Modified Jacobi-Gradient Iterative Method for Generalized Sylvester Matrix Equation

Nopparut Sasaki and Pattrawut Chansangiam *

Department of Mathematics, Faculty of Science, King Mongkut’s Institute of Technology Ladkrabang, Bangkok 10520, Thailand; 61605027@kmitl.ac.th

* Correspondence: pattrawut.ch@kmitl.ac.th; Tel.: +66-935-266600

Received: 14 October 2020; Accepted: 2 November 2020; Published: 5 November 2020

Abstract: We propose a new iterative method for solving a generalized Sylvester matrix equation $A_1X A_2 + A_3X A_4 = E$ with given square matrices $A_1, A_2, A_3, A_4$ and an unknown rectangular matrix $X$. The method aims to construct a sequence of approximated solutions converging to the exact solution, no matter the initial value is. We decompose the coefficient matrices to be the sum of its diagonal part and others. The recursive formula for the iteration is derived from the gradients of quadratic norm-error functions, together with the hierarchical identification principle. We find equivalent conditions on a convergent factor, relied on eigenvalues of the associated iteration matrix, so that the method is applicable as desired. The convergence rate and error estimation of the method are governed by the spectral norm of the related iteration matrix. Furthermore, we illustrate numerical examples of the proposed method to show its capability and efficacy, compared to recent gradient-based iterative methods.

Keywords: generalized Sylvester matrix equation; iterative method; gradient; Kronecker product; matrix norm

MSC: 65F45; 15A12; 15A60; 15A69

1. Introduction

In control engineering, certain problems concerning the analysis and design of control systems can be formulated as the Sylvester matrix equation:

$$A_1X + XA_2 = C$$  \hspace{1cm} (1)

where $X \in \mathbb{R}^{m \times n}$ is an unknown matrix, and $A_1, A_2, C$ are known matrices of appropriate dimensions. Here, $\mathbb{R}^{m \times n}$ stands for the set of $m \times n$ real matrices. Let us denote by $(\cdot)^T$ the transpose of a matrix. When $A_2 = A_1^T$, the equation is reduced to the Lyapunov equation, which is often found in continuous- and discrete-time stability analysis [1,2]. The Sylvester equation is a special case of a generalized Sylvester matrix equation:

$$A_1X A_2 + A_3X A_4 = E$$  \hspace{1cm} (2)

where $X \in \mathbb{R}^{m \times n}$ is unknown, and $A_1, A_2, A_3, A_4, E$ are known constant matrices of appropriate dimensions. This equation also includes the equation $A_1X A_2 = C$ and the Kalman–Yakubovich equation $A_1X A_2 - X = C$ as special cases. All of these equations have profound applications in linear system theory and related areas.

Normally, a direct way to solve the generalized Sylvester Equation (2) is to reduce it to a linear system by taking the vector operator. Then, Equation (2) reduces to $Px = b$ where
\[ P = A_2^T \otimes A_1 + A_4^T \otimes A_3, \quad x = \text{vec}(X), \quad \text{and} \quad b = \text{vec}(E). \]

Here, vec(\cdot) is the vector operator and the operation \( \otimes \) is the Kronecker multiplication. So, Equation (2) has a unique solution if and only if the square matrix \( P \) is invertible. However, it is not easy to compute \( P^{-1} \) when the sizes of \( A_1, A_2, A_3, A_4 \) are not small, since the size of \( P \) can be very large. Such a size problem leads to computation difficulty in that excessive computer memory is required for the inversion of large matrices. Thus, another way exists which transform the matrix coefficients into a Schur or Hessenberg form, for which solutions may be readily computed—see [3,4].

For matrix equations of large dimensions, iterative algorithms to find an approximated/exact solution are remarkable. There are many techniques to construct an iterative procedure for Equation (2) or its special cases—e.g., matrix sign function [5], block successive over-relaxation [6], block recursion [7,8], Krylov subspace [9,10], and truncated low-rank algorithm [11]. Lately, there have been some variants of Hermitian and skew-Hermitian splitting—e.g., a generalized modified Hermitian and skew-Hermitian splitting algorithm [12], accelerated double-step scale splitting algorithm [13], PHSS algorithm [14], and the four parameter PSS algorithm [15]. Furthermore, the idea of conjugate gradient leads to finite-step iterative methods to find the exact solution such as the generalized conjugated direction algorithm [16], the conjugated gradient least-squares algorithm [17], and generalized product-type methods based on the Bi-conjugate gradient algorithm [18].

