu W_4]\mathrm{vec}[(\tilde{\tilde{Z}}(k))_\sigma] \coloneqq M_4\mathrm{vec}[(\tilde{\tilde{Z}}(k))_\sigma],
\end{aligned}
$$

with

$$
\begin{aligned}
W_4 &= [(A_1^H D_{41})_\sigma^T \otimes (D_{42}B_1^H)_\sigma + (A_2^H D_{41})_\sigma^T P_n \otimes (D_{42}B_2^H)_\sigma P_n]P(2n, 2n) + (A_3^H D_{41})_\sigma^T \otimes (D_{42}B_3^H)_\sigma + (A_4^H D_{41})_\sigma^T P_n\\
&\quad \otimes (D_{42}B_4^H)_\sigma P_n,\\
M_4 &= I_{4n^2} - \mu W_4.
\end{aligned}
$$

Combining (13) with Line 8 of Step 3 of IMGI algorithm leads to

$$
\begin{aligned}
\mathrm{vec}[(\tilde{Z}^{(1)}(k+1))_\sigma] &= M_1\mathrm{vec}[(\tilde{Z}(k))_\sigma]\\
&= M_1\mathrm{vec}\left[\left(\frac{1}{4}\tilde{Z}^{(1)}(k) + \frac{1}{4}\tilde{Z}^{(2)}(k) + \frac{1}{4}\tilde{Z}^{(3)}(k) + \frac{1}{4}\tilde{Z}^{(4)}(k)\right)_\sigma\right]\\
&= \frac{1}{4}M_1\mathrm{vec}[(\tilde{Z}^{(1)}(k))_\sigma] + \frac{1}{4}M_1\mathrm{vec}[(\tilde{Z}^{(2)}(k))_\sigma]\\
&\quad + \frac{1}{4}M_1\mathrm{vec}[(\tilde{Z}^{(3)}(k))_\sigma]\\
&\quad + \frac{1}{4}M_1\mathrm{vec}[(\tilde{Z}^{(4)}(k))_\sigma].
\end{aligned}
\tag{23}
$$

Besides, according to (16), (23), and Line 2 of Step 3 of IMGI algorithm, we obtain

$$
\begin{aligned}
\mathrm{vec}[(\tilde{Z}^{(2)}(k+1))_\sigma] &= M_2\mathrm{vec}[(\tilde{\tilde{Z}}(k))_\sigma]\\
&= M_2\mathrm{vec}\left[\left(\frac{1}{4}\tilde{Z}^{(1)}(k+1) + \frac{1}{4}\tilde{Z}^{(2)}(k) + \frac{1}{4}\tilde{Z}^{(3)}(k) + \frac{1}{4}\tilde{Z}^{(4)}(k)\right)_\sigma\right]\\
&= \frac{1}{4}M_2\mathrm{vec}[(\tilde{Z}^{(1)}(k+1))_\sigma] + \frac{1}{4}M_2\mathrm{vec}[(\tilde{Z}^{(2)}(k))_\sigma] + \frac{1}{4}M_2\mathrm{vec}[(\tilde{Z}^{(3)}(k))_\sigma]\\
&\quad + \frac{1}{4}M_2\mathrm{vec}[(\tilde{Z}^{(4)}(k))_\sigma]\\
&= \frac{1}{4}M_2\left\{\frac{1}{4}M_1\mathrm{vec}[(\tilde{Z}^{(1)}(k))_\sigma] + \frac{1}{4}M_1\mathrm{vec}[(\tilde{Z}^{(2)}(k))_\sigma] + \frac{1}{4}M_1\mathrm{vec}[(\tilde{Z}^{(3)}(k))_\sigma]\right.\\
&\quad \left. + \frac{1}{4}M_1\mathrm{vec}[(\tilde{Z}^{(4)}(k))_\sigma]\right\} + \frac{1}{4}M_2\mathrm{vec}(\tilde{Z}^{(2)}(k))\\
&\quad + \frac{1}{4}M_2\mathrm{vec}[(\tilde{Z}^{(3)}(k))_\sigma] + \frac{1}{4}M_2\mathrm{vec}[(\tilde{Z}^{(4)}(k))_\sigma]\\
&= \frac{1}{16}M_2M_1\mathrm{vec}[(\tilde{Z}^{(1)}(k))_\sigma] + \left(\frac{1}{16}M_2M_1 + \frac{1}{4}M_2\right)\mathrm{vec}[(\tilde{Z}^{(2)}(k))_\sigma]\\
&\quad + \left(\frac{1}{16}M_2M_1 + \frac{1}{4}M_2\right)\mathrm{vec}[(\tilde{Z}^{(3)}(k))_\sigma] + \left(\frac{1}{16}M_2M_1 + \frac{1}{4}M_2\right)\mathrm{vec}[(\tilde{Z}^{(4)}(k))_\sigma].
\end{aligned}
\tag{24}
$$

It follows from (19), (23), (24), and Line 4 of Step 3 of the IMGI algorithm that

$$
\begin{aligned}
\mathrm{vec}[(\tilde{Z}^{(3)}(k+1))_\sigma] &= M_3\mathrm{vec}[(\tilde{\tilde{Z}}(k))_\sigma]\\
&= M_3\mathrm{vec}\left[\left(\frac{1}{4}\tilde{Z}^{(1)}(k+1) + \frac{1}{4}\tilde{Z}^{(2)}(k+1) + \frac{1}{4}\tilde{Z}^{(3)}(k) + \frac{1}{4}\tilde{Z}^{(4)}(k)\right)_\sigma\right]\\
&= \frac{1}{4}M_3\mathrm{vec}[(\tilde{Z}^{(1)}(k+1))_\sigma] + \frac{1}{4}M_3\mathrm{vec}[(\tilde{Z}^{(2)}(k+1))_\sigma] + \frac{1}{4}M_3\mathrm{vec}[(\tilde{Z}^{(3)}(k))_\sigma]\\
&\quad + \frac{1}{4}M_3\mathrm{vec}[(\tilde{Z}^{(4)}(k))_\sigma]
\end{aligned}
\tag{25}
$$

$$
\begin{aligned}
&= \left(\frac{1}{64}M_3M_2M_1 + \frac{1}{16}M_3M_1\right)\mathrm{vec}[(\tilde{Z}^{(1)}(k))_\sigma] + \left(\frac{1}{16}M_3M_1 + \frac{1}{64}M_3M_2M_1\right.\\
&\quad \left. + \frac{1}{16}M_3M_2\right)\mathrm{vec}[(\tilde{Z}^{(2)}(k))_\sigma] + \left(\frac{1}{16}M_3M_1 + \frac{1}{64}M_3M_2M_1 + \frac{1}{16}M_3M_2 + \frac{1}{4}M_3\right)\\
&\quad \times \mathrm{vec}[(\tilde{Z}^{(3)}(k))_\sigma] + \left(\frac{1}{16}M_3M_1 + \frac{1}{64}M_3M_2M_1 + \frac{1}{16}M_3M_2 + \frac{1}{4}M_3\right)\mathrm{vec}[(\tilde{Z}^{(4)}(k))_\sigma].
\end{aligned}
$$

Finally, combining (22)–(25) with Line 6 of Step 3 of IMGI algorithm leads to

$$
\begin{aligned}
\mathrm{vec}[(\tilde{Z}^{(4)}(k+1))_\sigma] &= M_4\mathrm{vec}[(\tilde{\tilde{Z}}(k))_\sigma]\\
&= M_4\mathrm{vec}\left[\left(\frac{1}{4}\tilde{Z}^{(1)}(k+1) + \frac{1}{4}\tilde{Z}^{(2)}(k+1) + \frac{1}{4}\tilde{Z}^{(3)}(k+1) + \frac{1}{4}\tilde{Z}^{(4)}(k)\right)_\sigma\right]\\
&= \frac{1}{4}M_4\mathrm{vec}[(\tilde{Z}^{(1)}(k+1))_\sigma] + \frac{1}{4}M_4\mathrm{vec}[(\tilde{Z}^{(2)}(k+1))_\sigma] + \frac{1}{4}M_4\mathrm{vec}[(\tilde{Z}^{(3)}(k+1))_\sigma]\\
&\quad + \frac{1}{4}M_4\mathrm{vec}[(\tilde{Z}^{(4)}(k))_\sigma]\\
&= \left(\frac{1}{16}M_4M_1 + \frac{1}{64}M_4M_2M_1 + \frac{1}{64}M_4M_3M_1 + \frac{1}{256}M_4M_3M_2M_1\right)\mathrm{vec}[(\tilde{Z}^{(1)}(k))_\sigma]\\
&\quad + \left(\frac{1}{16}M_4M_1 + \frac{1}{16}M_4M_2 + \frac{1}{64}M_4M_2M_1 + \frac{1}{64}M_4M_3M_1 + \frac{1}{64}M_4M_3M_2\right.\\
&\quad \left. + \frac{1}{256}M_4M_3M_2M_1\right)\mathrm{vec}[(\tilde{Z}^{(2)}(k))_\sigma] + \left(\frac{1}{16}M_4M_1 + \frac{1}{16}M_4M_2 + \frac{1}{16}M_4M_3 + \frac{1}{64}M_4M_2M_1\right.\\
&\quad \left. + \frac{1}{64}M_4M_3M_1 + \frac{1}{64}M_4M_3M_2 + \frac{1}{256}M_4M_3M_2M_1\right)\mathrm{vec}[(\tilde{Z}^{(3)}(k))_\sigma]\\
&\quad + \left(\frac{1}{16}M_4M_1 + \frac{1}{16}M_4M_2 + \frac{1}{16}M_4M_3 + \frac{1}{64}M_4M_2M_1\right.\\
&\quad \left. + \frac{1}{64}M_4M_3M_1 + \frac{1}{64}M_4M_3M_2 + \frac{1}{256}M_4M_3M_2M_1 + \frac{1}{4}M_4\right)\mathrm{vec}[(\tilde{Z}^{(4)}(k))_\sigma].
\end{aligned}
\tag{26}
$$

According to (23)–(26) and by some computations, we derive

$$
\begin{pmatrix}
\mathrm{vec}[(\tilde{Z}^{(1)}(k+1))_\sigma]\\
\mathrm{vec}[(\tilde{Z}^{(2)}(k+1))_\sigma]\\
\mathrm{vec}[(\tilde{Z}^{(3)}(k+1))_\sigma]\\
\mathrm{vec}[(\tilde{Z}^{(4)}(k+1))_\sigma]
\end{pmatrix} = ST\begin{pmatrix}
\mathrm{vec}[(\tilde{Z}^{(1)}(k))_\sigma]\\
\mathrm{vec}[(\tilde{Z}^{(2)}(k))_\sigma]\\
\mathrm{vec}[(\tilde{Z}^{(3)}(k))_\sigma]\\
\mathrm{vec}[(\tilde{Z}^{(4)}(k))_\sigma]
\end{pmatrix},
\tag{27}
$$

where the matrices S and T are defined as in (10). Then the matrix ST is the iteration matrix of the IMGI algorithm, and hence, the IMGI algorithm is convergent if and only if $\rho(ST) < 1$. □

Note that the order of the matrix ST in Theorem 3.1 is very high if n is large. This results in difficulty of computing the spectral radius of the matrix ST. To overcome this drawback, we deduce the following corollary in terms of Lemma 2.4, which is easier to be computed compared with Theorem 3.1 because the order of related matrix is only 4.

Corollary 3.1. *Suppose that the conditions of* Theorem 3.1*are satisfied and let* $ST = [L_{ij}]_{4\times4}$. *Then the IMGI algorithm is convergent if the parameter* μ *is chosen to satisfy*

$$
\rho\begin{pmatrix}
\|L_{11}\| & \|L_{12}\| & \|L_{13}\| & \|L_{14}\|\\
\|L_{21}\| & \|L_{22}\| & \|L_{23}\| & \|L_{24}\|\\
\|L_{31}\| & \|L_{32}\| & \|L_{33}\| & \|L_{34}\|\\
\|L_{41}\| & \|L_{42}\| & \|L_{43}\| & \|L_{44}\|
\end{pmatrix} < 1,
$$

with $\|\cdot\|$ *being the matrix norm of a matrix.*

Although the convergence conditions of the IMGI algorithm are derived in Theorem 3.1 and Corollary 3.1, they do not determine any interval of the step size factor μ. The following theorem proposes the interval of the step size factor μ in the IMGI algorithm under proper conditions.

Theorem 3.2. *Let the conditions of* Theorem 3.1 *be satisfied, and S and T defined in* Theorem 3.1 *be normal matrices. Denote by $\xi_i^{(1)}$, $\xi_i^{(2)}$, $\xi_i^{(3)}$, and $\xi_i^{(4)}$ $(i = 1, 2, \ldots, 4n^2)$ the eigenvalues of the matrices W_1, W_2, W_3, and W_4, respectively, and $\mathrm{Re}(\xi_i^{(j)}) > 0$ for $i = 1, 2, \ldots, 4n^2$; $j = 1, 2, 3, 4$. Then the IMGI algorithm is convergent if*

$$0 < \mu < \min_{1\le i\le 4n^2}\left\{\frac{2\mathrm{Re}(\xi_i^{(1)})}{\mathrm{Re}(\xi_i^{(1)})^2 + \mathrm{Im}(\xi_i^{(1)})^2}, \frac{2\mathrm{Re}(\xi_i^{(2)})}{\mathrm{Re}(\xi_i^{(2)})^2 + \mathrm{Im}(\xi_i^{(2)})^2}, \frac{2\mathrm{Re}(\xi_i^{(3)})}{\mathrm{Re}(\xi_i^{(3)})^2 + \mathrm{Im}(\xi_i^{(3)})^2}, \frac{2\mathrm{Re}(\xi_i^{(4)})}{\mathrm{Re}(\xi_i^{(4)})^2 + \mathrm{Im}(\xi_i^{(4)})^2}\right\}. \tag{28}$$

Proof. Inasmuch as S and T are normal matrices, it follows from Lemma 2.5 that

$$\begin{aligned}
\rho(ST) &\le \|ST\|_2 \le \|S\|_2\|T\|_2 = \rho(S)\rho(T) = \rho(T)\\
&= \max\left\{\rho\left(\frac{1}{4}M_1\right), \rho\left(\frac{1}{4}M_2\right), \rho\left(\frac{1}{4}M_3\right), \rho\left(\frac{1}{4}M_4\right)\right\}\\
&= \max_{1\le i\le 4n^2}\left\{\left|\frac{1}{4}(1-\mu\xi_i^{(1)})\right|, \left|\frac{1}{4}(1-\mu\xi_i^{(2)})\right|, \left|\frac{1}{4}(1-\mu\xi_i^{(3)})\right|, \left|\frac{1}{4}(1-\mu\xi_i^{(4)})\right|\right\}\\
&\le \max_{1\le i\le 4n^2}\{|1-\mu\xi_i^{(1)}|, |1-\mu\xi_i^{(2)}|, |1-\mu\xi_i^{(3)}|, |1-\mu\xi_i^{(4)}|\}.
\end{aligned}$$

This implies that $\rho(ST) < 1$ holds if

$$|1-\mu\xi_i^{(1)}| < 1, \quad |1-\mu\xi_i^{(2)}| < 1, \quad |1-\mu\xi_i^{(3)}| < 1, \quad |1-\mu\xi_i^{(4)}| < 1, \quad i = 1, 2, \ldots, 4n^2,$$

which is equivalent to

$$\begin{aligned}
&(1-\mu\mathrm{Re}(\xi_i^{(1)}))^2 + \mu^2\mathrm{Im}(\xi_i^{(1)})^2 < 1, \quad (1-\mu\mathrm{Re}(\xi_i^{(2)}))^2 + \mu^2\mathrm{Im}(\xi_i^{(2)})^2 < 1,\\
&(1-\mu\mathrm{Re}(\xi_i^{(3)}))^2 + \mu^2\mathrm{Im}(\xi_i^{(3)})^2 < 1, \quad (1-\mu\mathrm{Re}(\xi_i^{(4)}))^2 + \mu^2\mathrm{Im}(\xi_i^{(4)})^2 < 1, \quad i = 1, 2, \ldots, 4n^2.
\end{aligned}$$

Solving the aforementioned inequalities yields that

$$\begin{aligned}
&0 < \mu < \frac{2\mathrm{Re}(\xi_i^{(1)})}{\mathrm{Re}(\xi_i^{(1)})^2 + \mathrm{Im}(\xi_i^{(1)})^2}, \quad 0 < \mu < \frac{2\mathrm{Re}(\xi_i^{(2)})}{\mathrm{Re}(\xi_i^{(2)})^2 + \mathrm{Im}(\xi_i^{(2)})^2},\\
&0 < \mu < \frac{2\mathrm{Re}(\xi_i^{(3)})}{\mathrm{Re}(\xi_i^{(3)})^2 + \mathrm{Im}(\xi_i^{(3)})^2}, \quad 0 < \mu < \frac{2\mathrm{Re}(\xi_i^{(4)})}{\mathrm{Re}(\xi_i^{(4)})^2 + \mathrm{Im}(\xi_i^{(4)})^2},
\end{aligned}$$

which gives the convergence condition (28) of the IMGI algorithm. □

In the sequel, we establish the different convergence condition of the IMGI algorithm below.

Theorem 3.3. *Assume that the CCT Sylvester matrix equation* (2) *has a unique solution Z^*. If μ satisfies*

$$0 < \mu < \min\left\{\frac{2}{\|D_{11}\|_2^2\|D_{12}\|_2^2}, \frac{2}{\|D_{21}\|_2^2\|D_{22}\|_2^2}, \frac{2}{\|D_{31}\|_2^2\|D_{32}\|_2^2}, \frac{2}{\|D_{41}\|_2^2\|D_{42}\|_2^2}\right\}, \tag{29}$$

then the matrix sequence $\{Z(k)\}$ generated by the IMGI algorithm converges to Z^*.