In the last decade, many authors have developed gradient-based iterative (GI) algorithms for certain linear matrix equations that satisfy the asymptotic stability (AS) in the following meaning:

**AS:** The sequence of approximated solutions converges to the exact solution, no matter the initial value is.

The first GI algorithm for solving (1) was developed by Ding and Chen [19]. In that paper, a sufficient condition in terms of a convergence factor is determined so that the algorithm satisfies (AS) property. By introducing of a relaxation parameter, Niu et al. [20] suggested a relaxed gradient iterative (RGI) gradient algorithm to overcome (1). Numerical studies show that when the relaxation factor is correctly selected the convergent behavior of the Niu’s algorithm is stronger than the Ding’s algorithm. Zhang and Sheng [21] introduced an RGI algorithm for finding the symmetric (skew-symmetric) solution of Equation (1). Xie et al. [22] improved the RGI algorithm to become an accelerated GI (AGBI) algorithm, on the basis of the information generated in the previous half-step and a relaxation factor. Ding and Chen [23] also applied the ideas of gradients and least-squares to formulate the least-squares iterative (LSI) algorithm. In [24], Fan et al. realized that multiplication of the matrix in GI would take great time and space if \( A_1 \) and \( A_2 \) were big and dense, so they proposed the following Jacobi-gradient iterative (JGI) method.

**Method 1** (Jacobi-Gradient based Iterative (JGI) algorithm [24]). For \( i = 1, 2 \), let \( D_i \) be the diagonal part of \( A_i \). Given any initial matrices \( X_1(0), X_2(0) \). Set \( k = 0 \) and compute \( X(0) = (1/2)(X_1(0) + X_2(0)) \). For \( k = 1, 2, \ldots, \End, \) do:

\[
X_1(k) = X(k-1) + \mu D_1[C - A_1 X(k-1) - X(k-1) A_2],
\]

\[
X_2(k) = X(k-1) + \mu [C - A_1 X(k-1) - X(k-1) A_2] D_2,
\]

\[
X(k) = \frac{1}{2} (X_1(k) + X_2(k)).
\]

After that, Tian et al. [25] proposed that an accelerated Jacobi-gradient iterative (AJGI) algorithm for solving the Sylvester matrix equation relies on two relaxation factors and the half-step update. However, the parameter values to apply to the algorithm are difficult to find since they are given in terms of a nonlinear inequality. For the generalized Sylvester Equation (2), the gradient iterative (GI) algorithm [19] and the least-squares iterative (LSI) algorithm [26] were established as follows.
Method 2 (GI algorithm [19]). Given any two initial matrices \(X_1(0), X_2(0)\). Set \(k = 0\) and compute \(X(0) = (1/2)(X_1(0) + X_2(0))\). For \(k = 1, 2, \ldots, \text{End, do:}\)

\[
\begin{align*}
X_1(k) &= X(k - 1) + \mu A_1^T [E - A_1 X(k - 1) A_2 - A_3 X(k - 1) A_4] A_2^T, \\
X_2(k) &= X(k - 1) + \mu A_3^T [E - A_1 X(k - 1) A_2 - A_3 X(k - 1) A_4] A_4^T, \\
X(k) &= \frac{1}{2}(X_1(k) + X_2(k)).
\end{align*}
\]

A sufficient condition for which the algorithm satisfies (AS) is

\[
0 < \mu < \frac{2}{\|A_1\|_2^2 \|A_2\|_2^2 + \|A_3\|_2^2 \|A_4\|_2^2}.
\]

Method 3 (LSI algorithm [26]). Given any two initial matrices \(X_1(0), X_2(0)\). Set \(k = 0\) and compute \(X(0) = \frac{1}{2}(X_1(0) + X_2(0))\). For \(k = 1, 2, \ldots, \text{End, do:}\)

\[
\begin{align*}
X_1(k) &= X(k - 1) + \mu (A_1^T A_1)^{-1} A_1^T [E - A_1 X(k - 1) A_2 - A_3 X(k - 1) A_4] A_2^T (A_2 A_2^T)^{-1}, \\
X_2(k) &= X(k - 1) + \mu (A_3^T A_3)^{-1} A_3^T [E - A_1 X(k - 1) A_2 - A_3 X(k - 1) A_4] A_4^T (A_4 A_4^T)^{-1}, \\
X(k) &= \frac{1}{2}(X_1(k) + X_2(k)).
\end{align*}
\]

If \(0 < \mu < 4\), then the algorithm satisfies (AS).

In this paper, we shall propose a new iterative method for solving the generalized Sylvester matrix Equation (2), when \(A_1, A_3 \in \mathbb{R}^{m \times m}, A_2, A_4 \in \mathbb{R}^{n \times n}\) and \(X, E \in \mathbb{R}^{m \times n}\). This algorithm requires only one initial value \(X(0)\) and only one parameter, called a convergence factor. We decompose the coefficient matrices to be the sum of its diagonal part and others. The recursive formula for iteration is derived from the gradients of quadratic norm-error functions together with hierarchical identification principle. Under assumptions on the real-parts sign of eigenvalues of matrix coefficients, we find necessary and sufficient conditions on a convergent factor for which (AS) holds. The convergence rate and error estimates are regulated by the iteration matrix spectral radius. In particular, when the iteration matrix is symmetric, we obtain a convergence criteria, error estimates and the optimal convergent factor in terms of spectral norms and condition number. Moreover, numerical simulations are also provided to illustrate our results for (2) and (1). We compare the efficiency of our algorithm to LSI, GI, RGI, AGBI and JGI algorithms.

Let us recall some terminology from matrix analysis—see e.g., [27]. For any square matrix \(X\), denote \(\sigma(X)\) its spectrum, \(\rho(X)\) its spectral radius, and \(\text{tr}(X)\) its trace. Let us denote the largest and the smallest eigenvalues of a matrix by \(\lambda_{\max}(\cdot)\) and \(\lambda_{\min}(\cdot)\), respectively. Recall that the spectral norm \(\|\cdot\|_2\) and the Frobenius norm \(\|\cdot\|_F\) of \(A \in \mathbb{R}^{m \times n}\) are, respectively, defined by

\[
\|A\|_2 = \sqrt{\lambda_{\max}(A^T A)} \quad \text{and} \quad \|A\|_F = \sqrt{\text{tr}(A^T A)}.
\]

The condition number of \(A \neq 0\) is defined by

\[
\kappa(A) = \frac{\lambda_{\max}(A^T A)}{\lambda_{\min}(A^T A)}.
\]

Denote the real part of a complex number \(z\) by \(\Re(z)\).

The rest of paper is organized as follows. We propose a modified Jacobi-gradient iterative algorithm in Section 2. Convergence criteria, convergence rate, error estimates, and optimal convergence factor are discussed in Section 3. In Section 4, we provide numerical simulations of the algorithm. Finally, we conclude the paper in Section 5.
2. A Modified Jacobi-Gradient Iterative Method for the Generalized Sylvester Equation

In this section, we propose an iterative algorithm for solving the generalized Sylvester equation, called a modified Jacobi-gradient iterative algorithm.

Throughout, let \( m, n \in \mathbb{N} \) and \( A_1, A_3 \in \mathbb{R}^{m \times m}, A_2, A_4 \in \mathbb{R}^{n \times n} \) and \( E \in \mathbb{R}^{m \times n} \). We would like to find a matrix \( X \in \mathbb{R}^{m \times n} \), such that

\[
A_1XA_2 + A_3XA_4 = E. \tag{3}
\]

Write \( A_1 = D_1 + F_1, A_2 = D_2 + F_2, A_3 = D_3 + F_3 \) and \( A_4 = D_4 + F_4 \), where \( D_1, D_2, D_3, D_4 \) are the diagonal parts of \( A_1, A_2, A_3, A_4 \), respectively. A necessary and sufficient condition for (3) to have a unique solution is the invertibility of the square matrix

\[
P := A_2^T \otimes A_1 + A_4^T \otimes A_3.
\]

In this case, the solution is given by \( \text{vec } X = P^{-1} \text{vec } E \).

To obtain an iterative algorithm for solving (3), we recall the hierarchical identification principle in [19]. We write (3) to

\[
(D_1 + F_1)XD_2 + A_3XA_4 = E, \tag{4}
\]

\[
A_1XA_2 + (D_3 + F_3)XD_4 = E. \tag{5}
\]

Define two matrices

\[
M := E - F_1XD_2 - D_1XF_2 - F_1XF_2 - A_3XA_4,
\]

\[
N := E - F_3XD_4 - D_3XF_4 - F_3XF_4 - A_1XA_2.
\]

From (4) and (5), we shall find the approximated solution of the following two subsystems

\[
D_1XD_2 = M \quad \text{and} \quad D_3XD_4 = N \tag{6}
\]

so that the following norm-error functions are minimized:

\[
L_1(X) := \|D_1XD_2 - M\|_F^2 \quad \text{and} \quad L_2(X) := \|D_3XD_4 - N\|_F^2. \tag{7}
\]