Proof. We first define the error matrices

$$\tilde{Z}(k) = Z(k) - Z^*, \quad \tilde{\hat{Z}}(k) = \hat{Z}(k) - Z^*, \quad \tilde{\check{Z}}(k) = \check{Z}(k) - Z^*, \quad \tilde{\acute{Z}}(k) = \acute{Z}(k) - Z^*, \tag{30}$$

and

$$P_1^{(k)} = A_1\tilde{Z}(k)B_1, \quad P_2^{(k)} = A_1\tilde{\hat{Z}}(k)B_1, \quad P_3^{(k)} = A_1\tilde{\check{Z}}(k)B_1, \quad P_4^{(k)} = A_1\tilde{\acute{Z}}(k)B_1,$$

$$
\begin{aligned}
&Q_1^{(k)} = A_2\bar{\tilde{Z}}(k)B_2, \quad Q_2^{(k)} = A_2\bar{\tilde{\hat{Z}}}(k)B_2, \quad Q_3^{(k)} = A_2\bar{\tilde{\check{Z}}}(k)B_2, \quad Q_4^{(k)} = A_2\bar{\tilde{Z}}(k)B_2,\\
&U_1^{(k)} = A_3\tilde{Z}^T(k)B_3, \quad U_2^{(k)} = A_3\tilde{\hat{Z}}^T(k)B_3, \quad U_3^{(k)} = A_3\tilde{\check{Z}}^T(k)B_3, \quad U_4^{(k)} = A_3\tilde{Z}(k)^TB_3,\\
&V_1^{(k)} = A_4\tilde{Z}^H(k)B_4, \quad V_2^{(k)} = A_4\tilde{\hat{Z}}^H(k)B_4, \quad V_3^{(k)} = A_4\tilde{\check{Z}}^H(k)B_4, \quad V_4^{(k)} = A_4\tilde{Z}(k)^HB_4,\\
&W_1^{(k)} = P_1^{(k)} + Q_1^{(k)} + U_1^{(k)} + V_1^{(k)}, \quad W_2^{(k)} = P_2^{(k)} + Q_2^{(k)} + U_2^{(k)} + V_2^{(k)},\\
&W_3^{(k)} = P_3^{(k)} + Q_3^{(k)} + U_3^{(k)} + V_3^{(k)}, \quad W_4^{(k)} = P_4^{(k)} + Q_4^{(k)} + U_4^{(k)} + V_4^{(k)}.
\end{aligned} \tag{31}
$$

Thus,

$$
\begin{aligned}
&H - \Theta(Z(k)) = -(A_1\tilde{Z}(k)B_1 + A_2\bar{\tilde{Z}}(k)B_2 + A_3\tilde{Z}^T(k)B_3 + A_4\tilde{Z}^H(k)B_4) = -(P_1^{(k)} + Q_1^{(k)} + U_1^{(k)} + V_1^{(k)}),\\
&\bar{H} - \overline{\Theta(\hat{Z}(k))} = -\overline{(A_1\tilde{\hat{Z}}(k)B_1 + A_2\bar{\tilde{\hat{Z}}}(k)B_2 + A_3\tilde{\hat{Z}}^T(k)B_3 + A_4\tilde{\hat{Z}}^H(k)B_4)} = -(\overline{P_2^{(k)} + Q_2^{(k)} + U_2^{(k)} + V_2^{(k)}}),\\
&H^T - \Theta^T(\check{Z}(k)) = -(A_1\tilde{\check{Z}}(k)B_1 + A_2\bar{\tilde{\check{Z}}}(k)B_2 + A_3\tilde{\check{Z}}^T(k)B_3 + A_4\tilde{\check{Z}}^H(k)B_4)^T = -(P_3^{(k)} + Q_3^{(k)} + U_3^{(k)} + V_3^{(k)})^T,\\
&H^H - \Theta^H(\acute{Z}(k)) = -(A_1\tilde{Z}(k)B_1 + A_2\bar{\tilde{Z}}(k)B_2 + A_3\tilde{Z}(k)^TB_3 + A_4\tilde{Z}(k)^HB_4)^H = -(P_4^{(k)} + Q_4^{(k)} + U_4^{(k)} + V_4^{(k)})^H.
\end{aligned} \tag{32}
$$

Then, it follows from the iteration scheme of the IMGI algorithm and (32) that

$$
\begin{aligned}
&\tilde{Z}^{(1)}(k+1) = \tilde{Z}(k) - \mu D_{11}^H(P_1^{(k)} + Q_1^{(k)} + U_1^{(k)} + V_1^{(k)})D_{12}^H,\\
&\tilde{Z}^{(2)}(k+1) = \tilde{\hat{Z}}(k) - \mu D_{21}^T(\overline{P_2^{(k)} + Q_2^{(k)} + U_2^{(k)} + V_2^{(k)}})D_{22}^T,\\
&\tilde{Z}^{(3)}(k+1) = \tilde{\check{Z}}(k) - \mu\bar{D}_{32}(P_3^{(k)} + Q_3^{(k)} + U_3^{(k)} + V_3^{(k)})^T\bar{D}_{31},\\
&\tilde{Z}^{(4)}(k+1) = \tilde{Z}(k) - \mu D_{42}(P_4^{(k)} + Q_4^{(k)} + U_4^{(k)} + V_4^{(k)})^HD_{41}.
\end{aligned} \tag{33}
$$

By taking the Frobenious norm of each equation in (33) and using the properties of the norm, it holds that

$$
\begin{aligned}
\|\tilde{Z}^{(1)}(k+1)\|^2 &= \|\tilde{Z}(k) - \mu D_{11}^H(P_1^{(k)} + Q_1^{(k)} + U_1^{(k)} + V_1^{(k)})D_{12}^H\|^2\\
&= \|\tilde{Z}(k)\|^2 - \mu\mathrm{tr}(\tilde{Z}^H(k)D_{11}^H(P_1^{(k)} + Q_1^{(k)} + U_1^{(k)} + V_1^{(k)})D_{12}^H)\\
&\quad - \mu\mathrm{tr}(D_{12}(P_1^{(k)} + Q_1^{(k)} + U_1^{(k)} + V_1^{(k)})^HD_{11}\tilde{Z}(k)) + \mu^2\|D_{11}^H(P_1^{(k)} + Q_1^{(k)} + U_1^{(k)} + V_1^{(k)})D_{12}^H\|^2\\
&\le \|\tilde{Z}(k)\|^2 - 2\mu\mathrm{Re}\,\mathrm{tr}[D_{12}^H\tilde{Z}^H(k)D_{11}^H(P_1^{(k)} + Q_1^{(k)} + U_1^{(k)} + V_1^{(k)})]\\
&\quad + \mu^2\|D_{11}\|_2^2\|D_{12}\|_2^2\|P_1^{(k)} + Q_1^{(k)} + U_1^{(k)} + V_1^{(k)}\|^2,
\end{aligned} \tag{34}
$$

$$
\begin{aligned}
\|\tilde{Z}^{(2)}(k+1)\|^2 &= \|\tilde{\hat{Z}}(k) - \mu D_{21}^T(\overline{P_2^{(k)} + Q_2^{(k)} + U_2^{(k)} + V_2^{(k)}})D_{22}^T\|^2\\
&= \|\tilde{\hat{Z}}(k)\|^2 - \mu\mathrm{tr}(\tilde{\hat{Z}}^H(k)D_{21}^T(\overline{P_2^{(k)} + Q_2^{(k)} + U_2^{(k)} + V_2^{(k)}})D_{22}^T)\\
&\quad - \mu\mathrm{tr}(\bar{D}_{22}(P_2^{(k)} + Q_2^{(k)} + U_2^{(k)} + V_2^{(k)})^T\bar{D}_{21}\tilde{\hat{Z}}(k)) + \mu^2\|D_{21}^T(\overline{P_2^{(k)} + Q_2^{(k)} + U_2^{(k)} + V_2^{(k)}})D_{22}^T\|^2\\
&\le \|\tilde{\hat{Z}}(k)\|^2 - 2\mu\mathrm{Re}\,\mathrm{tr}[D_{22}^T\tilde{\hat{Z}}^H(k)D_{21}^T(\overline{P_2^{(k)} + Q_2^{(k)} + U_2^{(k)} + V_2^{(k)}})]\\
&\quad + \mu^2\|D_{21}\|_2^2\|D_{22}\|_2^2\|P_2^{(k)} + Q_2^{(k)} + U_2^{(k)} + V_2^{(k)}\|^2,
\end{aligned} \tag{35}
$$

$$
\begin{aligned}
\|\tilde{Z}^{(3)}(k+1)\|^2 &= \|\tilde{\check{Z}}(k) - \mu\bar{D}_{32}(P_3^{(k)} + Q_3^{(k)} + U_3^{(k)} + V_3^{(k)})^T\bar{D}_{31}\|^2\\
&= \|\tilde{\check{Z}}(k)\|^2 - \mu\mathrm{tr}(\tilde{\check{Z}}^H(k)\bar{D}_{32}(P_3^{(k)} + Q_3^{(k)} + U_3^{(k)} + V_3^{(k)})^T\bar{D}_{31})\\
&\quad - \mu\mathrm{tr}(D_{31}^T(\overline{P_3^{(k)} + Q_3^{(k)} + U_3^{(k)} + V_3^{(k)}})D_{32}^T\tilde{\check{Z}}(k)) + \mu^2\|\bar{D}_{32}(P_3^{(k)} + Q_3^{(k)} + U_3^{(k)} + V_3^{(k)})^T\bar{D}_{31}\|^2\\
&\le \|\tilde{\check{Z}}(k)\|^2 - 2\mu\mathrm{Re}\,\mathrm{tr}[\bar{D}_{31}\tilde{\check{Z}}^H(k)\bar{D}_{32}(P_3^{(k)} + Q_3^{(k)} + U_3^{(k)} + V_3^{(k)})^T] + \mu^2\|D_{31}\|_2^2\|D_{32}\|_2^2\|P_3^{(k)} + Q_3^{(k)}\\
&\quad + U_3^{(k)} + V_3^{(k)}\|^2,
\end{aligned} \tag{36}
$$

and

$$
\begin{aligned}
\|\tilde{Z}^{(4)}(k+1)\|^2 &= \|\tilde{\tilde{Z}}(k) - \mu D_{42}(P_4^{(k)} + Q_4^{(k)} + U_4^{(k)} + V_4^{(k)})^H D_{41}\|^2 \\
&= \|\tilde{\tilde{Z}}(k)\|^2 - \mu \mathrm{tr}(\tilde{\tilde{Z}}^H(k) D_{42}(P_4^{(k)} + Q_4^{(k)} + U_4^{(k)} + V_4^{(k)})^H D_{41}) \\
&\quad - \mu \mathrm{tr}(D_{41}^H(P_4^{(k)} + Q_4^{(k)} + U_4^{(k)} + V_4^{(k)}) D_{42}^H \tilde{\tilde{Z}}(k)) + \mu^2 \|D_{42}(P_4^{(k)} + Q_4^{(k)} + U_4^{(k)} + V_4^{(k)})^H D_{41}\|^2 \\
&\le \|\tilde{\tilde{Z}}(k)\|^2 - 2\mu \mathrm{Re}\,\mathrm{tr}[D_{41}\tilde{\tilde{Z}}^H(k) D_{42}(P_4^{(k)} + Q_4^{(k)} + U_4^{(k)} + V_4^{(k)})^H] \\
&\quad + \mu^2 \|D_{41}\|_2^2 \|D_{42}\|_2^2 \|P_4^{(k)} + Q_4^{(k)} + U_4^{(k)} + V_4^{(k)}\|^2.
\end{aligned} \tag{37}
$$

In view of (34)–(37) and Cauchy inequality, it has

$$
\begin{aligned}
\|\tilde{Z}(k+1)\|^2 &= \left\| \frac{1}{4}\tilde{Z}^{(1)}(k+1) + \frac{1}{4}\tilde{Z}^{(2)}(k+1) + \frac{1}{4}\tilde{Z}^{(3)}(k+1) + \frac{1}{4}\tilde{Z}^{(4)}(k+1) \right\|^2 \\
&\le \left(\left\| \frac{1}{4}\tilde{Z}^{(1)}(k+1) \right\| + \left\| \frac{1}{4}\tilde{Z}^{(2)}(k+1) \right\| + \left\| \frac{1}{4}\tilde{Z}^{(3)}(k+1) \right\| + \left\| \frac{1}{4}\tilde{Z}^{(4)}(k+1) \right\| \right)^2 \\
&\le 4\left(\left\| \frac{1}{4}\tilde{Z}^{(1)}(k+1) \right\|^2 + \left\| \frac{1}{4}\tilde{Z}^{(2)}(k+1) \right\|^2 + \left\| \frac{1}{4}\tilde{Z}^{(3)}(k+1) \right\|^2 + \left\| \frac{1}{4}\tilde{Z}^{(4)}(k+1) \right\|^2 \right) \\
&= \frac{1}{4}\|\tilde{Z}^{(1)}(k+1)\|^2 + \frac{1}{4}\|\tilde{Z}^{(2)}(k+1)\|^2 + \frac{1}{4}\|\tilde{Z}^{(3)}(k+1)\|^2 + \frac{1}{4}\|\tilde{Z}^{(4)}(k+1)\|^2 \\
&\le \frac{1}{4}\|\tilde{Z}(k)\|^2 - \frac{\mu}{2}\mathrm{Re}\,\mathrm{tr}[D_{12}^H \tilde{Z}^H(k) D_{11}^H (P_1^{(k)} + Q_1^{(k)} + U_1^{(k)} + V_1^{(k)})] + \frac{\mu^2}{4}\|D_{11}\|_2^2 \|D_{12}\|_2^2 \|P_1^{(k)} + Q_1^{(k)} + U_1^{(k)} \\
&\quad + V_1^{(k)}\|^2 + \frac{1}{4}\|\tilde{\tilde{Z}}(k)\|^2 - \frac{\mu}{2}\mathrm{Re}\,\mathrm{tr}[D_{22}^T \tilde{\tilde{Z}}^H(k) D_{21}^T \overline{(P_2^{(k)} + Q_2^{(k)} + U_2^{(k)} + V_2^{(k)})}] + \frac{\mu^2}{4}\|D_{21}\|_2^2 \|D_{22}\|_2^2 \\
&\quad \times \|P_2^{(k)} + Q_2^{(k)} + U_2^{(k)} + V_2^{(k)}\|^2 + \frac{1}{4}\|\tilde{\tilde{Z}}(k)\|^2 - \frac{\mu}{2}\mathrm{Re}\,\mathrm{tr}[\bar{D}_{31}\tilde{\tilde{Z}}^H(k)\bar{D}_{32}(P_3^{(k)} + Q_3^{(k)} + U_3^{(k)} + V_3^{(k)})^T] \\
&\quad + \frac{\mu^2}{4}\|D_{31}\|_2^2 \|D_{32}\|_2^2 \|P_3^{(k)} + Q_3^{(k)} + U_3^{(k)} + V_3^{(k)}\|^2 + \frac{1}{4}\|\tilde{\tilde{Z}}(k)\|^2 \\
&\quad - \frac{\mu}{2}\mathrm{Re}\,\mathrm{tr}[D_{41}\tilde{\tilde{Z}}^H(k) D_{42}(P_4^{(k)} + Q_4^{(k)} + U_4^{(k)} + V_4^{(k)})^H] \\
&\quad + \frac{\mu^2}{4}\|D_{41}\|_2^2 \|D_{42}\|_2^2 \|P_4^{(k)} + Q_4^{(k)} + U_4^{(k)} + V_4^{(k)}\|^2 \\
&\le \frac{1}{4}\|\tilde{Z}(k)\|^2 - \frac{\mu}{2}\|P_1^{(k)} + Q_1^{(k)} + U_1^{(k)} + V_1^{(k)}\|^2 + \frac{\mu^2}{4}\|D_{11}\|_2^2 \|D_{12}\|_2^2 \|P_1^{(k)} + Q_1^{(k)} + U_1^{(k)} + V_1^{(k)}\|^2 \\
&\quad + \frac{1}{4}\|\tilde{\tilde{Z}}(k)\|^2 - \frac{\mu}{2}\|P_2^{(k)} + Q_2^{(k)} + U_2^{(k)} + V_2^{(k)}\|^2 + \frac{\mu^2}{4}\|D_{21}\|_2^2 \|D_{22}\|_2^2 \|P_2^{(k)} + Q_2^{(k)} + U_2^{(k)} + V_2^{(k)}\|^2 \\
&\quad + \frac{1}{4}\|\tilde{\tilde{Z}}(k)\|^2 - \frac{\mu}{2}\|P_3^{(k)} + Q_3^{(k)} + U_3^{(k)} + V_3^{(k)}\|^2 + \frac{\mu^2}{4}\|D_{31}\|_2^2 \|D_{32}\|_2^2 \|P_3^{(k)} + Q_3^{(k)} + U_3^{(k)} + V_3^{(k)}\|^2 \\
&\quad + \frac{1}{4}\|\tilde{\tilde{Z}}(k)\|^2 - \frac{\mu}{2}\|P_4^{(k)} + Q_4^{(k)} + U_4^{(k)} + V_4^{(k)}\|^2 + \frac{\mu^2}{4}\|D_{41}\|_2^2 \|D_{42}\|_2^2 \|P_4^{(k)} + Q_4^{(k)} + U_4^{(k)} + V_4^{(k)}\|^2 \\
&= \frac{1}{4}\|\tilde{Z}(k)\|^2 - \frac{\mu}{4}(2 - \mu\|D_{11}\|_2^2 \|D_{12}\|_2^2)\|W_1^{(k)}\|^2 + \frac{1}{4}\|\tilde{\tilde{Z}}(k)\|^2 - \frac{\mu}{4}(2 - \mu\|D_{21}\|_2^2 \|D_{22}\|_2^2)\|W_2^{(k)}\|^2 \\
&\quad + \frac{1}{4}\|\tilde{\tilde{Z}}(k)\|^2 - \frac{\mu}{4}(2 - \mu\|D_{31}\|_2^2 \|D_{32}\|_2^2)\|W_3^{(k)}\|^2 + \frac{1}{4}\|\tilde{\tilde{Z}}(k)\|^2 - \frac{\mu}{4}(2 - \mu\|D_{41}\|_2^2 \|D_{42}\|_2^2)\|W_4^{(k)}\|^2 \\
&\le \|\tilde{Z}(k)\|^2 - \frac{\mu}{4}(2 - \mu\|D_{11}\|_2^2 \|D_{12}\|_2^2)\|W_1^{(k)}\|^2 + \frac{1}{4}\|\tilde{\tilde{Z}}(k)\|^2 - \frac{\mu}{4}(2 - \mu\|D_{21}\|_2^2 \|D_{22}\|_2^2)\|W_2^{(k)}\|^2 \\
&\quad + \frac{1}{4}\|\tilde{\tilde{Z}}(k)\|^2 - \frac{\mu}{4}(2 - \mu\|D_{31}\|_2^2 \|D_{32}\|_2^2)\|W_3^{(k)}\|^2 + \frac{1}{4}\|\tilde{\tilde{Z}}(k)\|^2 - \frac{\mu}{4}(2 - \mu\|D_{41}\|_2^2 \|D_{42}\|_2^2)\|W_4^{(k)}\|^2 \\
&\le \|\tilde{Z}(0)\|^2 - \frac{\mu}{4}(2 - \mu\|D_{11}\|_2^2 \|D_{12}\|_2^2)\sum_{i=0}^{k}\|W_1^{(i)}\|^2 + \frac{1}{4}\sum_{i=0}^{k}\|\tilde{\tilde{Z}}(i)\|^2 - \frac{\mu}{4}(2 - \mu\|D_{21}\|_2^2 \|D_{22}\|_2^2)\sum_{i=0}^{k}\|W_2^{(i)}\|^2 \\
&\quad + \frac{1}{4}\sum_{i=0}^{k}\|\tilde{\tilde{Z}}(i)\|^2 - \frac{\mu}{4}(2 - \mu\|D_{31}\|_2^2 \|D_{32}\|_2^2)\sum_{i=0}^{k}\|W_3^{(i)}\|^2
\end{aligned} \tag{38}
$$

$$+\frac{1}{4}\sum_{i=0}^{k}\|\tilde{Z}(i)\|^2-\frac{\mu}{4}(2-\mu\|D_{41}\|_2^2\|D_{42}\|_2^2)\sum_{i=0}^{k}\|W_4^{(i)}\|^2.$$

By the analogous methods in [19,20,24], we can obtain

$$\frac{1}{4}\sum_{i=0}^{k}\|\tilde{\tilde{Z}}(i)\|^2<+\infty,\quad \frac{1}{4}\sum_{i=0}^{k}\|\tilde{\tilde{Z}}(i)\|^2<+\infty,\quad \frac{1}{4}\sum_{i=0}^{k}\|\tilde{\tilde{Z}}(i)\|^2<+\infty. \tag{39}$$

Thus, if the parameter μ satisfies the following condition

$$0<\mu<\min\left\{\frac{2}{\|D_{11}\|_2^2\|D_{12}\|_2^2},\frac{2}{\|D_{21}\|_2^2\|D_{22}\|_2^2},\frac{2}{\|D_{31}\|_2^2\|D_{32}\|_2^2},\frac{2}{\|D_{41}\|_2^2\|D_{42}\|_2^2}\right\}, \tag{40}$$

then (38) implies that $\sum_{i=0}^{\infty}\|W_1^{(i)}\|^2<+\infty$, which results in $\lim_{i\to+\infty}\|W_1^{(i)}\|=0$, that is,

$$\lim_{i\to+\infty}W_1^{(i)}=\lim_{i\to+\infty}A_1\tilde{Z}(i)B_1+A_2\bar{\tilde{Z}}(i)B_2+A_3\tilde{Z}^T(i)B_3+A_4\tilde{Z}^H(i)B_4=0. \tag{41}$$

Having in mind that the CCT Sylvester matrix equation (2) has a unique solution, then it follows from (41) and Lemma 3.2 that $\lim_{i\to+\infty}\tilde{Z}(i)=0$, i.e., $\lim_{i\to+\infty}Z(i)=Z^*$. The proof is completed. □

In [17], Niu et al. introduced a relaxation parameter into the GI algorithm and established the RGI algorithm. And the numerical results of [17] illustrated that the RGI algorithm outperforms the GI one with proper relaxation parameter. Then it motivates many researchers to derive the relaxed versions of some existing algorithms for several kinds of matrix equations, see [15,16,19] for details. Recently, Wang et al. [21] constructed a RGI algorithm for the CCT Sylvester matrix equation (2). The framework of the RGI algorithm is as follows.