From the gradient formula

\[
\frac{d}{dX} \text{tr}(AX) = A^T,
\]

we can deduce the gradient of the error \( L_1 \) as follows:

\[
\frac{\partial}{\partial X} L_1(X) = \frac{\partial}{\partial X} \text{tr} \left[ (D_1XD_2 - M)^T(D_1XD_2 - M) \right]
+ \frac{\partial}{\partial X} \text{tr}(XD_2D_2X^T) - \frac{\partial}{\partial X} \text{tr}(XD_2M^TD_1) - \frac{\partial}{\partial X} \text{tr}(X^TD_1MD_2)
= (D_1D_2)^T D_2 D_1 + D_1D_1XD_2D_2 - D_1MD_2 - (D_2M^TD_1)^T
= 2D_1(D_1XD_2 - M)D_2. \tag{8}
\]

Similarly, we have

\[
\frac{\partial}{\partial X} L_2(X) = 2A_3^T(A_3XA_4 - N)A_4^T. \tag{9}
\]
Let $X_1(k)$ and $X_2(k)$ be the estimates or iterative solutions of the system (6) at $k$-th iteration. The recursive formulas of $X_1(k)$ and $X_2(k)$ come from the gradient formulas (8) and (9), as follows:

$$X_1(k) = X(k-1) + \mu D_1(M - D_1 X(k-1) D_2) D_2$$
$$= X(k-1) + \mu D_1(E - A_1 X(k-1) A_2 + A_3 X(k-1) A_4) D_2,$$

$$X_2(k) = X(k-1) + \mu D_3(N - D_3 X(k-1) D_3) D_4$$
$$= X(k-1) + \mu D_3(E - A_1 X(k-1) A_2 + A_3 X(k-1) A_4) D_4.$$

Based on the hierarchical identification principle, the unknown variable $X$ is replaced by its estimates at the $(k-1)$-th iteration. To avoid duplicated computation, we introduce a matrix

$$S(k) = E - (A_1 X(k) A_2 + A_3 X(k) A_4),$$

so we have

$$X(k) = \frac{1}{2} (X_1(k) + X_2(k)) = X(k-1) + \mu (D_1 S(k-1) D_2 + D_3 S(k-1) D_4). \tag{10}$$

Since any diagonal matrix is sparse, the operation counts in the computation (10) can be substantially reduced. Let us denote $S(k) = \{s_{ij}(k)\}$, $X(k) = \{x_{ij}(k)\}$, and $D_l = [d_{ij}^{(l)}]$ for each $l = 1, 2, 3, 4$. Indeed, the multiplication of $D_1 S(k) D_2$ results in a matrix whose $(ij)$-th entry is the product of the $i$-th entry in the diagonal of $D_1$, the $(ij)$-th entry of $S(k)$, and the $j$-th entry of $D_2$—i.e., $D_1 S(k) D_2 = [d_{ij}^{(1)} s_{ij}(k) d_{jj}^{(2)}]$. Similarly, $D_3 S(k) D_4 = [d_{ii}^{(3)} s_{ij}(k) d_{jj}^{(4)}]$. Thus,

$$D_1 S(k) D_2 + D_3 S(k) D_4 = [(d_{ij}^{(1)} + d_{ij}^{(2)}) s_{ij}(k)].$$

The above discussion leads to the following Algorithm 1.

**Algorithm 1: Modified Jacobi-gradient based iterative (MJGI) algorithm**

1. Choose $\mu \in \mathbb{R}, \epsilon > 0$ and set $k = 1$;
2. for $k = 1, \ldots, n$
   - $S(k-1) = E - (A_1 X(k-1) A_2 + A_3 X(k-1) A_4)$;
   - $x_{ij}(k) = x_{ij}(k-1) + \mu(d_{ii}^{(1)} d_{jj}^{(2)} + d_{ii}^{(3)} d_{jj}^{(4)}) s_{ij}(k-1)$;
   - if $\|S(k-1)\|_F < \epsilon$ then
     - break;
   - else
     - $k = k + 1$;
   - end
3. end

The complexity analysis for each step of the algorithm is given by $2mn(m + n + 5)$. When $m = n$, the complexity analysis is $4n^3 + 10n^2 \in O(n^3)$, so that the algorithm runtime complexity is cubic time. The convergence property of the algorithm relies on the convergence factor $\mu$. The appropriate value of this parameter is determined in the next section.