The RGI algorithm [21]:

Step 1: Given matrices A_i, B_i, $H\in\mathbb{C}^{n\times n}$ $(i=1,2,3,4)$, two constants $\varepsilon,\mu>0$ and a relaxed factor $0<\gamma<1$. Choose the initial matrices $Z(0)$ and $Z_i(0)(i=1,2,3,4)$, and set $m=0$;

Step 2: If $\delta_m=\frac{\|H-A_1Z(m)B_1-A_2\bar{Z}(m)B_2-A_3Z(m)^TB_3-A_4Z(m)^HB_4\|}{\|H\|}<\varepsilon$, stop; otherwise, go to Step 3;

Step 3: Update the sequences

$$\begin{aligned}
Z^{(1)}(m+1)&=Z^{(1)}(m)+\frac{1}{2}\gamma\mu A_1^H(H-\Theta(Z(m)))B_1^H,\\
Z^{(2)}(m+1)&=Z^{(2)}(m)+\frac{1}{2}\gamma\mu A_2^T(\bar{H}-\overline{\Theta(Z(m))})B_2^T,\\
Z^{(3)}(m+1)&=Z^{(3)}(m)+\frac{1-\gamma}{2}\mu\overline{B_3}(H^T-\Theta^T(Z(m)))\overline{A_3},\\
Z^{(4)}(m+1)&=Z^{(4)}(m)+\frac{1-\gamma}{2}\mu B_4(H^H-\Theta^H(Z(m)))A_4,\\
Z(m+1)&=\frac{1}{2}(1-\gamma)Z^{(1)}(m+1)+\frac{1}{2}(1-\gamma)Z^{(2)}(m+1)+\frac{1}{2}\gamma Z^{(3)}(m+1)+\frac{1}{2}\gamma Z^{(4)}(m+1);
\end{aligned}$$

Step 4: Set $m:=m+1$ and return to Step 2.

Numerical results of [21] validated that the RGI algorithm with proper ω is faster than the GI one in [14]. To further improve the computational efficiency of the IMGI algorithm, by making use of Lemma 3.1, and enlightened by the idea of [17,21], we propose the following IMRGI algorithm.

The IMRGI algorithm:

Step 1: Given matrices A_i, B_i, $H\in\mathbb{C}^{n\times n}$ $(i=1,2,3,4)$, two constants $\varepsilon,\mu>0$ and a relaxed factor $0<\omega<1$. Choose the initial matrices $Z(0)$ and $Z_i(0)(i=1,2,3,4)$, and set $k=0$;

Step 2: If $\delta_k=\frac{\|H-A_1Z(k)B_1-A_2\bar{Z}(k)B_2-A_3Z(k)^TB_3-A_4Z(k)^HB_4\|}{\|H\|}<\varepsilon$, stop; otherwise, go to Step 3;

Step 3: Update the sequences

$$
\begin{aligned}
Z^{(1)}(k+1) &= Z(k) + \frac{1}{2}\omega\mu D_{11}^H(H - \Theta(Z(k)))D_{12}^H,\\
\hat{Z}(k) &= \frac{1}{2}(1-\omega)Z^{(1)}(k+1) + \frac{1}{2}(1-\omega)Z^{(2)}(k) + \frac{1}{2}\omega Z^{(3)}(k) + \frac{1}{2}\omega Z^{(4)}(k),\\
Z^{(2)}(k+1) &= \hat{Z}(k) + \frac{1}{2}\omega\mu D_{21}^T(\bar{H} - \overline{\Theta(\hat{Z}(k))})D_{22}^T,\\
\check{Z}(k) &= \frac{1}{2}(1-\omega)Z^{(1)}(k+1) + \frac{1}{2}(1-\omega)Z^{(2)}(k+1) + \frac{1}{2}\omega Z^{(3)}(k) + \frac{1}{2}\omega Z^{(4)}(k),\\
Z^{(3)}(k+1) &= \check{Z}(k) + \frac{1}{2}(1-\omega)\mu\overline{D_{32}}(H^T - \Theta^T(\check{Z}(k)))\overline{D_{31}},\\
\acute{Z}(k) &= \frac{1}{2}(1-\omega)Z^{(1)}(k+1) + \frac{1}{2}(1-\omega)Z^{(2)}(k+1) + \frac{1}{2}\omega Z^{(3)}(k+1) + \frac{1}{2}\omega Z^{(4)}(k),\\
Z^{(4)}(k+1) &= \acute{Z}(k) + \frac{1}{2}(1-\omega)\mu D_{42}(H^H - \Theta^H(\acute{Z}(k)))D_{41},\\
Z(k+1) &= \frac{1}{2}(1-\omega)Z^{(1)}(k+1) + \frac{1}{2}(1-\omega)Z^{(2)}(k+1) + \frac{1}{2}\omega Z^{(3)}(k+1) + \frac{1}{2}\omega Z^{(4)}(k+1);
\end{aligned}
$$

Step 4: Set $k \coloneqq k+1$ and return to Step 2.

In what follows, we establish the convergence theorem of the IMRGI algorithm.

Theorem 3.4. *Assume that the CCT Sylvester matrix equation* (2) *has a unique solution Z^*. Then the IMRGI algorithm is convergent if and only if the parameters μ and ω are chosen to satisfy $\rho(UV) < 1$, where*

$$
U = \begin{pmatrix}
I & 0 & 0 & 0\\
\frac{1-\omega}{2}G_2 & I & 0 & 0\\
\frac{(1-\omega)^2}{4}G_3G_2 + \frac{1-\omega}{2}G_3 & \frac{1-\omega}{2}G_3 & I & 0\\
\frac{1-\omega}{2}G_4 + \frac{(1-\omega)^2}{4}G_4G_2 + \frac{\omega(1-\omega)}{4}G_4G_3 + \frac{\omega(1-\omega)^2}{8}G_4G_3G_2 & \frac{(1-\omega)^2}{4}G_4G_3 + \frac{1-\omega}{2}G_4 & \frac{\omega}{2}G_4 & I
\end{pmatrix},
\tag{42}
$$

$$
V = \begin{pmatrix}
\frac{1}{2}(1-\omega)G_1 & \frac{1}{2}(1-\omega)G_1 & \frac{1}{2}\omega G_1 & \frac{1}{2}\omega G_1\\
0 & \frac{1}{2}(1-\omega)G_2 & \frac{1}{2}\omega G_2 & \frac{1}{2}\omega G_2\\
0 & 0 & \frac{1}{2}\omega G_3 & \frac{1}{2}\omega G_3\\
0 & 0 & 0 & \frac{1}{2}\omega G_4
\end{pmatrix},
$$

with

$$
\begin{aligned}
&G_1 = I_{4n^2} - \frac{1}{2}\omega\mu W_1, \quad G_2 = I_{4n^2} - \frac{1}{2}\omega\mu W_2,\\
&G_3 = I_{4n^2} - \frac{1}{2}(1-\omega)\mu W_3, \quad G_4 = I_{4n^2} - \frac{1}{2}(1-\omega)\mu W_4,\\
&W_1 = (B_1D_{12}^H)_\sigma^T P_n \otimes (D_{11}^H A_1)_\sigma P_n + (B_2D_{12}^H)_\sigma^T \otimes (D_{11}^H A_2)_\sigma + [(B_3D_{12}^H)_\sigma^T P_n \otimes (D_{11}^H A_3)_\sigma P_n + (B_4D_{12}^H)_\sigma^T\\
&\qquad \otimes (D_{11}^H A_4)_\sigma]P(2n, 2n),\\
&W_2 = (\bar{B}_1D_{22}^T)_\sigma^T \otimes (D_{21}^T\bar{A}_1)_\sigma + [(\bar{B}_3D_{22}^T)_\sigma^T \otimes (D_{21}^T\bar{A}_3)_\sigma + (\bar{B}_4D_{22}^T)_\sigma^T P_n \otimes (D_{21}^T\bar{A}_4)_\sigma P_n]P(2n, 2n) + (\bar{B}_2D_{22}^T)_\sigma^T P_n\\
&\qquad \otimes (D_{21}^T\bar{A}_2)_\sigma P_n,
\end{aligned}
$$

$$W_3 = [(A_1^T \bar{D}_{31})_\sigma^T P_n \otimes (\bar{D}_{32} B_1^T)_\sigma P_n + (A_2^T \bar{D}_{31})_\sigma^T \otimes (\bar{D}_{32} B_2^T)_\sigma] P(2n, 2n) + (A_3^T \bar{D}_{31})_\sigma^T P_n \otimes (\bar{D}_{32} B_3^T)_\sigma P_n + (A_4^T \bar{D}_{31})_\sigma^T \otimes (\bar{D}_{32} B_4^T)_\sigma,$$

$$W_4 = [(A_1^H D_{41})_\sigma^T \otimes (D_{42} B_1^H)_\sigma + (A_2^H D_{41})_\sigma^T P_n \otimes (D_{42} B_2^H)_\sigma P_n] P(2n, 2n) + (A_3^H D_{41})_\sigma^T \otimes (D_{42} B_3^H)_\sigma + (A_4^H D_{41})_\sigma^T P_n \otimes (D_{42} B_4^H)_\sigma P_n.$$

Proof. The proof of this theorem is similar to that of Theorem 3.1, and hence, we omit it here. □

Theorem 3.4 gives the convergence condition of the IMRGI algorithm, but it does not propose the intervals of the parameters ω, μ, and their optimal expressions. The reason is that the matrices G_1, G_2, G_3, and G_4 in the matrices U and V of Theorem 3.4 contain the parameters ω, μ, and they cannot be separated from the matrices U and V. Hence, it is difficult to obtain the expression of the iteration matrix of the IMRGI algorithm with respect to ω, μ, and the optimal values of the parameters ω, μ. Nevertheless, we can derive the intervals of the parameters ω, μ contained in the IMRGI algorithm and the quasi-optimal expression of ω under some suitable conditions.

Theorem 3.5. *Let the conditions of* Theorem 3.4 *be satisfied, and U and V defined in* Theorem 3.4 *be normal matrices. Denote by $\lambda_i^{(1)}$, $\lambda_i^{(2)}$, $\lambda_i^{(3)}$, and $\lambda_i^{(4)}$ $(i = 1, 2, \ldots, 4n^2)$ the eigenvalues of the matrices W_1, W_2, W_3, and W_4, respectively, and $\mathrm{Re}(\lambda_i^{(j)}) > 0$ for $i = 1, 2, \ldots, 4n^2$; $j = 1, 2, 3, 4$. Then the IMRGI algorithm is convergent if $0 < \omega < 1$ and*

$$0 < \mu < \min_{1 \le i \le 4n^2} \left\{ \frac{4\mathrm{Re}(\lambda_i^{(1)})}{\omega(\mathrm{Re}(\lambda_i^{(1)})^2 + \mathrm{Im}(\lambda_i^{(1)})^2)}, \frac{4\mathrm{Re}(\lambda_i^{(2)})}{\omega(\mathrm{Re}(\lambda_i^{(2)})^2 + \mathrm{Im}(\lambda_i^{(2)})^2)}, \frac{4\mathrm{Re}(\lambda_i^{(3)})}{(1-\omega)(\mathrm{Re}(\lambda_i^{(3)})^2 + \mathrm{Im}(\lambda_i^{(3)})^2)}, \frac{4\mathrm{Re}(\lambda_i^{(4)})}{(1-\omega)(\mathrm{Re}(\lambda_i^{(4)})^2 + \mathrm{Im}(\lambda_i^{(4)})^2)} \right\}. \tag{43}$$

Besides, if all the eigenvalues of the matrices W_1, W_2, W_3, and W_4 are real, then $\omega^* = \frac{4}{\mu(\bar{\lambda}_{\max} + \bar{\lambda}_{\min})} := \omega_1^*$ or $\omega^* = 1 - \frac{4}{\mu(\tilde{\lambda}_{\max} + \tilde{\lambda}_{\min})} := \omega_2^*$. If $\Phi(\omega_1^*) < \Phi(\omega_2^*)$, then $\omega^* = \omega_1^*$. Otherwise, $\omega^* = \omega_2^*$. Here, $\bar{\lambda}_{\max}$ and $\bar{\lambda}_{\min}$ denote the maximum and minimum eigenvalues of two matrices W_1, W_2, respectively, and $\tilde{\lambda}_{\max}$ and $\tilde{\lambda}_{\min}$ denote the maximum and minimum eigenvalues of two matrices W_3, W_4, respectively.

Proof. Inasmuch as U and V in Theorem 3.4 are normal matrices, according to Lemma 2.5, it has

$$\begin{aligned}
\rho(UV) &\le \|UV\|_2 \le \|U\|_2 \|V\|_2 = \rho(U)\rho(V) = \rho(V) \\
&= \max\left\{\rho\left(\frac{1}{2}(1-\omega)G_1\right), \rho\left(\frac{1}{2}(1-\omega)G_2\right), \rho\left(\frac{1}{2}\omega G_3\right), \rho\left(\frac{1}{2}\omega G_4\right)\right\} \\
&= \max_{1 \le i \le 4n^2}\left\{\left|\frac{1-\omega}{2}\left(1 - \frac{1}{2}\omega\mu\lambda_i^{(1)}\right)\right|, \left|\frac{1-\omega}{2}\left(1 - \frac{1}{2}\omega\mu\lambda_i^{(2)}\right)\right|, \left|\frac{\omega}{2}\left(1 - \frac{1-\omega}{2}\mu\lambda_i^{(3)}\right)\right|, \left|\frac{\omega}{2}\left(1 - \frac{1-\omega}{2}\mu\lambda_i^{(4)}\right)\right|\right\} \\
&\le \max_{1 \le i \le 4n^2}\left\{\left|1 - \frac{1}{2}\omega\mu\lambda_i^{(1)}\right|, \left|1 - \frac{1}{2}\omega\mu\lambda_i^{(2)}\right|, \left|1 - \frac{1-\omega}{2}\mu\lambda_i^{(3)}\right|, \left|1 - \frac{1-\omega}{2}\mu\lambda_i^{(4)}\right|\right\} \\
&= \max\left\{\max_{1 \le i \le 4n^2}\left\{\left|1 - \frac{1}{2}\omega\mu\lambda_i^{(1)}\right|, \left|1 - \frac{1}{2}\omega\mu\lambda_i^{(2)}\right|\right\}, \max_{1 \le i \le 4n^2}\left\{\left|1 - \frac{1-\omega}{2}\mu\lambda_i^{(3)}\right|, \left|1 - \frac{1-\omega}{2}\mu\lambda_i^{(4)}\right|\right\}\right\},
\end{aligned} \tag{44}$$

in view of $0 < \omega < 1$, $\frac{\omega}{2} < 1$ and $\frac{1-\omega}{2} < 1$. This means that $\rho(UV) < 1$ if

$$\left|1 - \frac{1}{2}\omega\mu\lambda_i^{(1)}\right| < 1, \quad \left|1 - \frac{1}{2}\omega\mu\lambda_i^{(2)}\right| < 1, \quad \left|1 - \frac{1-\omega}{2}\mu\lambda_i^{(3)}\right| < 1, \quad \left|1 - \frac{1-\omega}{2}\mu\lambda_i^{(4)}\right| < 1, \quad i = 1, 2, \ldots, 4n^2,$$

which can be equivalently transformed into the following inequalities

$$\left(1 - \frac{1}{2}\omega\mu\mathrm{Re}(\lambda_i^{(1)})\right)^2 + \frac{1}{4}\omega^2\mu^2\mathrm{Im}(\lambda_i^{(1)})^2 < 1, \quad \left(1 - \frac{1}{2}\omega\mu\mathrm{Re}(\lambda_i^{(2)})\right)^2 + \frac{1}{4}\omega^2\mu^2\mathrm{Im}(\lambda_i^{(2)})^2 < 1,$$

$$\left(1-\frac{1-\omega}{2}\mu\mathrm{Re}(\lambda_i^{(3)})\right)^2+\frac{(1-\omega)^2}{4}\mu^2\mathrm{Im}(\lambda_i^{(3)})^2<1,\quad \left(1-\frac{1-\omega}{2}\mu\mathrm{Re}(\lambda_i^{(4)})\right)^2+\frac{(1-\omega)^2}{4}\mu^2\mathrm{Im}(\lambda_i^{(4)})^2<1,$$

$i=1,2,\ldots,4n^2$.

By solving the above inequalities, it has

$$\begin{aligned}
&0<\mu<\frac{4\mathrm{Re}(\lambda_i^{(1)})}{\omega(\mathrm{Re}(\lambda_i^{(1)})^2+\mathrm{Im}(\lambda_i^{(1)})^2)},\quad 0<\mu<\frac{4\mathrm{Re}(\lambda_i^{(2)})}{\omega(\mathrm{Re}(\lambda_i^{(2)})^2+\mathrm{Im}(\lambda_i^{(2)})^2)},\\
&0<\mu<\frac{4\mathrm{Re}(\lambda_i^{(3)})}{(1-\omega)(\mathrm{Re}(\lambda_i^{(3)})^2+\mathrm{Im}(\lambda_i^{(3)})^2)},\quad 0<\mu<\frac{4\mathrm{Re}(\lambda_i^{(3)})}{(1-\omega)(\mathrm{Re}(\lambda_i^{(4)})^2+\mathrm{Im}(\lambda_i^{(4)})^2)}.
\end{aligned}\tag{45}$$

Then we obtain the convergence condition (43) of the IMRGI algorithm in terms of (45).

Besides, if all the eigenvalues of the matrices W_1, W_2, W_3, and W_4 are real, then it follows from (44) that

$$\begin{aligned}
\rho(UV)&\le\max\left\{\left|1-\frac{1}{2}\omega\mu\bar{\lambda}_{\max}\right|,\left|1-\frac{1}{2}\omega\mu\bar{\lambda}_{\min}\right|,\left|1-\frac{1-\omega}{2}\mu\tilde{\lambda}_{\max}\right|,\left|1-\frac{1-\omega}{2}\mu\tilde{\lambda}_{\min}\right|\right\}\\
&=\max\{\Delta_1(\omega),\Delta_2(\omega)\}:=\Phi(\omega),
\end{aligned}\tag{46}$$

where $\bar{\lambda}_{\max}$ and $\bar{\lambda}_{\min}$ denote the maximum and minimum eigenvalues of two matrices W_1, W_2, respectively, and $\tilde{\lambda}_{\max}$ and $\tilde{\lambda}_{\min}$ denote the maximum and minimum eigenvalues of two matrices W_3, W_4, respectively. And $\Delta_1(\omega)=\max\left\{\left|1-\frac{1}{2}\omega\mu\bar{\lambda}_{\max}\right|,\left|1-\frac{1}{2}\omega\mu\bar{\lambda}_{\min}\right|\right\}$ and $\Delta_2(\omega)=\max\left\{\left|1-\frac{1-\omega}{2}\mu\tilde{\lambda}_{\max}\right|,\left|1-\frac{1-\omega}{2}\mu\tilde{\lambda}_{\min}\right|\right\}$.

To minimize $\Delta_1(\omega)$, the parameter ω should satisfy $1-\frac{1}{2}\omega\mu\bar{\lambda}_{\min}=\frac{1}{2}\omega\mu\bar{\lambda}_{\max}-1$, which leads to $\omega_1^*=\frac{4}{\mu(\bar{\lambda}_{\max}+\bar{\lambda}_{\min})}$. In addition, the parameter ω minimizing $\Delta_2(\omega)$ satisfies $1-\frac{1-\omega}{2}\mu\tilde{\lambda}_{\min}=\frac{1-\omega}{2}\mu\tilde{\lambda}_{\max}-1$, which results in $\omega_2^*=1-\frac{4}{\mu(\tilde{\lambda}_{\max}+\tilde{\lambda}_{\min})}$. Substituting ω_1^* and ω_2^* into $\Phi(\omega)$, if $\Phi(\omega_1^*)<\Phi(\omega_2^*)$, then we take $\omega^*=\omega_1^*$. Otherwise, $\omega^*=\omega_2^*$. □

Remark 3.2. Theorem 3.5 gives the convergence conditions for the parameters μ,ω of the IMRGI algorithm. However, it is involved in the real and imaginary parts of the eigenvalues of the matrices W_i $(i=1,2,3,4)$, and it is not easy to compute it. Moreover, the assumptions that the matrices U, V are normal matrices, and all the eigenvalues of the matrices W_i $(i=1,2,3,4)$ are real are somewhat strong. Besides, (43) is a sufficient condition for the convergence of the IMRGI algorithm, and the assumption is stronger than Theorem 3.1. So we will study a more practical convergence condition for the IMRGI algorithm in the future.

In addition, we deduce another convergence theorem of the IMRGI algorithm as follows.

Theorem 3.6. *Suppose that the CCT Sylvester matrix equation* (2) *has a unique solution* Z^*. *If* $0<\omega<1$ *and* μ *satisfies*

$$0<\mu<\min\left\{\frac{4}{\omega||D_{11}||_2^2||D_{12}||_2^2},\frac{4}{\omega||D_{21}||_2^2||D_{22}||_2^2},\frac{4}{(1-\omega)||D_{31}||_2^2||D_{32}||_2^2},\frac{4}{(1-\omega)||D_{41}||_2^2||D_{42}||_2^2}\right\},\tag{47}$$

or the parameters μ *and* ω *satisfy*

$$\begin{aligned}
&\max\left\{0,1-\min\left\{\frac{4}{\mu||D_{31}||_2^2||D_{32}||_2^2},\frac{4}{\mu||D_{41}||_2^2||D_{42}||_2^2}\right\}\right\}<\omega<\min\left\{\frac{4}{\mu||D_{11}||_2^2||D_{12}||_2^2},\frac{4}{\mu||D_{21}||_2^2||D_{22}||_2^2},1\right\},\\
&\min\left\{\frac{4}{\mu||D_{11}||_2^2||D_{12}||_2^2},\frac{4}{\mu||D_{21}||_2^2||D_{22}||_2^2}\right\}+\min\left\{\frac{4}{\mu||D_{31}||_2^2||D_{32}||_2^2},\frac{4}{\mu||D_{41}||_2^2||D_{42}||_2^2}\right\}>1,
\end{aligned}\tag{48}$$

then the matrix sequence $\{Z(k)\}$ *generated by the IMRGI algorithm converges to* Z^*.

Proof. Again using the notations in (30)–(31), we have

$$\begin{aligned}
&H - \Theta(Z(k)) = -(A_1\tilde{Z}(k)B_1 + A_2\bar{\tilde{Z}}(k)B_2 + A_3\tilde{Z}^T(k)B_3 + A_4\tilde{Z}^H(k)B_4) = -(P_1^{(k)} + Q_1^{(k)} + U_1^{(k)} + V_1^{(k)}),\\
&\bar{H} - \overline{\Theta(\hat{Z}(k))} = -\overline{(A_1\hat{\tilde{Z}}(k)B_1 + A_2\bar{\hat{\tilde{Z}}}(k)B_2 + A_3\hat{\tilde{Z}}^T(k)B_3 + A_4\hat{\tilde{Z}}^H(k)B_4)} = -(\overline{P_2^{(k)} + Q_2^{(k)} + U_2^{(k)} + V_2^{(k)}}),\\
&H^T - \Theta^T(\check{Z}(k)) = -(A_1\check{\tilde{Z}}(k)B_1 + A_2\bar{\check{\tilde{Z}}}(k)B_2 + A_3\check{\tilde{Z}}^T(k)B_3 + A_4\check{\tilde{Z}}^H(k)B_4)^T = -(P_3^{(k)} + Q_3^{(k)} + U_3^{(k)} + V_3^{(k)})^T,\\
&H^H - \Theta^H(\acute{Z}(k)) = -(A_1\acute{\tilde{Z}}(k)B_1 + A_2\bar{\acute{\tilde{Z}}}(k)B_2 + A_3\acute{\tilde{Z}}(k)^TB_3 + A_4\acute{\tilde{Z}}(k)^HB_4)^H = -(P_4^{(k)} + Q_4^{(k)} + U_4^{(k)} + V_4^{(k)})^H.
\end{aligned} \tag{49}$$

Then it follows from the IMRGI algorithm and (49) that

$$\begin{aligned}
\tilde{Z}^{(1)}(k+1) &= \tilde{Z}(k) - \frac{1}{2}\omega\mu D_{11}^H(P_1^{(k)} + Q_1^{(k)} + U_1^{(k)} + V_1^{(k)})D_{12}^H,\\
\tilde{Z}^{(2)}(k+1) &= \bar{\tilde{Z}}(k) - \frac{1}{2}\omega\mu D_{21}^T(\overline{P_2^{(k)} + Q_2^{(k)} + U_2^{(k)} + V_2^{(k)}})D_{22}^T,\\
\tilde{Z}^{(3)}(k+1) &= \check{\tilde{Z}}(k) - \frac{1}{2}(1-\omega)\mu\bar{D}_{32}(P_3^{(k)} + Q_3^{(k)} + U_3^{(k)} + V_3^{(k)})^T\bar{D}_{31},\\
\tilde{Z}^{(4)}(k+1) &= \acute{\tilde{Z}}(k) - \frac{1}{2}(1-\omega)\mu D_{42}(P_4^{(k)} + Q_4^{(k)} + U_4^{(k)} + V_4^{(k)})^HD_{41}.
\end{aligned} \tag{50}$$

Taking the Frobenius norm in (50) yields that

$$\begin{aligned}
\|\tilde{Z}^{(1)}(k+1)\|^2 &= \|\tilde{Z}(k) - \frac{1}{2}\omega\mu D_{11}^H(P_1^{(k)} + Q_1^{(k)} + U_1^{(k)} + V_1^{(k)})D_{12}^H\|^2\\
&= \|\tilde{Z}(k)\|^2 - \frac{1}{2}\omega\mu\mathrm{tr}(\tilde{Z}^H(k)D_{11}^H(P_1^{(k)} + Q_1^{(k)} + U_1^{(k)} + V_1^{(k)})D_{12}^H)\\
&\quad - \frac{1}{2}\omega\mu\mathrm{tr}(D_{12}(P_1^{(k)} + Q_1^{(k)} + U_1^{(k)} + V_1^{(k)})^HD_{11}\tilde{Z}(k)) + \frac{1}{4}\omega^2\mu^2\|D_{11}^H(P_1^{(k)} + Q_1^{(k)} + U_1^{(k)} + V_1^{(k)})D_{12}^H\|^2\\
&\le \|\tilde{Z}(k)\|^2 - \omega\mu\mathrm{Re}\,\mathrm{tr}[D_{12}^H\tilde{Z}^H(k)D_{11}^H(P_1^{(k)} + Q_1^{(k)} + U_1^{(k)} + V_1^{(k)})] + \frac{1}{4}\omega^2\mu^2\|D_{11}\|_2^2\|D_{12}\|_2^2\|P_1^{(k)} + Q_1^{(k)}\\
&\quad + U_1^{(k)} + V_1^{(k)}\|^2\\
&\le \|\tilde{Z}(k)\|^2 - \omega\mu\|P_1^{(k)} + Q_1^{(k)} + U_1^{(k)} + V_1^{(k)}\|^2 + \frac{1}{4}\omega^2\mu^2\|D_{11}\|_2^2\|D_{12}\|_2^2\|P_1^{(k)} + Q_1^{(k)} + U_1^{(k)} + V_1^{(k)}\|^2\\
&= \|\tilde{Z}(k)\|^2 - \omega\mu\left(1 - \frac{1}{4}\omega\mu\|D_{11}\|_2^2\|D_{12}\|_2^2\right)\|W_1^{(k)}\|^2,
\end{aligned} \tag{51}$$

$$\begin{aligned}
\|\tilde{Z}^{(2)}(k+1)\|^2 &= \|\bar{\tilde{Z}}(k) - \frac{1}{2}\omega\mu D_{21}^T(\overline{P_2^{(k)} + Q_2^{(k)} + U_2^{(k)} + V_2^{(k)}})D_{22}^T\|^2\\
&= \|\bar{\tilde{Z}}(k)\|^2 - \frac{1}{2}\omega\mu\mathrm{tr}(\bar{\tilde{Z}}^H(k)D_{21}^T(\overline{P_2^{(k)} + Q_2^{(k)} + U_2^{(k)} + V_2^{(k)}})D_{22}^T)\\
&\quad - \frac{1}{2}\omega\mu\mathrm{tr}(\bar{D}_{22}(P_2^{(k)} + Q_2^{(k)} + U_2^{(k)} + V_2^{(k)})^T\bar{D}_{21}\bar{\tilde{Z}}(k)) + \frac{1}{4}\omega^2\mu^2\|D_{21}^T(\overline{P_2^{(k)} + Q_2^{(k)} + U_2^{(k)} + V_2^{(k)}})D_{22}^T\|^2\\
&\le \|\bar{\tilde{Z}}(k)\|^2 - \omega\mu\mathrm{Re}\,\mathrm{tr}[D_{22}^T\bar{\tilde{Z}}^H(k)D_{21}^T(\overline{P_2^{(k)} + Q_2^{(k)} + U_2^{(k)} + V_2^{(k)}})] + \frac{1}{4}\omega^2\mu^2\|D_{21}\|_2^2\|D_{22}\|_2^2\|P_2^{(k)} + Q_2^{(k)}\\
&\quad + U_2^{(k)} + V_2^{(k)}\|^2\\
&\le \|\bar{\tilde{Z}}(k)\|^2 - \omega\mu\|P_2^{(k)} + Q_2^{(k)} + U_2^{(k)} + V_2^{(k)}\|^2 + \frac{1}{4}\omega^2\mu^2\|D_{21}\|_2^2\|D_{22}\|_2^2\|P_2^{(k)} + Q_2^{(k)} + U_2^{(k)} + V_2^{(k)}\|^2\\
&= \|\bar{\tilde{Z}}(k)\|^2 - \omega\mu\left(1 - \frac{1}{4}\omega\mu\|D_{21}\|_2^2\|D_{22}\|_2^2\right)\|W_2^{(k)}\|^2,
\end{aligned} \tag{52}$$

$$\begin{aligned}
\|\tilde{Z}^{(3)}(k+1)\|^2 &= \|\check{\tilde{Z}}(k) - \frac{1}{2}(1-\omega)\mu\bar{D}_{32}(P_3^{(k)} + Q_3^{(k)} + U_3^{(k)} + V_3^{(k)})^T\bar{D}_{31}\|^2\\
&= \|\check{\tilde{Z}}(k)\|^2 - \frac{1}{2}(1-\omega)\mu\mathrm{tr}(\bar{\tilde{Z}}^H(k)\bar{D}_{32}(P_3^{(k)} + Q_3^{(k)} + U_3^{(k)} + V_3^{(k)})^T\bar{D}_{31})\\
&\quad - \frac{1}{2}(1-\omega)\mu\mathrm{tr}(D_{31}^T(\overline{P_3^{(k)} + Q_3^{(k)} + U_3^{(k)} + V_3^{(k)}})D_{32}^T\check{\tilde{Z}}(k))
\end{aligned} \tag{53}$$

$$
\begin{aligned}
&+\frac{1}{4}(1-\omega)^2\mu^2\|\bar{D}_{32}(P_3^{(k)}+Q_3^{(k)}+U_3^{(k)}+V_3^{(k)})^T\bar{D}_{31}\|^2\\
&\le\|\tilde{\bar{Z}}(k)\|^2-(1-\omega)\mu\mathrm{Retr}[\bar{D}_{31}\tilde{\bar{Z}}^H(k)\bar{D}_{32}(P_3^{(k)}+Q_3^{(k)}+U_3^{(k)}+V_3^{(k)})^T]\\
&+\frac{1}{4}(1-\omega)^2\mu^2\|D_{31}\|_2^2\|D_{32}\|_2^2\|P_3^{(k)}+Q_3^{(k)}+U_3^{(k)}+V_3^{(k)}\|^2\\
&\le\|\tilde{\bar{Z}}(k)\|^2-(1-\omega)\mu\|P_3^{(k)}+Q_3^{(k)}+U_3^{(k)}+V_3^{(k)}\|^2\\
&+\frac{1}{4}(1-\omega)^2\mu^2\|D_{31}\|_2^2\|D_{32}\|_2^2\|P_3^{(k)}+Q_3^{(k)}+U_3^{(k)}+V_3^{(k)}\|^2\\
&=\|\tilde{\bar{Z}}(k)\|^2-(1-\omega)\mu\left(1-\frac{1}{4}(1-\omega)\mu\|D_{31}\|_2^2\|D_{32}\|_2^2\right)\|W_3^{(k)}\|^2,
\end{aligned}
$$

and

$$
\begin{aligned}
\|\tilde{Z}^{(4)}(k+1)\|^2&=\|\tilde{\bar{Z}}(k)-\frac{1}{2}(1-\omega)\mu D_{42}(P_4^{(k)}+Q_4^{(k)}+U_4^{(k)}+V_4^{(k)})^HD_{41}\|^2\\
&=\|\tilde{\bar{Z}}(k)\|^2-\frac{1}{2}(1-\omega)\mu\mathrm{tr}(\tilde{\bar{Z}}^H(k)D_{42}(P_4^{(k)}+Q_4^{(k)}+U_4^{(k)}+V_4^{(k)})^HD_{41})\\
&\quad-\frac{1}{2}(1-\omega)\mu\mathrm{tr}(D_{41}^H(P_4^{(k)}+Q_4^{(k)}+U_4^{(k)}+V_4^{(k)})D_{42}^H\tilde{\bar{Z}}(k))\\
&\quad+\frac{1}{4}(1-\omega)^2\mu^2\|D_{42}(P_4^{(k)}+Q_4^{(k)}+U_4^{(k)}+V_4^{(k)})^HD_{41}\|^2\\
&\le\|\tilde{\bar{Z}}(k)\|^2-(1-\omega)\mu\mathrm{Retr}[D_{41}\tilde{\bar{Z}}^H(k)D_{42}(P_4^{(k)}+Q_4^{(k)}+U_4^{(k)}+V_4^{(k)})^H]\\
&\quad+\frac{1}{4}(1-\omega)^2\mu^2\|D_{41}\|_2^2\|D_{42}\|_2^2\|P_4^{(k)}+Q_4^{(k)}+U_4^{(k)}+V_4^{(k)}\|^2\\
&\le\|\tilde{\bar{Z}}(k)\|^2-(1-\omega)\mu\|P_4^{(k)}+Q_4^{(k)}+U_4^{(k)}+V_4^{(k)}\|^2\\
&\quad+\frac{1}{4}(1-\omega)^2\mu^2\|D_{41}\|_2^2\|D_{42}\|_2^2\|P_4^{(k)}+Q_4^{(k)}+U_4^{(k)}+V_4^{(k)}\|^2\\
&=\|\tilde{\bar{Z}}(k)\|^2-(1-\omega)\mu\left(1-\frac{1}{4}(1-\omega)\mu\|D_{41}\|_2^2\|D_{42}\|_2^2\right)\|W_4^{(k)}\|^2.
\end{aligned}
\tag{54}
$$

By making use of Cauchy-Schwarz inequality and (51)–(54), we derive

$$
\begin{aligned}
\|\tilde{Z}(k+1)\|^2&=\left\|\frac{1}{2}(1-\omega)\tilde{Z}^{(1)}(k+1)+\frac{1}{2}(1-\omega)\tilde{Z}^{(2)}(k+1)+\frac{1}{2}\omega\tilde{Z}^{(3)}(k+1)+\frac{1}{2}\omega\tilde{Z}^{(4)}(k+1)\right\|^2\\
&=\frac{1}{4}(1-\omega)^2\|\tilde{Z}^{(1)}(k+1)\|^2+\frac{1}{4}(1-\omega)^2\|\tilde{Z}^{(2)}(k+1)\|^2+\frac{1}{4}\omega^2\|\tilde{Z}^{(3)}(k+1)\|^2+\frac{1}{4}\omega^2\|\tilde{Z}^{(4)}(k+1)\|^2\\
&\quad+\frac{1}{2}(1-\omega)^2\mathrm{Retr}((\tilde{Z}^{(1)}(k+1))^H\tilde{Z}^{(2)}(k+1))+\frac{1}{2}\omega(1-\omega)\,\mathrm{Retr}((\tilde{Z}^{(1)}(k+1))^H\tilde{Z}^{(3)}(k+1))\\
&\quad+\frac{1}{2}\omega(1-\omega)\,\mathrm{Retr}((\tilde{Z}^{(1)}(k+1))^H\tilde{Z}^{(4)}(k+1))+\frac{1}{2}\omega(1-\omega)\,\mathrm{Retr}((\tilde{Z}^{(2)}(k+1))^H\tilde{Z}^{(3)}(k+1))\\
&\quad+\frac{1}{2}\omega(1-\omega)\,\mathrm{Retr}((\tilde{Z}^{(2)}(k+1))^H\tilde{Z}^{(4)}(k+1))+\frac{1}{2}\omega^2\mathrm{Retr}((\tilde{Z}^{(3)}(k+1))^H\tilde{Z}^{(4)}(k+1))\\
&\le\frac{1}{4}(1-\omega)^2\|\tilde{Z}^{(1)}(k+1)\|^2+\frac{1}{4}(1-\omega)^2\|\tilde{Z}^{(2)}(k+1)\|^2+\frac{1}{4}\omega^2\|\tilde{Z}^{(3)}(k+1)\|^2+\frac{1}{4}\omega^2\|\tilde{Z}^{(4)}(k+1)\|^2\\
&\quad+\frac{1}{2}(1-\omega)^2\|\tilde{Z}^{(1)}(k+1)\|\|\tilde{Z}^{(2)}(k+1)\|+\frac{1}{2}\omega(1-\omega)\|\tilde{Z}^{(1)}(k+1)\|\|\tilde{Z}^{(3)}(k+1)\|\\
&\quad+\frac{1}{2}\omega(1-\omega)\|\tilde{Z}^{(1)}(k+1)\|\|\tilde{Z}^{(4)}(k+1)\|+\frac{1}{2}\omega(1-\omega)\|\tilde{Z}^{(2)}(k+1)\|\|\tilde{Z}^{(3)}(k+1)\|\\
&\quad+\frac{1}{2}\omega(1-\omega)\|\tilde{Z}^{(2)}(k+1)\|\|\tilde{Z}^{(4)}(k+1)\|+\frac{1}{2}\omega^2\|\tilde{Z}^{(3)}(k+1)\|\|\tilde{Z}^{(4)}(k+1)\|
\end{aligned}
$$