3. Convergence Analysis of the Proposed Method

In this section, we make convergence analysis of Algorithm 1. First, we transform it into a linear iterative method of the first order: $x(k) = Tx(k-1)$ where $x(k)$ is a vector variable and $T$ is a matrix. The iteration matrix $T$ will reflect convergence criteria, convergence rate, and error estimates of the algorithm.
3.1. Convergence Criteria

**Theorem 1.** Assume that the generalized Sylvester matrix Equation (3) has a unique solution $X^*$. Denote $H = D(P)P$ and write $\sigma(H) = \{\lambda_1, \ldots, \lambda_{mn}\}$. Let $\{X(k)\}$ be a sequence generated from Algorithm 1.

1. Then, (AS) holds if and only if $\rho(I_{mn} - \mu H) < 1$.
2. If $\Re(\lambda_j) > 0$ for all $j = 1, \ldots, mn$, then (AS) holds if and only if
   
   \[ 0 < \mu < \max_{j=1,\ldots, mn} \frac{2\Re(\lambda_j)}{|\lambda_j|^2}. \]

3. If $\Re(\lambda_j) < 0$ for all $j = 1, \ldots, mn$, then (AS) holds if and only if
   
   \[ \min_{j=1,\ldots, mn} \frac{2\Re(\lambda_j)}{|\lambda_j|^2} < \mu < 0. \]

4. If $H$ is symmetric, then (AS) holds if and only if $\lambda_{max}(H)$ and $\lambda_{min}(H)$ have the same sign, and $\mu$ is chosen so that
   
   \[
   \begin{cases}
   0 < \mu < \frac{2}{\lambda_{max}(H)} & \text{if } \lambda_{min}(H) > 0, \\
   \frac{2}{\lambda_{min}(H)} < \mu < 0 & \text{if } \lambda_{max}(H) < 0.
   \end{cases}
   \]

**Proof.** From Algorithm 1, we start with considering the error matrices

\[
\hat{X}(k) = X(k) - X^*, \quad \hat{X}_1(k) = X_1(k) - X^* \quad \text{and} \quad \hat{X}_2(k) = X_2(k) - X^*.
\]

We will show that $\hat{X}(k) \to 0$, or equivalently, $\text{vec}(\hat{X}(k)) \to 0$ as $k \to \infty$. A direct computation reveals that

\[
\hat{X}(k) = \frac{1}{2}(X_1(k) + X_2(k)) = \hat{X}(k - 1) - \frac{1}{2}\mu D_1(A_1 \hat{X}(k - 1)A_2 + A_3 \hat{X}(k - 1)A_4)D_2 \\
- \frac{1}{2}\mu D_3(A_1 \hat{X}(k - 1)A_2 + A_3 \hat{X}(k - 1)A_4)D_4.
\]

By taking the vector operator and using properties of the Kronecker product, we have

\[
\text{vec}(\hat{X}(k)) = \text{vec}(\hat{X}(k - 1)) - \mu \text{vec}(D_1 A_1 \hat{X}(k - 1)A_2 D_2 + D_1 A_3 \hat{X}(k - 1)A_4 D_2) \\
- \mu \text{vec}(D_3 A_1 \hat{X}(k - 1)A_2 D_4 + D_3 A_3 \hat{X}(k - 1)A_4 D_4) \\
= \{I_{mn} - \mu[(D_2 \otimes D_1)(A_2^T \otimes A_1) + (D_2 \otimes D_1)(A_2^T \otimes A_3) \\
+ (D_4 \otimes D_3)(A_2^T \otimes A_1) + (D_4 \otimes D_3)(A_2^T \otimes A_3)]\} \text{vec}(\hat{X}(k - 1)) \\
= \left[I_{mn} - \mu(D_2 \otimes D_1 + D_4 \otimes D_3)(A_2^T \otimes A_1 + A_2^T \otimes A_3)\right] \text{vec}(\hat{X}(k - 1)).
\]

Let us denote the diagonal part of $P$ by $D(P)$. Indeed,

\[
D(P) = D_2 \otimes D_1 + D_4 \otimes D_3.
\]

Thus, we arrive at a linear iterative process

\[
\text{vec}(\hat{X}(k)) = [I_{mn} - \mu H] \text{vec}(\hat{X}(k - 1)), \quad (11)
\]

where $H = D(P)P$. Hence, the following statements are equivalent:
(i) $\text{vec } \tilde{X}(k) \to 0$ for any initial value $\text{vec } \tilde{X}(0)$.
(ii) System (11) has an asymptotically-stable zero solution.
(iii) The iteration matrix $I_{mn} - \mu H$ has spectral radius less than 1.