$$
\begin{aligned}
&\le \frac{1}{4}(1-\omega)^2\|\tilde{Z}^{(1)}(k+1)\|^2 + \frac{1}{4}(1-\omega)^2\|\tilde{Z}^{(2)}(k+1)\|^2 + \frac{1}{4}\omega^2\|\tilde{Z}^{(3)}(k+1)\|^2 + \frac{1}{4}\omega^2\|\tilde{Z}^{(4)}(k+1)\|^2\\
&\quad + \frac{1}{4}(1-\omega)^2(\|\tilde{Z}^{(1)}(k+1)\|^2 + \|\tilde{Z}^{(2)}(k+1)\|^2) + \frac{1}{4}\omega^2(\|\tilde{Z}^{(3)}(k+1)\|^2 + \|\tilde{Z}^{(4)}(k+1)\|^2)\\
&\quad + \frac{1}{4}\omega(1-\omega)(\|\tilde{Z}^{(1)}(k+1)\|^2 + \|\tilde{Z}^{(3)}(k+1)\|^2) + \frac{1}{4}\omega(1-\omega)(\|\tilde{Z}^{(1)}(k+1)\|^2 + \|\tilde{Z}^{(4)}(k+1)\|^2)\\
&\quad + \frac{1}{4}\omega(1-\omega)(\|\tilde{Z}^{(2)}(k+1)\|^2 + \|\tilde{Z}^{(3)}(k+1)\|^2) + \frac{1}{4}\omega(1-\omega)(\|\tilde{Z}^{(2)}(k+1)\|^2 + \|\tilde{Z}^{(4)}(k+1)\|^2)\\
&= \frac{1}{4}(1-\omega)^2\|\tilde{Z}^{(1)}(k+1)\|^2 + \frac{1}{4}(1-\omega)^2\|\tilde{Z}^{(2)}(k+1)\|^2 + \frac{1}{4}\omega^2\|\tilde{Z}^{(3)}(k+1)\|^2 + \frac{1}{4}\omega^2\|\tilde{Z}^{(4)}(k+1)\|^2\\
&\quad + \frac{1}{4}(1-\omega)^2(\|\tilde{Z}^{(1)}(k+1)\|^2 + \|\tilde{Z}^{(2)}(k+1)\|^2) + \frac{1}{4}\omega^2(\|\tilde{Z}^{(3)}(k+1)\|^2 + \|\tilde{Z}^{(4)}(k+1)\|^2)\\
&\quad + \frac{1}{4}\omega(1-\omega)(2\|\tilde{Z}^{(1)}(k+1)\|^2 + 2\|\tilde{Z}^{(2)}(k+1)\|^2 + 2\|\tilde{Z}^{(3)}(k+1)\|^2 + 2\|\tilde{Z}^{(4)}(k+1)\|^2)\\
&= \frac{1}{2}(1-\omega)\|\tilde{Z}^{(1)}(k+1)\|^2 + \frac{1}{2}(1-\omega)\|\tilde{Z}^{(2)}(k+1)\|^2 + \frac{1}{2}\omega\|\tilde{Z}^{(3)}(k+1)\|^2 + \frac{1}{2}\omega\|\tilde{Z}^{(4)}(k+1)\|^2\\
&\le \frac{1}{2}(1-\omega)\|\tilde{Z}(k)\|^2 - \frac{1}{2}(1-\omega)\omega\mu\left(1-\frac{1}{4}\omega\mu\|D_{11}\|_2^2\|D_{12}\|_2^2\right)\|W_1^{(k)}\|^2\\
&\quad + \frac{1}{2}(1-\omega)\|\tilde{\tilde{Z}}(k)\|^2 - \frac{1}{2}(1-\omega)\omega\mu\left(1-\frac{1}{4}\omega\mu\|D_{21}\|_2^2\|D_{22}\|_2^2\right)\|W_2^{(k)}\|^2\\
&\quad + \frac{1}{2}\omega\|\tilde{\tilde{Z}}(k)\|^2 - \frac{1}{2}\omega(1-\omega)\mu\left(1-\frac{1}{4}(1-\omega)\mu\|D_{31}\|_2^2\|D_{32}\|_2^2\right)\|W_3^{(k)}\|^2\\
&\quad + \frac{1}{2}\omega\|\tilde{\tilde{Z}}(k)\|^2 - \frac{1}{2}\omega(1-\omega)\mu\left(1-\frac{1}{4}(1-\omega)\mu\|D_{41}\|_2^2\|D_{42}\|_2^2\right)\|W_4^{(k)}\|^2\\
&\le \|\tilde{Z}(k)\|^2 - \frac{1}{2}(1-\omega)\omega\mu\left(1-\frac{1}{4}\omega\mu\|D_{11}\|_2^2\|D_{12}\|_2^2\right)\|W_1^{(k)}\|^2\\
&\quad + \frac{1}{2}(1-\omega)\|\tilde{\tilde{Z}}(k)\|^2 - \frac{1}{2}(1-\omega)\omega\mu\left(1-\frac{1}{4}\omega\mu\|D_{21}\|_2^2\|D_{22}\|_2^2\right)\|W_2^{(k)}\|^2\\
&\quad + \frac{1}{2}\omega\|\tilde{\tilde{Z}}(k)\|^2 - \frac{1}{2}\omega(1-\omega)\mu\left(1-\frac{1}{4}(1-\omega)\mu\|D_{31}\|_2^2\|D_{32}\|_2^2\right)\|W_3^{(k)}\|^2\\
&\quad + \frac{1}{2}\omega\|\tilde{\tilde{Z}}(k)\|^2 - \frac{1}{2}\omega(1-\omega)\mu\left(1-\frac{1}{4}(1-\omega)\mu\|D_{41}\|_2^2\|D_{42}\|_2^2\right)\|W_4^{(k)}\|^2\\
&\le \|\tilde{Z}(0)\|^2 - \frac{1}{2}(1-\omega)\omega\mu\left(1-\frac{1}{4}\omega\mu\|D_{11}\|_2^2\|D_{12}\|_2^2\right)\sum_{i=0}^{k}\|W_1^{(i)}\|^2\\
&\quad + \frac{1}{2}(1-\omega)\sum_{i=0}^{k}\|\tilde{\tilde{Z}}(i)\|^2 - \frac{1}{2}(1-\omega)\omega\mu\left(1-\frac{1}{4}\omega\mu\|D_{21}\|_2^2\|D_{22}\|_2^2\right)\sum_{i=0}^{k}\|W_2^{(i)}\|^2\\
&\quad + \frac{1}{2}\omega\sum_{i=0}^{k}\|\tilde{\tilde{Z}}(i)\|^2 - \frac{1}{2}\omega(1-\omega)\mu\left(1-\frac{1}{4}(1-\omega)\mu\|D_{31}\|_2^2\|D_{32}\|_2^2\right)\sum_{i=0}^{k}\|W_3^{(i)}\|^2\\
&\quad + \frac{1}{2}\omega\sum_{i=0}^{k}\|\tilde{\tilde{Z}}(i)\|^2 - \frac{1}{2}\omega(1-\omega)\mu\left(1-\frac{1}{4}(1-\omega)\mu\|D_{41}\|_2^2\|D_{42}\|_2^2\right)\sum_{i=0}^{k}\|W_4^{(i)}\|^2.
\end{aligned}
\tag{55}
$$

According to the methods applied in [19,20,24], it follows that

$$
\frac{1}{2}(1-\omega)\sum_{i=0}^{k}\|\tilde{\tilde{Z}}(i)\|^2 < +\infty, \quad \frac{1}{2}\omega\sum_{i=0}^{k}\|\tilde{\tilde{Z}}(i)\|^2 < +\infty, \quad \frac{1}{2}\omega\sum_{i=0}^{k}\|\tilde{Z}(i)\|^2 < +\infty. \tag{56}
$$

Hence if

$$
\begin{aligned}
&1-\frac{1}{4}\omega\mu\|D_{11}\|_2^2\|D_{12}\|_2^2 > 0, \quad 1-\frac{1}{4}\omega\mu\|D_{21}\|_2^2\|D_{22}\|_2^2 > 0, \quad 1-\frac{1}{4}(1-\omega)\mu\|D_{31}\|_2^2\|D_{32}\|_2^2 > 0,\\
&1-\frac{1}{4}(1-\omega)\mu\|D_{41}\|_2^2\|D_{42}\|_2^2 > 0,
\end{aligned}
\tag{57}
$$

then (55) implies that $\sum_{i=0}^{\infty}\|W_1^{(i)}\|^2 < +\infty$. According to the convergence theorem of series, we have $\lim_{i\to+\infty}\|W_1^{(i)}\| = 0$, i.e.,

$$\lim_{i\to+\infty} W_1^{(i)} = \lim_{i\to+\infty} A_1\tilde{Z}(i)B_1 + A_2\bar{\tilde{Z}}(i)B_2 + A_3\tilde{Z}^T(i)B_3 + A_4\tilde{Z}^H(i)B_4 = 0. \tag{58}$$

Inasmuch as the CCT Sylvester matrix equation (2) has a unique solution, it follows from (58) that $\lim_{k\to+\infty}\tilde{Z}(k) = 0$ by Lemma 3.2. This means that $\lim_{k\to+\infty}Z(k) = Z^*$. From (57), we have

$$0 < \mu < \min\left\{\frac{4}{\omega\|D_{11}\|_2^2\|D_{12}\|_2^2}, \frac{4}{\omega\|D_{21}\|_2^2\|D_{22}\|_2^2}, \frac{4}{(1-\omega)\|D_{31}\|_2^2\|D_{32}\|_2^2}, \frac{4}{(1-\omega)\|D_{41}\|_2^2\|D_{42}\|_2^2}\right\}, \tag{59}$$

or

$$0 < \omega < \frac{4}{\mu\|D_{11}\|_2^2\|D_{12}\|_2^2}, \quad 0 < \omega < \frac{4}{\mu\|D_{21}\|_2^2\|D_{22}\|_2^2}, \quad 0 < 1-\omega < \frac{4}{\mu\|D_{31}\|_2^2\|D_{32}\|_2^2}, \quad 0 < 1-\omega < \frac{4}{\mu\|D_{41}\|_2^2\|D_{42}\|_2^2},$$

which results in

$$1 - \min\left\{\frac{4}{\mu\|D_{31}\|_2^2\|D_{32}\|_2^2}, \frac{4}{\mu\|D_{41}\|_2^2\|D_{42}\|_2^2}\right\} < \omega < \min\left\{\frac{4}{\mu\|D_{11}\|_2^2\|D_{12}\|_2^2}, \frac{4}{\mu\|D_{21}\|_2^2\|D_{22}\|_2^2}\right\}. \tag{60}$$

Inequality (60) holds if

$$\min\left\{\frac{4}{\mu\|D_{11}\|_2^2\|D_{12}\|_2^2}, \frac{4}{\mu\|D_{21}\|_2^2\|D_{22}\|_2^2}\right\} + \min\left\{\frac{4}{\mu\|D_{31}\|_2^2\|D_{32}\|_2^2}, \frac{4}{\mu\|D_{41}\|_2^2\|D_{42}\|_2^2}\right\} > 1, \tag{61}$$

then combining (60)–(61) with $0 < \omega < 1$ gives (48). □

4 Numerical experiments

This section gives several numerical examples to validate the effectiveness of the IMGI and the IMRGI algorithms, and compare their numerical performances with those of the GI, OGI, RGI, and MGI ones, in terms of the number of iteration steps (IT) and the elapsed time in seconds (CPU).

All the computations were performed in MATLAB (version R2016b) on a personal computer with Intel (R) Pentium (R) CPU G3240T 2.870 GHz, 16.0 GB memory and Windows 10 system. For all tested algorithms, the initial matrices are taken to be $Z(0) = Z_i(0) = I_2 \times 10^{-6}$ $(i = 1, 2, 3, 4)$.

Example 4.1. [21] We consider the CCT Sylvester matrix equation (2) with the following coefficient matrices:

$$A_1 = \begin{pmatrix} 13+2i & 1+2i \\ 2-i & 16+8i \end{pmatrix}, \quad B_1 = \begin{pmatrix} 15+7i & 2+5i \\ 9+7i & 18+10i \end{pmatrix}, \quad A_2 = \begin{pmatrix} 9+20i & 5+3i \\ 2+2i & 2+9i \end{pmatrix},$$
$$B_2 = \begin{pmatrix} 9i+19 & 4i+5 \\ 1+5i & 16+16i \end{pmatrix}, \quad A_3 = \begin{pmatrix} 11i+3 & 5i+7 \\ 5+10i & 13+19i \end{pmatrix}, \quad B_3 = \begin{pmatrix} 1+12i & 5-5i \\ 6+2i & 19+18i \end{pmatrix},$$
$$A_4 = \begin{pmatrix} 16+7i & 7+8i \\ 1+7i & 12+13i \end{pmatrix}, \quad B_4 = \begin{pmatrix} 20+13i & 7+5i \\ 5+2i & 14+10i \end{pmatrix}, \quad H = \begin{pmatrix} 706+1397i & 126-2886i \\ -2294-1179i & -426-4404i \end{pmatrix}.$$

The unique solution of this matrix equation is

$$Z^* = \begin{pmatrix} 3+i & 1-i \\ -5+i & -2+3i \end{pmatrix}.$$

For this example, all runs are terminated once, and the number of iterations k exceeds 20,000, or $\frac{\|Z(k)-Z^*\|}{\|X^*\|} \le \delta$ (denoted as "ERR") with δ being a positive number.

In Table 1, we compare the proposed IMGI and IMRGI algorithms with GI, OGI, RGI, and MGI ones with respect to IT and CPU times. Here, the parameters of the GI, OGI, and RGI algorithms are taken as in [21]. And the parameters of the MGI, IMGI, and IMRGI algorithms are as follows:

$$\mu_{\mathrm{MGI}} = \min\left\{\frac{1}{\|A_1\|_2^2\|B_1\|_2^2}, \frac{1}{\|A_2\|_2^2\|B_2\|_2^2}, \frac{1}{\|A_3\|_2^2\|B_3\|_2^2}, \frac{1}{\|A_4\|_2^2\|B_4\|_2^2}\right\} = 1.6317 \times 10^{-6},$$

$$\mu_{\mathrm{IMGI}} = \min\left\{\frac{2}{\|D_{11}\|_2^2\|D_{12}\|_2^2}, \frac{2}{\|D_{21}\|_2^2\|D_{22}\|_2^2}, \frac{2}{\|D_{31}\|_2^2\|D_{32}\|_2^2}, \frac{2}{\|D_{41}\|_2^2\|D_{42}\|_2^2}\right\} = 5.5089 \times 10^{-6}, \quad \omega_{\mathrm{IMRGI}} = \frac{1}{1.8},$$

$$\mu_{\mathrm{IMRGI}} = \min\left\{\frac{4}{\omega\|D_{11}\|_2^2\|D_{12}\|_2^2}, \frac{4}{\omega\|D_{21}\|_2^2\|D_{22}\|_2^2}, \frac{4}{(1-\omega)\|D_{31}\|_2^2\|D_{32}\|_2^2}, \frac{4}{(1-\omega)\|D_{41}\|_2^2\|D_{42}\|_2^2}\right\} = 2.4790 \times 10^{-5}.$$

According to Table 1, we see that all tested methods are convergent within the maximum number of iterations for all cases, and the IMGI and the IMRGI algorithms outperform the other ones with respect to IT and CPU times. The growth rates of the IT for the IMGI and the IMRGI algorithms are slower than the other ones, which reveals that the proposed algorithms are more stable than the other tested ones. Besides, the IMRGI algorithm performs better than the IMGI algorithm with proper relaxation factor ω. This means that the relaxation technique can improve the efficiency of an algorithm.

With the parameters in Table 1, the iterative sequences $\{Z(k)\}$ generated by the IMGI and the IMRGI algorithms with the changing of IT are reported in Table 2. We conclude from Table 2 that with the increases of IT, the iterative sequences $\{Z(k)\}$ produced by the IMGI and the IMRGI algorithms are gradually tended to the exact solution Z^*. This validates the proposed algorithms can work out the approximate solution of the CCT Sylvester matrix equations effectively.

To compare the tested algorithms fairly, we test the algorithms with the same values of μ in Table 3. We adopt three different values of μ for the tested algorithms, which are obtained by the weighted combinations of μ used for the tested algorithms in Table 1. And the expression of μ is $\mu = k_1\mu_{\mathrm{GI}} + k_2\mu_{\mathrm{OGI}} + k_3\mu_{\mathrm{RGI}} + k_4\mu_{\mathrm{MGI}} + k_5\mu_{\mathrm{IMGI}} + k_6\mu_{\mathrm{IMRGI}}$, where μ_{GI}, μ_{OGI}, μ_{RGI}, μ_{MGI}, μ_{IMGI}, and μ_{IMRGI} are the values of μ used in the GI, OGI, RGI, MGI, IMGI, and IMRGI algorithms, respectively, and $\sum_{i=1}^{6} k_i = 1 (k_i \geq 0)$. We take three group numbers $(k_1, k_2. k_3, k_4, k_5, k_6) = (1/6, 1/6, 1/6, 1/6, 1/6, 1/6)$, $(0.4, 0.2, 0.1, 0.25, 0, 0.05)$, and $(0.3, 0.25, 0, 0.3, 0, 0.15)$. By direct computations, it has $\mu = 7.0990 \times 10^{-6}$, 3.3924×10^{-6}, and 5.1411×10^{-6}, respectively. The numerical

Table 1: Numerical results of the tested iterative algorithms for Example 4.1

Method	δ	0.1	0.01	0.001	0.0001	0.00001
	IT	250	819	1389	1959	2529
GI	CPU	0.0265	0.0957	0.1378	0.2177	0.2299
	ERR	9.97×10^{-2}	1.00×10^{-2}	9.99×10^{-4}	1.00×10^{-4}	1.00×10^{-5}
	IT	196	634	1,075	1,516	1,957
OGI	CPU	0.0190	0.0604	0.1023	0.1319	0.1698
	ERR	9.95×10^{-2}	1.00×10^{-2}	9.98×10^{-4}	9.99×10^{-5}	9.99×10^{-6}
	IT	94	238	403	569	735
RGI	CPU	0.0157	0.0261	0.0405	0.0584	0.0714
	ERR	9.86×10^{-2}	1.00×10^{-2}	9.91×10^{-4}	9.89×10^{-5}	9.89×10^{-6}
	IT	59	149	251	354	457
MGI	CPU	0.0099	0.0166	0.0275	0.0449	0.0493
	ERR	9.96×10^{-2}	9.90×10^{-3}	9.89×10^{-4}	9.85×10^{-5}	9.83×10^{-6}
	IT	19	42	70	98	127
IMGI	CPU	0.0041	0.0083	0.0097	0.0133	0.0142
$\mu = 5.5089 \times 10^{-6}$	ERR	9.48×10^{-2}	1.00×10^{-2}	9.63×10^{-4}	1.00×10^{-4}	9.68×10^{-6}
	IT	17	38	63	91	116
IMRGI	CPU	0.0019	0.0045	0.0073	0.0089	0.0121
$\omega = 1/1.8, \mu = 2.4790 \times 10^{-5}$	ERR	7.21×10^{-2}	8.20×10^{-3}	9.44×10^{-4}	8.38×10^{-5}	9.33×10^{-6}

Table 2: The iterative solutions for IMGI ($\mu = 5.5089 \times 10^{-6}$) and IMRGI ($\omega = 1/1.8$, $\mu = 2.4790 \times 10^{-5}$) algorithms for Example 4.1

Algorithm	k	z_{11}	z_{12}	z_{21}	z_{22}
	30	$3.0034 + 1.0028i$	$0.9357 - 1.0784i$	$-4.8978 + 0.9072i$	$-2.0080 + 2.9096i$
	60	$2.9999 + 1.0010i$	$0.9961 - 1.0066i$	$-4.9937 + 0.9903i$	$-2.0006 + 2.9967i$
IMGI	90	$3.0000 + 1.0001i$	$0.9997 - 1.0006i$	$-4.9995 + 0.9991i$	$-2.0001 + 2.9998i$
	120	$3.0000 + 1.0000i$	$1.0000 - 1.0001i$	$-5.0000 + 0.9999i$	$-2.0000 + 3.0000i$
	150	$3.0000 + 1.0000i$	$1.0000 - 1.0000i$	$-5.0000 + 1.0000i$	$-2.0000 + 3.0000i$
	30	$3.0018 + 0.9786i$	$0.9493 - 1.0854i$	$-4.9112 + 0.9117i$	$-1.9906 + 2.9054i$
	60	$2.9995 + 1.0005i$	$0.9968 - 1.0050i$	$-4.9961 + 0.9930i$	$-2.0012 + 2.9970i$
IMRGI	90	$3.0000 + 1.0001i$	$0.9998 - 1.0003i$	$-4.9997 + 0.9996i$	$-2.0000 + 2.9999i$
	120	$3.0000 + 1.0000i$	$1.0000 - 1.0000i$	$-5.0000 + 1.0000i$	$-2.0000 + 3.0000i$
	150	$3.0000 + 1.0000i$	$1.0000 - 1.0000i$	$-5.0000 + 1.0000i$	$-2.0000 + 3.0000i$
	Solution	$3 + i$	$1 - i$	$-5 + i$	$-2 + 3i$

Table 3: Numerical results of the tested iterative algorithms for Example 4.1 with three different values of μ

	Method	τ	0.1	0.01	0.001	0.0001	0.00001
$\mu = 7.0990 \times 10^{-6}$	GI	IT	Fail	Fail	Fail	Fail	Fail
		CPU	—	—	—	—	—
	OGI	IT	Fail	Fail	Fail	Fail	Fail
		CPU	—	—	—	—	—
	RGI	IT	Fail	Fail	Fail	Fail	Fail
		CPU	—	—	—	—	—
	MGI	IT	Fail	Fail	Fail	Fail	Fail
		CPU	—	—	—	—	—
	IMGI	IT	Fail	Fail	Fail	Fail	Fail
		CPU	—	—	—	—	—
	IMRGI	IT	52	113	196	284	372
	$\omega = 0.5$	CPU	0.0064	0.0134	0.0224	0.0322	0.0463
$\mu = 3.3924 \times 10^{-6}$	GI	IT	Fail	Fail	Fail	Fail	Fail
		CPU	—	—	—	—	—
	OGI	IT	Fail	Fail	Fail	Fail	Fail
		CPU	—	—	—	—	—
	RGI	IT	178	454	769	1086	1403
	$\omega = 0.5$	CPU	0.0203	0.0501	0.0849	0.1224	0.1543
	MGI	IT	29	73	122	171	220
		CPU	0.0035	0.0089	0.0145	0.0203	0.0257
	IMGI	IT	29	63	107	153	199
		CPU	8.0020×10^{-4}	0.0039	0.0129	0.0189	0.0226
	IMRGI	IT	108	231	403	586	770
	$\omega = 0.5$	CPU	0.0124	0.0279	0.0459	0.0671	0.0868
$\mu = 5.1411 \times 10^{-6}$	GI	IT	Fail	Fail	Fail	Fail	Fail
		CPU	—	—	—	—	—
	OGI	IT	Fail	Fail	Fail	Fail	Fail
		CPU	—	—	—	—	—
	RGI	IT	118	300	507	716	924
	$\omega = 0.5$	CPU	0.0134	0.0336	0.0558	0.0806	0.1239
	MGI	IT	Fail	Fail	Fail	Fail	Fail
		CPU	—	—	—	—	—
	IMGI	IT	20	45	74	105	135
		CPU	0.0025	0.0057	0.0090	0.0131	0.0152
	IMRGI	IT	72	154	268	389	510
	$\omega = 0.5$	CPU	0.0091	0.0178	0.0324	0.0442	0.0584

results of the tested algorithms with these three values of μ are listed in Table 3, and the relaxation factor ω used in the RGI and the IMRGI algorithms are the experimental optimal ones. From the numerical results shown in Table 3, we see that the GI and the OGI algorithms are invalid for all cases. For $\mu = 7.0990 \times 10^{-6}$, all algorithms are not convergent except for the IMRGI one. And for $\mu = 3.3924 \times 10^{-6}$, the IMGI algorithm performs the best, and the IMRGI algorithm outperforms the RGI one in terms of IT and CPU times. In addition, for $\mu = 5.1411 \times 10^{-6}$, the proposed IMGI and IMRGI algorithms are superior to the other ones. Although the IMGI algorithm has better numerical behavior than the IMRGI one for some cases, the latter one is convergent for all cases.

The graphs of ERR(log10) against the IT of the tested algorithms for four different values of δ are plotted in Figures 1 and 2. As shown in Figure 1, the IMGI and the IMRGI algorithms are superior to the other ones because they require less IT to achieve the termination criterion. And the numerical performances of the IMGI and the IMRGI algorithms are comparable, and the latter one is slightly better than the former one.

Example 4.2. [30] We consider the CCT Sylvester matrix equation (2) with the following parametric matrices

$$A_1 = \begin{pmatrix} 13+10i & 6+6i \\ 2+i & 16+18i \end{pmatrix}, \quad B_1 = \begin{pmatrix} 15+17i & 8+5i \\ 9+7i & 18+10i \end{pmatrix}, \quad A_2 = \begin{pmatrix} 19+20i & 5+3i \\ 2+2i & 20+19i \end{pmatrix},$$
$$B_2 = \begin{pmatrix} 19+19i & 5+4i \\ 1+5i & 16+16i \end{pmatrix}, \quad A_3 = \begin{pmatrix} 13+11i & 7+5i \\ 5+10i & 13+19i \end{pmatrix}, \quad B_3 = \begin{pmatrix} 11+12i & 5+5i \\ 6+2i & 19+18i \end{pmatrix},$$
$$A_4 = \begin{pmatrix} 16+17i & 7+8i \\ 1+7i & 12+13i \end{pmatrix}, \quad B_4 = \begin{pmatrix} 20+13i & 7+5i \\ 5+2i & 14+10i \end{pmatrix}, \quad H = \begin{pmatrix} -633+2558i & -1304-4267i \\ -665-6248i & -556-7565i \end{pmatrix}.$$

The unique solution of this matrix equation is

$$Z^* = \begin{pmatrix} 3 & -i \\ -5+i & -2+3i \end{pmatrix}.$$

In this example, we take

$$\text{RES} = \frac{\|E - A_1 Z(k) B_1 - A_2 \bar{Z}(k) B_2 - A_3 Z(k)^T B_3 - A_4 Z(k)^H B_4\|}{\|H\|} \le \tau,$$

as the termination condition with a positive number $\tau > 0$. And the prescribed maximum iterative number $k_{\max}$ is set to be 20,000.

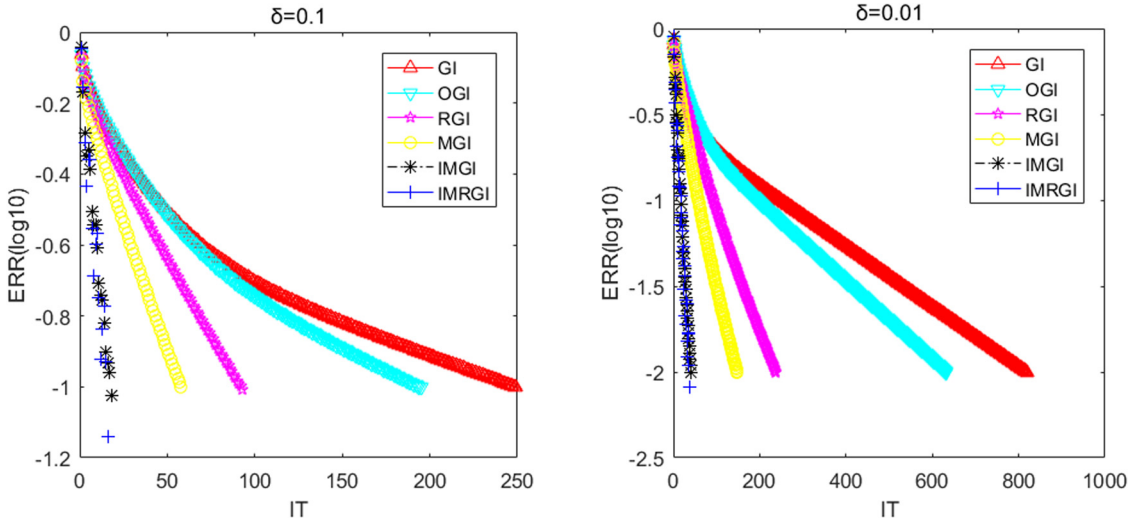

Figure 1: The logarithmic of the ERR of the tested algorithms for Example 4.1 with $\delta = 0.1$ (left) and $\delta = 0.01$ (right).

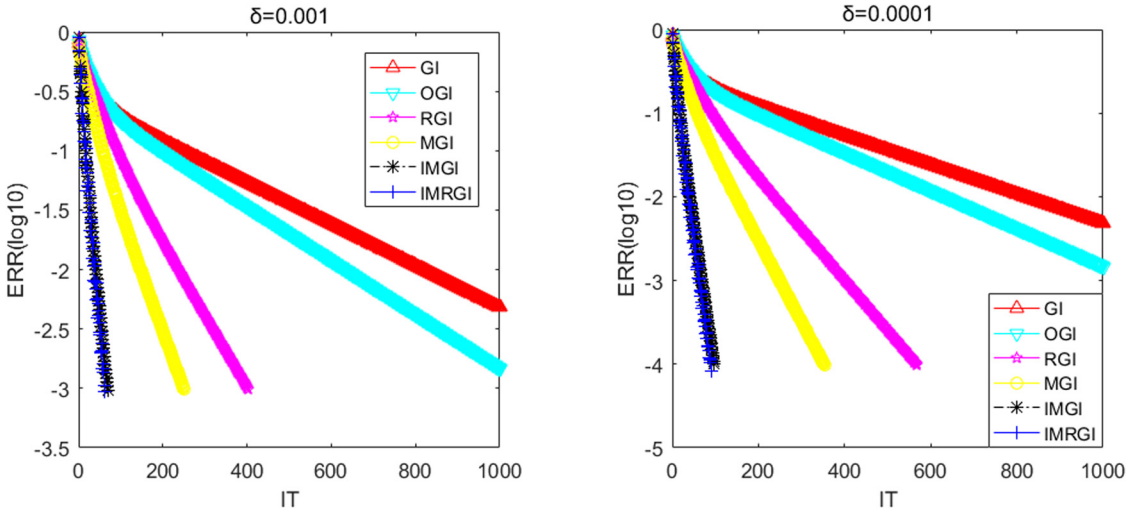

Figure 2: The logarithmic of the ERR of the tested algorithms for Example 4.1 with $\delta = 0.001$ (left) and $\delta = 0.0001$ (right).

Numerical results for different values of τ are reported in Table 4. In this table, we list the parameters, IT, CPU, and RES of the tested algorithms. The parameters taken in the tested algorithms are determined by using the same methods in Example 4.1. It is found in Table 4 that the GI and the IMGI algorithms are divergence for all cases, and the OGI and the RGI algorithms are not convergent for $\tau = 0.00001$, while the MGI and the IMRGI algorithms are convergent for all cases. With the proper parameter μ, the IMGI algorithm outperforms the GI, OGI and RGI ones in terms of IT and CPU times. In addition, the IMRGI algorithms performs the best among all algorithms. The exception is that the MGI algorithm needs less IT than the IMRGI one for $\tau = 0.1$. Finally, the

Table 4: Numerical results of the tested iterative algorithms for Example 4.2

Method	τ	0.1	0.01	0.001	0.0001	0.00001
	IT	Fail	Fail	Fail	Fail	Fail
GI	CPU	Fail	Fail	Fail	Fail	Fail
$\mu = 6.9817 \times 10^{-7}$	RES	—	—	—	—	—
	IT	4,363	9,450	14,537	19,624	Fail
OGI	CPU	0.4973	1.2514	1.7253	2.2924	Fail
$\mu = 6.9801 \times 10^{-7}$	RES	1.00×10^{-1}	1.00×10^{-2}	1.00×10^{-3}	1.00×10^{-4}	—
	IT	4,363	9,450	14,537	19,624	Fail
RGI	CPU	0.5793	1.1722	1.6961	2.2689	Fail
$\mu = 3.1411 \times 10^{-6}$	RES	1.00×10^{-1}	1.00×10^{-2}	1.00×10^{-3}	1.00×10^{-4}	—
	IT	5	91	198	306	413
MGI	CPU	0.0103	0.0137	0.0236	0.0343	0.0500
$\mu = 1.0360 \times 10^{-6}$	RES	9.42×10^{-2}	1.00×10^{-2}	9.94×10^{-4}	9.80×10^{-5}	9.95×10^{-6}
	IT	Fail	Fail	Fail	Fail	Fail
IMGI	CPU	Fail	Fail	Fail	Fail	Fail
$\mu = 3.6401 \times 10^{-6}$	RES	—	—	—	—	—
	IT	5	100	227	355	482
IMGI	CPU	6.5790×10^{-4}	0.0115	0.0256	0.0373	0.0508
$\mu = 1.8200 \times 10^{-6}$	RES	8.40×10^{-2}	9.90×10^{-3}	1.00×10^{-3}	9.79×10^{-5}	9.82×10^{-6}
	IT	14	77	165	254	343
IMRGI	CPU	0.0039	0.0096	0.0185	0.0283	0.0377
$\omega = 1/4, \mu = 1.4690 \times 10^{-5}$	RES	7.17×10^{-2}	1.00×10^{-2}	9.88×10^{-4}	9.91×10^{-5}	9.97×10^{-6}

IMRGI algorithm is more stable than the MGI one, due to the fact that the changing scope of the IT of the IMRGI algorithm is smaller than that for the MGI one.

Table 5 lists the same items as those in Table 2. The results in Table 5 indicate that the iterative solutions generated by the IMGI and the IMRGI algorithms are gradually tended to the exact solution Z^* as the IT increases.

Furthermore, to further show the advantages of the IMGI and the IMRGI algorithms over the other tested ones, the performances of the tested algorithms with three different values of μ are exhibited in Table 6. And the three values of μ are determined in the same manner as we used in Example 4.1. It is observed from Table 6 that the RGI and the IMRGI algorithms are convergent, while the other tested ones are invalid for $\mu = 3.6805 \times 10^{-6}$. And the IMRGI algorithm is superior to the RGI one with respect to IT and CPU times. In addition, as $\mu = 1.7265 \times 10^{-6}$, the GI and the OGI algorithms are divergence, and the RGI, IMGI, and IMRGI algorithms always outperform the MGI one. The IMRGI algorithm is less efficient than the RGI one except for the case of $\tau = 0.1$, and the IMGI algorithm leads to much better performance than the RGI one, and the advantage of the IMGI algorithm becomes more pronounced as τ decreases. Finally, the proposed IMGI and IMRGI algorithms have advantages over the other ones for $\mu = 2.8983 \times 10^{-6}$, and the IMGI algorithm is more stable than the IMRGI one for this case.

Figures 3 and 4 display the convergence curves of the GI, OGI, RGI, MGI, IMGI, and IMRGI algorithms for four different values of τ. From these two figures, we observe that the IMRGI algorithm converges faster than the other ones, and the IMGI algorithm outperforms the GI, OGI, and RGI ones with respect to IT.

Example 4.3. [30] Consider the CCT Sylvester matrix equation (2) with

$$A_1 = \begin{pmatrix} 16 & -2i \\ 3-i & 9-2i \end{pmatrix}, \quad B_1 = \begin{pmatrix} 6-2i & 2 \\ i & 15+3i \end{pmatrix}, \quad A_2 = B_2 = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix},$$
$$A_3 = B_3 = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}, \quad A_4 = \begin{pmatrix} 6+10i & -1 \\ 10-i & 5i \end{pmatrix}, \quad B_4 = \begin{pmatrix} 16 & 5i \\ 1-3i & 5 \end{pmatrix},$$
$$H = \begin{pmatrix} 585-235i & 1079+318i \\ -401-516i & 453+232i \end{pmatrix}, \quad Z^* = \begin{pmatrix} -2+5i & 3-i \\ 1 & 3i \end{pmatrix}.$$

In this example, we adopt the termination criterion as in Example 4.1, i.e., ERR = $\frac{\|Z(k)-Z^*\|}{\|Z^*\|} \le \delta$ with $\delta > 0$ or k exceeds the prescribed maximal number of iteration steps 20,000.

The numerical results of the tested algorithms for Example 4.3 are shown in Table 7. It follows from Table 7 that the GI algorithm has the slowest convergence speed, and the IMGI and the IMRGI algorithms have advantages over the other ones except for the case of $\delta = 0.1$. The OGI and the RGI algorithms have the same numerical performances, and they outperform the GI one. Also, the MGI algorithm is better than the IMGI and the IMRGI ones when $\delta = 0.1$ and 0.01, whereas the latter ones are superior to the former one when δ becomes small.

Table 5: The iterative solutions for IMGI ($\mu = 1.8200 \times 10^{-6}$) and IMRGI ($\omega = 1/4$, $\mu = 1.4690 \times 10^{-5}$) algorithms for Example 4.2

Algorithm	k	z_{11}	z_{12}	z_{21}	z_{22}
IMGI	100	2.9781 – 0.0219i	–0.1105 – 1.1006i	–4.7773 + 0.6409i	–2.0645 + 3.1367i
	200	2.9930 – 0.0661i	–0.2367 – 1.0113i	–4.6391 + 0.9976i	–2.0944 + 2.8720i
	300	2.9910 – 0.0652i	–0.2182 – 1.0370i	–4.6550 + 0.9220i	–2.0933 + 2.8583i
	400	2.9911 – 0.0641i	–0.2187 – 1.0338i	–4.6554 + 0.9313i	–2.0927 + 2.8688i
	500	2.9911 – 0.0644i	–0.2190 – 1.0339i	–4.6549 + 0.9311i	–2.0929 + 2.8667i
IMRGI	100	2.9894 – 0.0333i	–0.1913 – 1.0051i	–4.6948 + 0.9957i	–2.0727 + 3.1052i
	200	2.9912 – 0.0663i	–0.2197 – 1.0372i	–4.6530 + 0.9219i	–2.0940 + 2.8504i
	300	2.9911 – 0.0642i	–0.2190 – 1.0336i	–4.6550 + 0.9318i	–2.0928 + 2.8680i
	400	2.9911 – 0.0643i	–0.2189 – 1.0340i	–4.6550 + 0.9308i	–2.0928 + 2.8668i
	500	2.9911 – 0.0643i	–0.2190 – 1.0340i	–4.6550 + 0.9309i	–2.0928 + 2.8669i
	Solution	3	$-i$	$-5+i$	$-2+3i$

Table 6: Numerical results of the tested iterative algorithms for Example 4.2 with three different values of μ

	Method	τ	0.1	0.01	0.001	0.0001	0.00001
$\mu = 3.6805 \times 10^{-6}$	GI	IT	Fail	Fail	Fail	Fail	Fail
		CPU	—	—	—	—	—
	OGI	IT	Fail	Fail	Fail	Fail	Fail
		CPU	—	—	—	—	—
	RGI	IT	89	230	481	741	1,005
	$\omega = 0.25$	CPU	0.0118	0.0299	0.0612	0.0968	0.1284
	MGI	IT	Fail	Fail	Fail	Fail	Fail
		CPU	—	—	—	—	—
	IMGI	IT	Fail	Fail	Fail	Fail	Fail
		CPU	—	—	—	—	—
	IMRGI	IT	4	189	439	689	938
	$\omega = 0.5$	CPU	6.2360×10^{-4}	0.0251	0.0587	0.0922	0.1005
$\mu = 1.7265 \times 10^{-6}$	GI	IT	Fail	Fail	Fail	Fail	Fail
		CPU	—	—	—	—	—
	OGI	IT	Fail	Fail	Fail	Fail	Fail
		CPU	—	—	—	—	—
	RGI	IT	15	356	769	1,186	1,608
	$\omega = 0.5$	CPU	0.0021	0.0450	0.0994	0.1139	0.2084
	MGI	IT	51	803	1998	3161	4356
		CPU	0.0071	0.1022	0.2652	0.3000	0.5491
	IMGI	IT	5	105	239	373	507
		CPU	8.0020×10^{-4}	0.0137	0.0321	0.0388	0.0645
	IMRGI	IT	7	394	923	1455	1986
	$\omega = 0.5$	CPU	0.0013	0.0549	0.1221	0.1413	0.2534
$\mu = 2.8983 \times 10^{-6}$	GI	IT	Fail	Fail	Fail	Fail	Fail
		CPU	—	—	—	—	—
	OGI	IT	Fail	Fail	Fail	Fail	Fail
		CPU	—	—	—	—	—
	RGI	IT	Fail	Fail	Fail	Fail	Fail
		CPU	—	—	—	—	—
	MGI	IT	Fail	Fail	Fail	Fail	Fail
		CPU	—	—	—	—	—
	IMGI	IT	14	105	306	397	605
		CPU	0.0019	0.0107	0.0386	0.0506	0.0765
	IMRGI	IT	5	238	554	871	1188
	$\omega = 0.5$	CPU	7.1740×10^{-4}	0.0245	0.0706	0.1114	0.1514

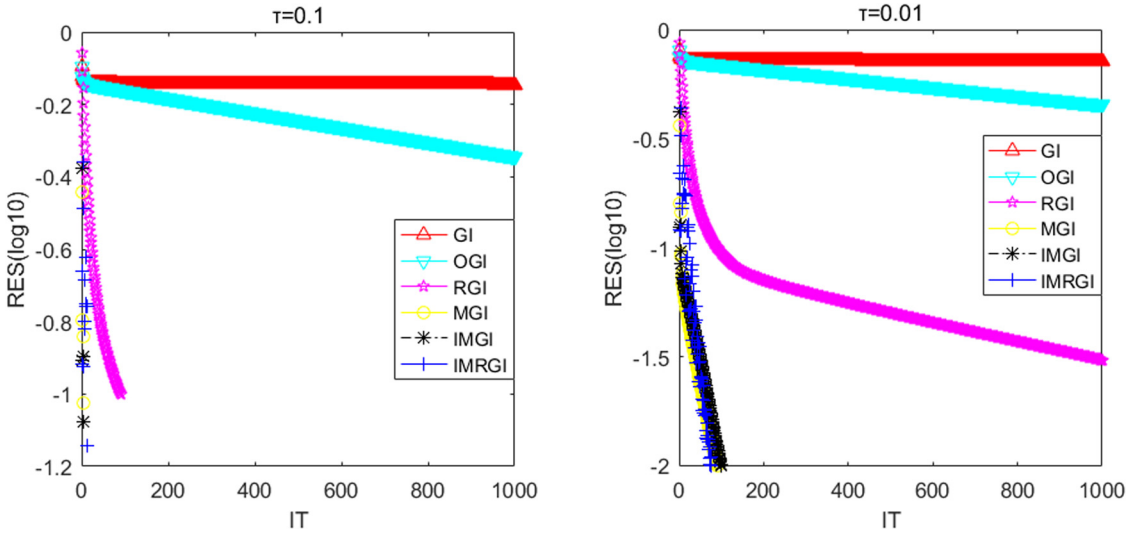

Figure 3: The logarithmic of the RES of the tested algorithms for Example 4.2 with $\tau = 0.1$ (left) and $\tau = 0.01$ (right).