Indeed, since $I_{mn} - \mu H$ is a polynomial of $H$, we get

$$\rho(I_{mn} - \mu H) = \max_{\lambda \in \sigma(H)} |1 - \mu \lambda|. \quad (12)$$

Thus, $\rho(I_{mn} - \mu H) < 1$ if and only if $|1 - \mu \lambda| < 1$ for all $\lambda \in \sigma(H)$. Write $\lambda_j = a_j + ib_j$ where $a_j, b_j \in \mathbb{R}$.

It follows that the condition $|1 - \mu \lambda_j| < 1$ is equivalent to $(1 - \mu \lambda_j)(1 - \mu \bar{\lambda}_j) < 1$, or

$$\mu(-2a_j + \mu(a_j^2 + b_j^2)) < 0.$$

Thus, we arrive at two alternative conditions:

(i) $\mu > 0$ and $-2a_j + \mu(a_j^2 + b_j^2) < 0$ for all $j = 1, 2, 3, \ldots, mn$;
(ii) $\mu < 0$ and $-2a_j + \mu(a_j^2 + b_j^2) > 0$ for all $j = 1, 2, 3, \ldots, mn$.

Case 1 $a_j = \Re(\lambda_j) > 0$ for all $j$. In this case, $\rho(I_{mn} - \mu H) < 1$ if and only if

$$0 < \mu < \max_{j=1,\ldots,mn} \frac{2a_j}{a_j^2 + b_j^2}. \quad (13)$$

Case 2 $a_j = \Re(\lambda_j) < 0$ for all $j$. In this case, $\rho(I_{mn} - \mu H) < 1$ if and only if

$$\min_{j=1,\ldots,mn} \frac{2a_j}{a_j^2 + b_j^2} < \mu < 0. \quad (14)$$

Now, suppose that $H$ is a symmetric matrix. Then $I_{mn} - \mu H$ is also symmetric, and thus all its eigenvalue are real. Hence,

$$\rho(I_{mn} - \mu H) = \max \{|1 - \mu \lambda_{\text{min}}(H)|, |1 - \mu \lambda_{\text{max}}(H)|\}. \quad (15)$$

It follows that $\rho(I_{mn} - \mu H) < 1$ if and only if

$$0 < \mu \lambda_{\text{min}}(H) < 2 \quad \text{and} \quad 0 < \mu \lambda_{\text{max}}(H) < 2. \quad (16)$$

So, $\lambda_{\text{min}}(H)$ and $\lambda_{\text{max}}(H)$ cannot be zero.

Case 1 If $\lambda_{\text{max}}(H) \geq \lambda_{\text{min}}(H) > 0$, then the condition (16) is equivalent to

$$0 < \mu < \frac{2}{\lambda_{\text{max}}(H)}.$$

Case 2 If $\lambda_{\text{min}}(H) \leq \lambda_{\text{max}}(H) < 0$, then the condition (16) is equivalent to

$$\frac{2}{\lambda_{\text{min}}(H)} < \mu < 0.$$

Case 3 If $\lambda_{\text{min}}(H) < 0 < \lambda_{\text{max}}(H)$, then

$$\frac{2}{\lambda_{\text{min}}(H)} < \mu < 0 \quad \text{and} \quad 0 < \mu < \frac{2}{\lambda_{\text{max}}(H)},$$

which is a contradiction.
Therefore, the condition (16) holds if and only if \( \lambda_{\text{max}}(H) \) and \( \lambda_{\text{min}}(H) \) have the same sign and \( \mu \) is chosen according to the above condition. \( \square \)

### 3.2. Convergence Rate and Error Estimate

We now discuss the convergence rate and error estimates of Algorithm 1 from the iterative process (11).

Suppose that Algorithm 1 satisfies the (AS) property—i.e., \( \rho(I_{nn} - \mu H) < 1 \). From (11), we have

\[
\|X(k) - X^\ast\|_F = \|\text{vec } \tilde{X}(k)\|_F = \|(I_{nn} - \mu H) \text{vec } \tilde{X}(k - 1)\|_F \\
\leq \|I_{nn} - \mu H\|_2 \|\tilde{X}(k - 1)\|_F = \|I_{nn} - \mu H\|_2 \|X(k - 1) - X^\ast\|_F.
\]

(17)

It follows inductively that for each \( k \in \mathbb{N} \),

\[
\|X(k) - X^\ast\|_F \leq \|I_{nn} - \mu H\|_2 \|X(0) - X^\ast\|_F.
\]