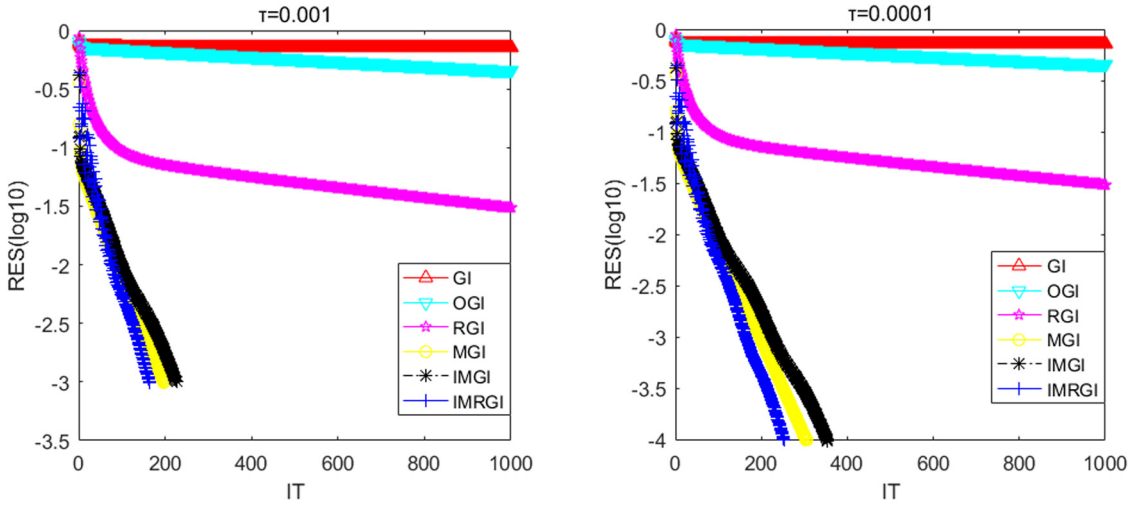

Figure 4: The logarithmic of the RES of the tested algorithms for Example 4.2 with $\tau = 0.001$ (left) and $\tau = 0.0001$ (right).

Table 7: Numerical results of the tested iterative algorithms for Example 4.3

Method	δ	0.1	0.01	0.001	0.0001	0.00001
	IT	27	138	489	841	1194
GI	CPU	0.0390	0.2016	0.7073	1.3760	1.6753
$\mu = 1.4041 \times 10^{-5}$	ERR	9.47×10^{-2}	1.00×10^{-2}	9.97×10^{-4}	9.99×10^{-5}	9.950×10^{-6}
	IT	24	115	205	296	387
OGI	CPU	0.0304	0.1610	0.3625	0.4663	0.6565
$\mu = 5.4000 \times 10^{-5}$	ERR	9.81×10^{-2}	9.80×10^{-3}	9.95×10^{-4}	9.89×10^{-5}	9.84×10^{-6}
	IT	24	115	205	296	387
RGI	CPU	0.0805	0.2041	0.3431	0.4895	0.8332
$\mu = 2.4300 \times 10^{-4}$	ERR	9.81×10^{-2}	9.80×10^{-3}	9.95×10^{-4}	9.89×10^{-5}	9.84×10^{-6}
	IT	11	77	189	306	425
MGI	CPU	0.0188	0.1115	0.2632	0.4301	0.6852
$\mu = 2.6145 \times 10^{-5}$	ERR	8.84×10^{-2}	9.80×10^{-3}	9.81×10^{-4}	9.95×10^{-5}	9.82×10^{-6}
	IT	47	99	132	194	221
IMGI	CPU	0.0052	0.0116	0.0141	0.0194	0.0954
$\mu = 3.3387 \times 10^{-5}$	ERR	9.93×10^{-2}	9.80×10^{-3}	9.53×10^{-4}	9.65×10^{-5}	9.51×10^{-6}
	IT	43	90	130	176	204
IMRGI	CPU	0.0054	0.0091	0.0125	0.0163	0.0178
$\omega = 1/3, \mu = 1.7233 \times 10^{-4}$	ERR	9.80×10^{-2}	9.40×10^{-3}	9.32×10^{-4}	9.88×10^{-5}	9.92×10^{-6}

Moreover, the IMRGI algorithm performs better than the IMGI one, and it is the most stable among all tested algorithms, because the change amplitude of the IT of the IMRGI algorithm is the lowest.

In Table 8, we list the same items as those in Tables 2 and 5. And we can obtain the similar conclusions from Table 8 as in Tables 2 and 5.

In addition, in Table 9, we list the same items as those in Tables 3 and 6. As observed in Table 9, besides the RGI and the IMRGI algorithms, the other tested ones are not convergent for $\mu = 9.0484 \times 10^{-5}$. Furthermore, the IMRGI algorithm has better convergence behaviors than the RGI one as $\delta \leq 0.001$, and the changing scope of the IT of the former one is smaller than the latter one. According to the results of $\mu = 5.5869 \times 10^{-5}$, we can assert that among the tested algorithms, the IMGI one is the best one as it requires the least IT and CPU times, and the IMRGI algorithm outperforms the RGI one as $\delta \leq 0.001$. Finally, by comparing the results of $\mu = 5.1405 \times 10^{-5}$, it can be seen that all tested algorithms are convergent except for the MGI one. And the IMGI algorithm is the most efficient as $\delta \leq 0.001$. Moreover, the GI and the OGI algorithms need less IT and CPU times than the RGI and the IMRGI ones, and the IMRGI algorithm is superior to the RGI one for most cases.

Table 8: The iterative solutions for IMGI ($\mu = 3.3387 \times 10^{-5}$) and IMRGI ($\omega = 1/3$, $\mu = 1.7233 \times 10^{-4}$) algorithms for Example 4.3

Method	k	x_{11}	x_{12}	x_{21}	x_{22}
IMGI	50	$-2.2638 + 5.3661i$	$2.9773 - 0.9517i$	$0.7559 + 0.1147i$	$-0.0281 + 3.0737i$
	100	$-1.9741 + 4.9711i$	$2.9953 - 0.9913i$	$0.9915 + 0.0320i$	$0.0200 + 2.9892i$
	150	$-2.0017 + 5.0022i$	$3.0003 - 1.0000i$	$0.9987 - 0.0012i$	$-0.0001 + 3.0001i$
	200	$-1.9999 + 4.9998i$	$2.9999 - 0.9999i$	$0.9999 + 0.0003i$	$0.0001 + 3.0000i$
	250	$-2.0000 + 5.0000i$	$3.0000 - 1.0000i$	$1.0000 - 0.0000i$	$0.0000 + 3.0000i$
IMRGI	50	$-2.1447 + 5.2224i$	$2.9300 - 0.9280i$	$0.8022 + 0.3178i$	$0.0028 + 3.0750i$
	100	$-1.9908 + 4.9890i$	$3.0031 - 0.9989i$	$0.9964 - 0.0069i$	$0.0093 + 2.9913i$
	150	$-2.0003 + 5.0006i$	$2.9997 - 0.9997i$	$0.9994 + 0.0015i$	$0.0001 + 3.0003i$
	200	$-2.0000 + 5.0000i$	$3.0000 - 1.0000i$	$1.0000 - 0.0000i$	$0.0000 + 3.0000i$
	250	$-2.0000 + 5.0000i$	$3.0000 - 1.0000i$	$1.0000 + 0.0000i$	$0.0000 + 3.0000i$
	Solution	$-2 + 5i$	$3 - i$	1	3i

Table 9: Numerical results of the tested iterative algorithms for Example 4.3 with three different values of μ

	Method	δ	0.1	0.01	0.001	0.0001	0.00001
$\mu = 9.0484 \times 10^{-5}$	GI	IT	Fail	Fail	Fail	Fail	Fail
		CPU	—	—	—	—	—
	OGI	IT	Fail	Fail	Fail	Fail	Fail
		CPU	—	—	—	—	—
	RGI	IT	17	86	304	522	740
	$\omega = 0.5$	CPU	0.0068	0.0084	0.0217	0.0474	0.0733
	MGI	IT	Fail	Fail	Fail	Fail	Fail
		CPU	—	—	—	—	—
	IMGI	IT	Fail	Fail	Fail	Fail	Fail
		CPU	—	—	—	—	—
	IMRGI	IT	68	144	205	282	322
	$\omega = 0.5$	CPU	0.0122	0.0111	0.0161	0.0223	0.0240
$\mu = 5.5869 \times 10^{-5}$	GI	IT	Fail	Fail	Fail	Fail	Fail
		CPU	—	—	—	—	—
	OGI	IT	Fail	Fail	Fail	Fail	Fail
		CPU	—	—	—	—	—
	RGI	IT	27	139	492	846	1,200
	$\omega = 0.5$	CPU	0.0028	0.0113	0.0453	0.0621	0.1513
	MGI	IT	Fail	Fail	Fail	Fail	Fail
		CPU	—	—	—	—	—
	IMGI	IT	30	62	89	120	145
		CPU	0.0031	0.0073	0.0099	0.0155	0.0347
	IMRGI	IT	108	231	328	451	519
	$\omega = 0.5$	CPU	0.0111	0.0229	0.0327	0.0464	0.0697
$\mu = 5.1405 \times 10^{-5}$	GI	IT	9	39	134	229	324
		CPU	0.0010	0.0036	0.0114	0.0185	0.0246
	OGI	IT	9	39	134	229	324
		CPU	9.5780×10^{-4}	0.0029	0.0103	0.0183	0.0249
	RGI	IT	29	150	534	919	1,305
	$\omega = 0.5$	CPU	0.0027	0.0119	0.0383	0.0682	0.1095
	MGI	IT	Fail	Fail	Fail	Fail	Fail
		CPU	—	—	—	—	—
	IMGI	IT	32	67	96	130	154
		CPU	0.0026	0.0057	0.0080	0.0103	0.0113
	IMRGI	IT	118	251	355	489	564
	$\omega = 0.5$	CPU	0.0110	0.0182	0.0252	0.0405	0.0396

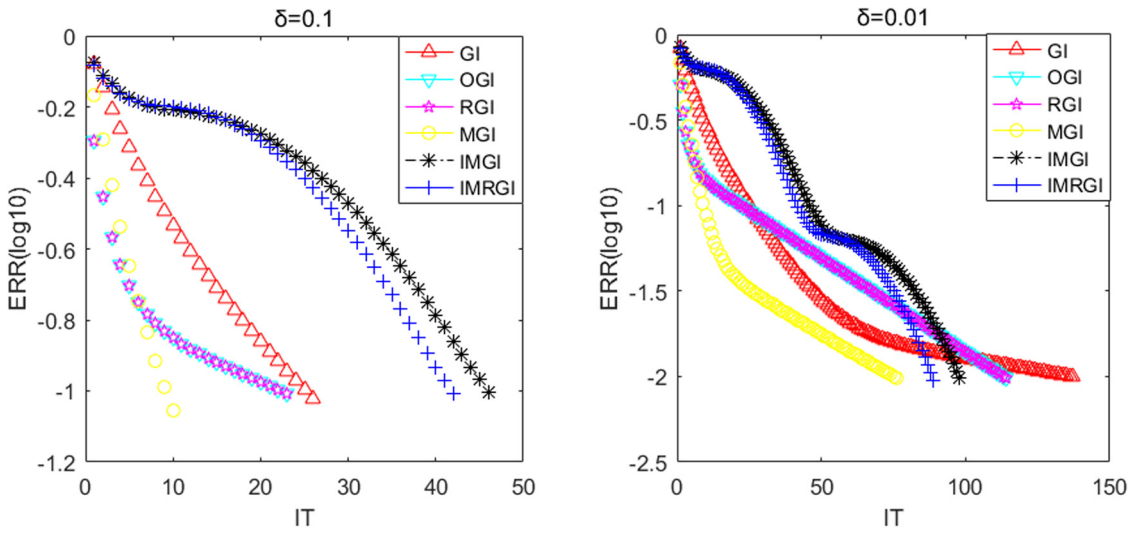

Figure 5: Comparison of convergence curves for Example 4.3 with $\delta = 0.1$ (left) and $\delta = 0.01$ (right).

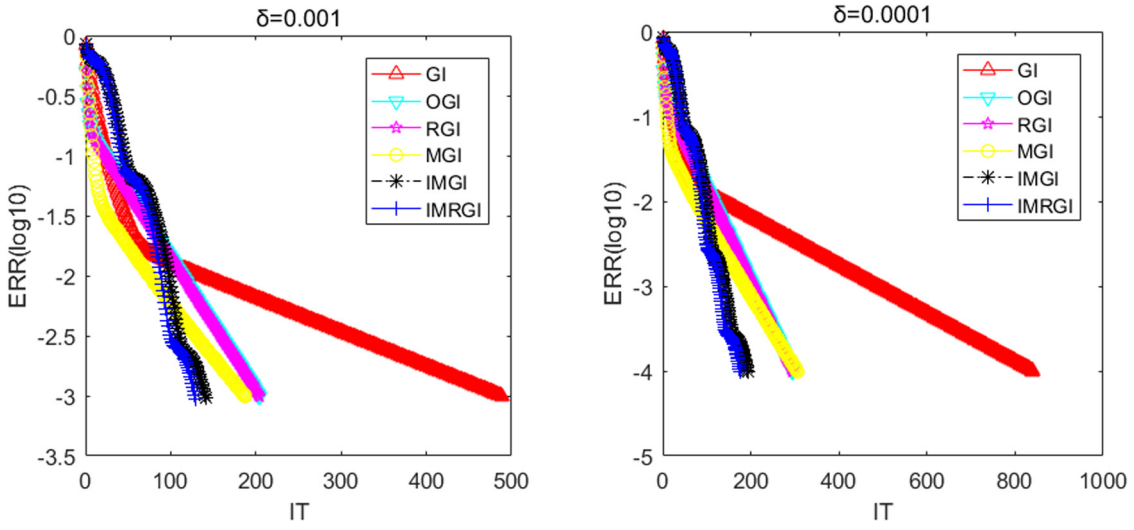

Figure 6: Comparison of convergence curves for Example 4.3 with $\delta = 0.001$ (left) and $\delta = 0.0001$ (right).

The graphs of ERR(log10) against the IT of the tested algorithms for four different values of δ are displayed in Figures 5 and 6, respectively. The two figures show that although the IMGI and the IMRGI algorithms are less efficient than the other ones when the value of δ is large, and they are more efficient than the other ones as the value of δ is small. This is in accordance with the results in Table 7. Besides, we can conclude that when the IT increases, the convergence rates of the GI, OGI, RGI, and MGI algorithms are fast first, but become slow later. And it is the opposite for the IMGI and the IMRGI algorithms.

Example 4.4. [18] Consider the CCT Sylvester matrix equation (2) with

$$A_1 = \begin{pmatrix} 1+2i & 2-i \\ 1-i & 2+3i \end{pmatrix}, \quad B_1 = \begin{pmatrix} 2-4i & i \\ -1+3i & 2 \end{pmatrix}, \quad A_2 \begin{pmatrix} -1-i & -3i \\ 0 & 1+2i \end{pmatrix},$$
$$B_2 = \begin{pmatrix} -2 & 1-i \\ 1+i & -1-i \end{pmatrix}, \quad A_3 = B_3 = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}, \quad A_4 = B_4 = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix},$$
$$H = \begin{pmatrix} 21+11i & -9+7i \\ 52-22i & -18+i \end{pmatrix}, \quad Z^* = \begin{pmatrix} 1+2i & -i \\ 2+i & -1+i \end{pmatrix}.$$

Table 10: Numerical results of the tested iterative algorithms for Example 4.4

Method	δ	0.1	0.01	0.001	0.0001	0.00001
	IT	53	255	726	1362	1,998
GI	CPU	0.0766	0.4319	1.1939	1.9604	2.7920
$\mu = 0.0028$	ERR	9.88×10^{-2}	9.90×10^{-2}	9.97×10^{-4}	9.97×10^{-5}	9.99×10^{-6}
	IT	46	269	499	729	959
OGI	CPU	0.0737	0.4681	0.8855	1.1386	1.4615
$\mu = 0.007607$	ERR	9.99×10^{-2}	1.00×10^{-2}	9.97×10^{-4}	9.98×10^{-5}	9.99×10^{-6}
	IT	46	269	499	729	959
RGI	CPU	0.0702	0.4569	0.8565	1.1687	1.4533
$\omega = 0.1, \mu = 0.08452$	ERR	9.99×10^{-2}	1.00×10^{-2}	9.97×10^{-4}	9.98×10^{-5}	9.99×10^{-6}
	IT	26	126	360	677	994
MGI	CPU	0.0433	0.2135	0.6016	1.2261	1.6008
$\mu = 0.0035$	ERR	9.84×10^{-2}	9.90×10^{-3}	9.97×10^{-4}	9.96×10^{-5}	9.95×10^{-6}
	IT	56	127	194	262	329
IMGI	CPU	0.0072	0.0158	0.0189	0.0235	0.0332
$\mu = 0.0077$	ERR	9.85×10^{-2}	9.70×10^{-3}	9.90×10^{-4}	9.69×10^{-5}	9.80×10^{-6}
	IT	36	80	122	164	205
IMRGI	CPU	0.0037	0.0071	0.0129	0.0175	0.0171
$\omega = 0.1, \mu = 0.1538$	ERR	9.77×10^{-2}	9.60×10^{-3}	9.60×10^{-4}	9.53×10^{-5}	9.98×10^{-6}

Let $\delta > 0$, we set

$$\text{ERR} = \frac{||Z(k) - Z^*||}{||Z^*||} \leq \delta$$

to be the termination condition. And the prescribed maximum iterative number is 20,000 as in Examples 4.1–4.3.

For Example 4.4, we compare the IMGI and the IMRGI algorithms with the GI, OGI, RGI, and MGI ones in Table 10. The parameters adopted in the tested algorithms are obtained by utilizing the methods in Example 4.1. From the results listed in Table 10, we can conclude some observations: First, the IT of the OGI and the RGI algorithms are the same, and they perform better than the GI one. Second, the MGI algorithm outperforms the GI, OGI, and RGI ones except for the case of $\delta = 0.00001$. Third, the IMGI algorithm has faster convergence speed than the GI, OGI, RGI, and MGI ones as $\delta \leq 0.001$. Fourth, the IMRGI algorithm performs the best in terms

Table 11: The iterative solutions for IMGI ($\mu = 0.0077$) and IMRGI ($\omega = 0.1, \mu = 0.1538$) algorithms for Example 4.4

Method	k	x_{11}	x_{12}	x_{21}	x_{22}
IMGI	100	$1.0155 + 2.0425i$	$0.0054 - 0.9530i$	$1.9630 + 0.9980i$	$-1.0365 + 0.9909i$
	200	$1.0005 + 2.0014i$	$0.0002 - 0.9984i$	$1.9987 + 0.9999i$	$-1.0012 + 0.9997i$
	300	$1.0000 + 2.0000i$	$0.0000 - 0.9999i$	$2.0000 + 1.0000i$	$-1.0000 + 1.0000i$
	400	$1.0000 + 2.0000i$	$0.0000 - 1.0000i$	$2.0000 + 1.0000i$	$-1.0000 + 1.0000i$
	500	$1.0000 + 2.0000i$	$0.0000 - 1.0000i$	$2.0000 + 1.0000i$	$-1.0000 + 1.0000i$
IMRGI	100	$1.0021 + 2.0055i$	$0.0008 - 0.9939i$	$1.9951 + 0.9997i$	$-1.0048 + 0.9988i$
	200	$1.0000 + 2.0000i$	$0.0000 - 1.0000i$	$2.0000 + 1.0000i$	$-1.0000 + 1.0000i$
	300	$1.0000 + 2.0000i$	$0.0000 - 1.0000i$	$2.0000 + 1.0000i$	$-1.0000 + 1.0000i$
	400	$1.0000 + 2.0000i$	$0.0000 - 1.0000i$	$2.0000 + 1.0000i$	$-1.0000 + 1.0000i$
	500	$1.0000 + 2.0000i$	$0.0000 - 1.0000i$	$2.0000 + 1.0000i$	$-1.0000 + 1.0000i$
	Solution	$1 + 2i$	$-i$	$2 + i$	$-1 + i$

Table 12: Numerical results of the tested iterative algorithms for Example 4.4 with three different values of μ

	Method	δ	0.1	0.01	0.001	0.0001	0.00001
μ = 0.0433	GI	IT	Fail	Fail	Fail	Fail	Fail
		CPU	—	—	—	—	—
	OGI	IT	Fail	Fail	Fail	Fail	Fail
		CPU	—	—	—	—	—
	RGI	IT	Fail	Fail	Fail	Fail	Fail
		CPU	—	—	—	—	—
	MGI	IT	Fail	Fail	Fail	Fail	Fail
		CPU	—	—	—	—	—
	IMGI	IT	Fail	Fail	Fail	Fail	Fail
		CPU	—	—	—	—	—
	IMRGI	IT	40	90	138	186	233
	ω = 0.5	CPU	0.0053	0.0169	0.0156	0.0195	0.0242
μ = 0.0197	GI	IT	Fail	Fail	Fail	Fail	Fail
		CPU	—	—	—	—	—
	OGI	IT	Fail	Fail	Fail	Fail	Fail
		CPU	—	—	—	—	—
	RGI	IT	30	143	408	764	1121
	ω = 0.5	CPU	0.0034	0.0156	0.0492	0.0803	0.1694
	MGI	IT	Fail	Fail	Fail	Fail	Fail
		CPU	—	—	—	—	—
	IMGI	IT	Fail	Fail	Fail	Fail	Fail
		CPU	—	—	—	—	—
	IMRGI	IT	87	198	304	410	515
	ω = 0.5	CPU	0.0109	0.0204	0.0444	0.0477	0.0750
μ = 0.0269	GI	IT	Fail	Fail	Fail	Fail	Fail
		CPU	—	—	—	—	—
	OGI	IT	Fail	Fail	Fail	Fail	Fail
		CPU	—	—	—	—	—
	RGI	IT	22	105	298	558	819
	ω = 0.5	CPU	0.0024	0.0107	0.0297	0.0555	0.0795
	MGI	IT	Fail	Fail	Fail	Fail	Fail
		CPU	—	—	—	—	—
	IMGI	IT	Fail	Fail	Fail	Fail	Fail
		CPU	—	—	—	—	—
	IMRGI	IT	64	145	222	299	376
	ω = 0.5	CPU	0.0069	0.0150	0.0229	0.0312	0.0384

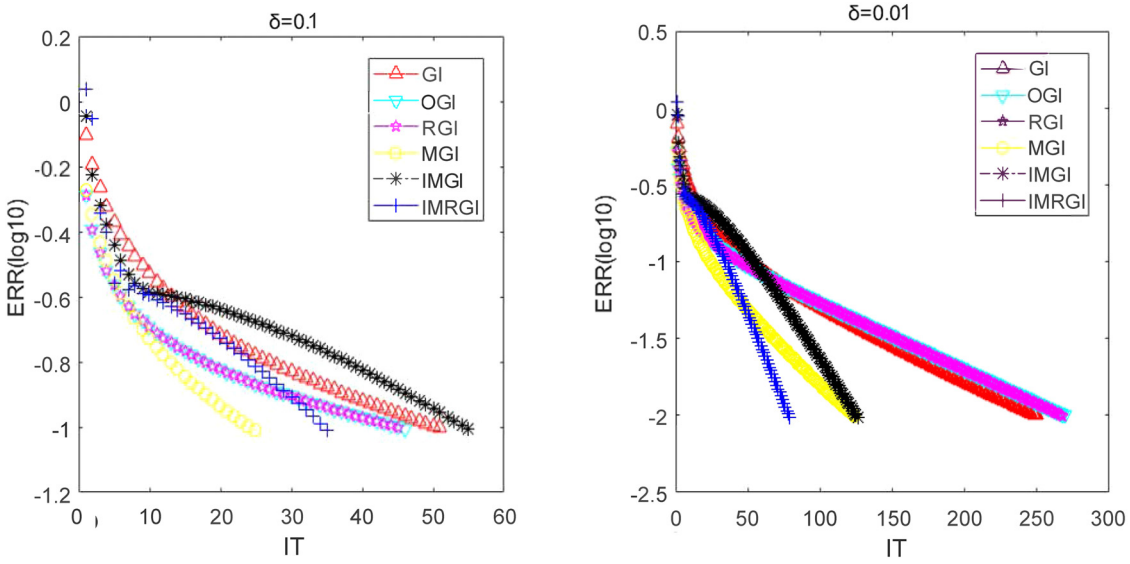

Figure 7: Comparison of convergence curves for Example 4.4 with δ = 0.1 (left) and δ = 0.01 (right).

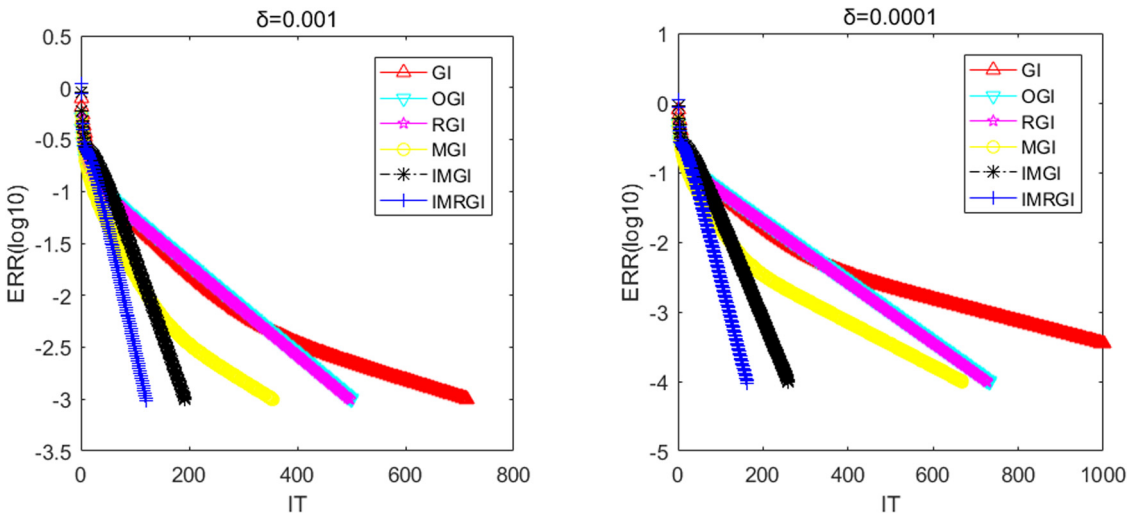

Figure 8: Comparison of convergence curves for Example 4.4 with $\delta = 0.001$ (left) and $\delta = 0.0001$ (right).

of IT and CPU times except for the case of $\delta = 0.1$. Finally, the proposed IMGI and IMRGI algorithms are more stable than the other ones, because the changing scopes of the IT of the former ones are less than those of the latter ones with the decreasing of δ, and the IMRGI algorithm is the most stable for this example.

In Table 11, the iterative solutions $\{Z(k)\}$ generated by the IMGI and the IMRGI algorithms are listed. We can see from Table 11 that with the increasing of IT, the iterative solutions $\{Z(k)\}$ are gradually tended to the exact solution Z^*.

Furthermore, the same items as those in Tables 3, 6 and 9 are exhibited in Table 12. From the results of $\mu = 0.0433$, we conclude that the GI, OGI, RGI, MGI, and IMGI algorithms cannot achieve the stopping criterion within the largest admissible number of iteration steps, while the IMRGI one is valid. Meanwhile, for $\mu = 0.0197$, the proposed IMRGI algorithm has better numerical performances than the RGI one when $\delta \leq 0.001$, and the former one is more stable than the latter one in view of the variation range of IT. Finally, for the case of $\mu = 0.0269$, we can obtain the similar conclusions as the case of $\mu = 0.0197$. All in all, the techniques utilized in the IMGI and the IMRGI algorithms can improve the convergence rates of the GI, OGI, RGI, and MGI ones from the point of view of computing efficiency.

In Figures 7 and 8, we depict the error curves of the tested algorithms for four different values of δ. Figures 7 and 8 clearly show that the convergence rates of the IMRGI and the IMGI algorithms are faster than the other ones for most cases. This reveals that the proposed algorithms are efficient and feasible for solving the CCT Sylvester matrix equations.

5 Conclusions

In this article, by replacing the coefficient matrices with their diagonal parts and making use of the updated technique, we first construct the IMGI algorithm for the CCT Sylvester matrix equations. Then we further apply the relaxation technique to the IMGI algorithm and establish the IMRGI algorithm. By making use of the real representation of a complex matrix and the vector stretching operator, we give the convergence conditions of the IMGI and the IMRGI algorithms. Some numerical examples are presented to validate the effectiveness and superiorities of the IMGI and the IMRGI algorithms. Compared with the GI, OGI, RGI, and MGI algorithms, the proposed IMGI and IMRGI algorithms can own higher computational efficiencies by adjusting the values of the relaxation factors and the step size factors. Then it is more conducive to solving Sylvester matrix equations in pole assignment, control theory, signal processes, prediction and stability, and so forth.

It is noteworthy that only one step size factor μ is used in the IMGI and the IMRGI algorithms. We will consider to take different step size factors into the IMGI and the IMRGI algorithms and propose their convergence analyses. Aside from that, we have not deduced the optimal parameters of the IMGI and the IMRGI algorithms in theory at present, which deserves to be investigated in the future.

Acknowledgements: We thank the editor and anonymous reviewers for their careful reading of the manuscript and for the helpful comments.

Funding information: This work was supported by the National Natural Science Foundation of China (No. 12361078), and the Guangxi Natural Science Foundation (No. 2024GXNSFAA010498).

Author contributions: All authors contributed to the study conception and design. All authors performed material preparation, data collection, and analysis. The authors read and approved the final manuscript.

Conflict of interest: The authors declare that they have no competing interests.

Ethical approval: The conducted research is not related to either human or animal use.

Data availability statement: Data sharing is not applicable to this article as no datasets were generated or analyzed during this study.

References

[1] F. P. A. Beik, D. K. Salkuyeh, and M. M. Moghadam, *Gradient-based iterative algorithm for solving the generalized coupled Sylvester-transpose and conjugate matrix equations over reflexive (anti-reflexive) matrices*, Trans. Inst. Meas. Control. **36** (2014), 99–110, DOI: https://doi.org/10.1177/0142331213482485.

[2] B. Zhou, L.-L. Lv, and G.-R. Duan, *Parametric pole assignment and robust pole assignment for discrete-time linear periodic systems*, SIAM J. Control Optimi. **48** (2010), 3975–3996, DOI: https://doi.org/10.1137/080730469.

[3] Y.-F. Cai, J. Qian, and S.-F. Xu, *Robust partial pole assignment problem for high order control systems*, Automatica **48** (2012), 1462–1466, DOI: https://doi.org/10.1016/j.automatica.2012.05.015.

[4] Z.-B. Chen and X.-S. Chen, *Conjugate gradient-based iterative algorithm for solving generalized coupled Sylvester matrix equations*, J. Frank. Inst. **359** (2022), 9925–9951, DOI: https://doi.org/10.1016/j.jfranklin.2022.09.049.

[5] Z.-B. Chen and X.-S. Chen, *Modification on the convergence results of the Sylvester matrix equation* $AX + XB = C$, J. Frank. Inst. **359** (2022), 3126–3147, DOI: https://doi.org/10.1016/j.jfranklin.2022.02.021.

[6] M. Hajarian, *Efficient iterative solutions to general coupled matrix equations*, Int. J. Autom. Comput. **10** (2013), 418–486, DOI: https://doi.org/10.1007/s11633-013-0745-6.

[7] M. Hajarian, *Solving the general Sylvester discrete-time periodic matrix equations via the gradient-based iterative method*, Appl. Math. Lett. **52** (2016), 87–95, DOI: https://doi.org/10.1016/j.aml.2015.08.017.

[8] F. Ding and T.-W. Chen, *Gradient based iterative algorithms for solving a class of matrix equations*, IEEE Trans. Automat. Contr. **50** (2005), 1216–1221, DOI: https://doi.org/10.1109/TAC.2005.852558.

[9] F. Ding, P.-X. Liu, and J. Ding, *Iterative solutions of the generalized Sylvester matrix equations by using the hierarchical identification principle*, Appl. Math. Comput. **197** (2008), 41–50, DOI: https://doi.org/10.1016/j.amc.2007.07.040.

[10] L. Xie, Y.-J. Liu, and H.-Z. Yang, *Gradient based and least squares based iterative algorithms for matrix equations* $AXB + CX^TD = F$, Appl. Math. Comput. **217** (2010), 2191–2199, DOI: https://doi.org/10.1016/j.amc.2010.07.019.

[11] C.-Q. Song, G.-L. Chen, and L.-L. Zhao, *Iterative solutions to coupled Sylvester-transpose matrix equations*, Appl. Math. Model. **35** (2011), 4675–4683, DOI: https://doi.org/10.1016/j.apm.2011.03.038.

[12] A.-G. Wu, X. Zeng, G.-R. Duan, and W.-J. Wu, *Iterative solutions to the extended Sylvester-conjugate matrix equation*, Appl. Math. Comput. **217** (2010), 4427–4438, DOI: https://doi.org/10.1016/j.amc.2010.05.029.

[13] A.-G. Wu, G. Feng, G.-R. Duan, and W.-J. Wu, *Iterative solutions to coupled Sylvester-conjugate matrix equations*, Comput. Math. Appl. **60** (2010), 54–66, DOI: https://doi.org/10.1016/j.camwa.2010.04.029.

[14] A.-G. Wu, L.-L. Lv, and G.-R. Duan, *Iterative algorithms for solving a class of complex conjugate and transpose matrix equations*, Appl. Math. Comput. **217** (2011), 8343–8353, DOI: https://doi.org/10.1016/j.amc.2011.02.113.

[15] X.-P. Sheng, *A relaxed gradient-based algorithm for solving generalized coupled Sylvester matrix equations*, J. Frank. Inst. **355** (2018), 4282–4297, DOI: https://doi.org/10.1016/j.jfranklin.2018.04.008.
[16] B.-H. Huang and C.-F. Ma, *The relaxed gradient-based iterative algorithms for a class of generalized coupled Sylvester-conjugate matrix equations*, J. Frank. Inst. **355** (2018), 3168–3195, DOI: https://doi.org/10.1016/j.jfranklin.2018.02.014.
[17] Q. Niu, X. Wang, and L.-Z. Lu, *A relaxed gradient-based algorithm for solving Sylvester equations*, Asian J. Control **13** (2011), 461–464, DOI: https://doi.org/10.1002/asjc.328.
[18] M. A. Ramadan and A. M. E. Bayoumi, *A modified gradient-based algorithm for solving extended Sylvester-conjugate matrix equations*, Asian J. Control **20** (2018), 228–235, DOI: https://doi.org/10.1002/asjc.1574.
[19] Y.-J. Xie and C.-F. Ma, *The accelerated gradient-based iterative algorithm for solving a class of generalized Sylvester-transpose matrix equation*, Appl. Math. Comput. **273** (2016), 1257–1269, DOI: https://doi.org/10.1016/j.amc.2015.07.022.
[20] X. Wang, L. Dai, and D. Liao, *A modified gradient-based algorithm for solving Sylvester equations*, Appl. Math. Comput. **218** (2012), 5620–5628, DOI: https://doi.org/10.1016/j.amc.2011.11.055.
[21] W.-L. Wang, C.-Q. Song, and S.-P. Ji, *Iterative solution to a class of complex matrix equations and its application in time-varying linear system*, J. Appl. Math. Comput. **67** (2021), 317–341, DOI: https://doi.org/10.1007/s12190-020-01486-6.
[22] B.-H. Huang and C.-F. Ma, *On the relaxed gradient-based iterative methods for the generalized coupled Sylvester-transpose matrix equations*, J. Frank. Inst. **359** (2022), 10688–10725, DOI: https://doi.org/10.1016/j.jfranklin.2022.07.051.
[23] Z.-L. Tian, M.-Y. Tian, C.-Q. Gu, and X.-N. Hao, *An accelerated Jacobi-gradient-based iterative algorithm for solving Sylvester matrix equations*, Filomat **31** (2017), 2381–2390, DOI: https://doi.org/10.2298/FIL1708381T.
[24] W.-L. Wang and C.-Q. Song, *Iterative algorithms for discrete-time periodic Sylvester matrix equations and its application in antilinear periodic system*, Appl. Numer. Math. **168** (2021), 251–273, DOI: https://doi.org/10.1016/j.apnum.2021.06.006.
[25] S.-K. Li, *A finite iterative method for solving the generalized Hamiltonian solutions of coupled Sylvester matrix equations with conjugate transpose*, Int. J. Comput. Math. **94** (2017), 757–773, DOI: https://doi.org/10.1080/00207160.2016.1148810.
[26] H.-M. Zhang, *A finite iterative algorithm for solving the complex generalized coupled Sylvester matrix equations by using the linear operators*, J. Frank. Inst. **354** (2017), 1856–1874, DOI: https://doi.org/10.1016/j.jfranklin.2016.12.011.
[27] T.-X. Yan and C.-F. Ma, *The BCR algorithm for solving the reflexive or anti-reflexive solutions of generalized coupled Sylvester matrix equations*, J. Frank. Inst. **357** (2020), 12787–12807, DOI: https://doi.org/10.1016/j.jfranklin.2020.09.030.
[28] T.-X. Yan and C.-F. Ma, *An iterative algorithm for generalized Hamiltonian solution of a class of generalized coupled Sylvester-conjugate matrix equations*, Appl. Math. Comput. **411** (2021), 126491, DOI: https://doi.org/10.1016/j.amc.2021.126491.
[29] C.-F. Ma and T.-X. Yan, *A finite iterative algorithm for the general discrete-time periodic Sylvester matrix equations*, J. Frank. Inst. **359** (2022), 4410–4432, DOI: https://doi.org/10.1016/j.jfranklin.2022.03.047.
[30] H.-M. Zhang and H.-C. Yin, *New proof of the gradient-based iterative algorithm for a complex conjugate and transpose matrix equation*, J. Frank. Inst. **354** (2017), 7585–7603, DOI: https://doi.org/10.1016/j.jfranklin.2017.09.005.
[31] W.-P. Hu and W.-G. Wang, *Improved gradient iteration algorithms for solving the coupled Sylvester matrix equation*, J. Nanjing Univ. Math. Biq. **33** (2016), 177–192, DOI: https://www.cnki.com.cn/Article/CJFDTotal-SXXT201602006.htm.
[32] A.-G. Wu, Y. Zhang, and Y.-Y. Qian, *Complex Conjugate Matrix Equations*, Science Press, Beijing, 2017.
[33] J.-J. Hu, Y.-F. Ke, and C.-F. Ma, *Generalized conjugate direction algorithm for solving generalized coupled Sylvester transpose matrix equations over reflexive or anti-reflexive matrices*, J. Frank. Inst. **359** (2022), 6958–6985, DOI: https://doi.org/10.1016/j.jfranklin.2022.07.005.
[34] A. Berman and R. J. Plemmons, *Nonnegative Matrices in the Mathematical Sciences*, SIAM, Philadelphia, 1994.