(18)

Hence, the spectral norm of \( I_{nn} - \mu H \) describes how fast the approximated solutions \( X(k) \) converges to the exact solution \( X^\ast \). The smaller spectral radius, the faster \( X(k) \) goes to \( X^\ast \). In that case, since \( \|I_{nn} - \mu H\|_2 < 1 \), if \( \|X(k - 1) - X^\ast\|_F \neq 0 \) (i.e., \( X(k) \) is not the exact solution) then

\[
\|X(k) - X^\ast\|_F < \|X(k - 1) - X^\ast\|_F.
\]

(19)

Thus, the error at each iteration gets smaller than the previous one.

The above discussion is summarized in the following theorem.

**Theorem 2.** Suppose that the parameter \( \mu \) is chosen as in Theorem 1 so that Algorithm 1 satisfies (AS). Then, the convergence rate of the algorithm is governed by the spectral radius (16). Moreover, the error estimate \( \|X(k) - X^\ast\|_F \) compared to the previous step and the fast step are provided by (17) and (18), respectively. In particular, the error at each iteration gets smaller than the (nonzero) previous one, as in (19).

From (16), if the eigenvalues of \( \mu H \) are close to 1, then the spectral radius of the iteration matrix is close to 0, and hence, the error \( \text{vec } \tilde{X}(k) \) or \( \tilde{X}(k) \) converge faster to 0.

**Remark 1.** The convergence criteria and the convergence rate of Algorithm 1 depend on \( A, B, C \) and \( D \) but not on \( E \). However, the matrix \( E \) can be used for the stopping criteria.

The next proposition determines the iteration number for which the approximated solution \( X(k) \) is close to the exact solution \( X^\ast \) so that \( \|X(k) - X^\ast\|_F < \epsilon \).

**Proposition 1.** According to Algorithm 1, for each given error \( \epsilon > 0 \), we have \( \|X(k) - X^\ast\|_F < \epsilon \) after \( k^\ast \), iterations for any \( k^\ast \), such that

\[
k^\ast > \frac{\log \epsilon - \log \|X(k) - X^\ast\|_F}{\log \|I_{nn} - \mu H\|_2}.
\]

(20)

**Proof.** From the estimation (18), we have

\[
\|X(k) - X^\ast\|_F \leq \|I_{nn} - \mu H\|_2 \|X(0) - X^\ast\|_F \rightarrow 0 \text{ as } k \rightarrow \infty.
\]

This means precisely that for each given \( \epsilon > 0 \), there is a \( k^\ast \in \mathbb{N} \) such that for all \( k \geq k^\ast \),

\[
\|I_{nn} - \mu H\|_2 \|X(0) - X^\ast\|_F < \epsilon.
\]
Taking logarithms, we have that the above condition is equivalent to (20). Thus, if we run Algorithm 1 \(k^*\) times, then we get \(\|X(k) - X^*\| < \epsilon\) as desired.

3.3. Optimal Parameter

We discuss the fastest convergence factor for Algorithm 1.

**Theorem 3.** The optimal convergence factor \(\mu\) for which Algorithm 1 satisfies (AS) is one that minimizes \(\|I_{mn} - \mu H\|_2\). If, in addition, \(H\) is symmetric, then the optimal convergence factor for which the algorithm satisfies (AS) is determined by

\[
\mu_{\text{opt}} = \frac{2}{\lambda_{\min}(H) + \lambda_{\max}(H)}.
\]

In this case, the convergence rate is governed by

\[
\rho(I_{mn} - \mu H) = \frac{\lambda_{\max}(H) - \lambda_{\min}(H)}{\lambda_{\max}(H) + \lambda_{\min}(H)} = \frac{\kappa^2 - 1}{\kappa^2 + 1},
\]

where \(\kappa\) denotes the condition number of \(H\), and we have the following estimates:

\[
\|X(k) - X^*\|_F \leq \frac{\kappa^2 - 1}{\kappa^2 + 1} \|X(k - 1) - X^*\|_F,
\]

\[
\|X(k) - X^*\|_F \leq (\frac{\kappa^2 - 1}{\kappa^2 + 1})^k \|X(0) - X^*\|_F.
\]

**Proof.** From Theorem 2, it is clear that the fastest convergence factor is attained at a convergence factor that minimizes \(\|I_{mn} - \mu H\|_2\). Now, assume that \(H\) is symmetric. Then, \(I_{mn} - \mu H\) is also symmetric, thus all its eigenvalues are real and

\[
\|I_{mn} - \mu H\|_2 = \rho(I_{mn} - \mu H).
\]

For convenience, denote \(a = \lambda_{\min}(H)\), \(b = \lambda_{\max}(H)\), and

\[
f(\mu) := \rho(I_{mn} - \mu H) = \max \{|1 - \mu a|, |1 - \mu b|\}.
\]

First, we consider the case \(\lambda_{\min}(H) > 0\). To obtain the fastest convergence factor, according to (15), we must solve the following optimization problem

\[
\min_{0 < \mu < \frac{2}{\lambda_{\max}(H)}} \|I_{mn} - \mu H\|_2 = \min_{0 < \mu < \frac{b}{a}} f(\mu).
\]

We obtain that the minimizer is given by \(\mu_{\text{opt}} = 2/(a + b)\), so that \(f(\mu_{\text{opt}}) = (b - a)/(b + a)\). For the case \(\lambda_{\max}(H) < 0\), we solve the following optimization problem

\[
\min_{\frac{2}{\lambda_{\max}(H)} < \mu < 0} \|I_{mn} - \mu H\|_2 = \min_{\frac{b}{a} < \mu < 0} f(\mu).
\]

A similar argument yields the same minimizer (21) and the same convergence rate (22). From (17), (18) and (25), we obtain the bounds (23) and (24).

4. Numerical Simulations

In this section, we report numerical results to illustrate the effectiveness of Algorithm 1. We consider various sizes of matrix systems, namely, small \((2 \times 2)\), medium \((10 \times 10)\) and large \((100 \times 100)\). For the generalized Sylvester equation, we compare the performance of Algorithm 1 to the GI and LSI algorithms.
For the Sylvester equation, we compare our algorithm with GI, RGI, AGBI and JGI algorithms. All iterations have been carried out the same environment: MATLAB R2017b, Intel(R) Core(TM) i7-7660U CPU @ 2.5GHz, RAM 8.00 GB Bus speed 2133 MHz. We abbreviate IT and CPU for iteration time and CPU time (in seconds), respectively. As step k-th of the iteration, we consider the following error:

$$\delta(k) := \| E - A_1 X(k) A_2 - A_3 X(k) A_4 \|_F$$

where \( X(k) \) is the k-th approximated solution of the corresponding system.

### 4.1. Numerical Simulation for the Generalized Sylvester Matrix Equation

**Example 1.** Consider the matrix equation \( A_1 X A_2 + A_3 X A_4 = E \) where

\[
A_1 = \begin{bmatrix}
0.6959 & -0.6385 \\
0.6999 & 0.0336 \\
\end{bmatrix},
A_2 = \begin{bmatrix}
-0.0688 & -0.5309 \\
0.3196 & 0.6544 \\
\end{bmatrix},
A_3 = \begin{bmatrix}
0.4076 & 0.7184 \\
-0.8200 & 0.9686 \\
\end{bmatrix},
A_4 = \begin{bmatrix}
0.5313 & 0.1056 \\
0.3251 & 0.6110 \\
\end{bmatrix},
E = \begin{bmatrix}
0.7788 & 0.0908 \\
0.4235 & 0.2665 \\
\end{bmatrix}.
\]

Then, the exact solution of \( X \) is

\[
X^* = \begin{bmatrix}
1.3036 & -0.0532 \\
1.2725 & 1.2284 \\
\end{bmatrix}.
\]

Choose \( X(0) = \text{zeros}(2) \). In this case, all eigenvalues of \( H \) have positive real parts. The effect of changing the convergence factor \( \mu \) is illustrated in Figure 1. According to Theorem 1, the criteria for the convergence of \( X(k) \) is that \( \mu \in (0, 4.1870) \). Since \( \mu_1, \mu_2, \mu_3, \mu_4 \) satisfy this criteria, the error is becoming smaller and goes to zero as \( k \) increase, as in Figure 1. Among them, \( \mu_4 = 4.0870 \) gives the fastest convergence. For \( \mu_5 \) and \( \mu_6 \), which do not meet the criteria, the error \( \delta(k) \) does not converge to zero.

![Figure 1. Error of Example 1.](image)

### Example 2.

Suppose that \( A_1 X A_2 + A_3 X A_4 = E \), where \( A_1, A_2, A_3, A_4 \) and \( E \) are 10 × 10 matrices where
\[ A_1 = \text{tridiag}(1, 3, -1), A_2 = \text{tridiag}(1, 1, -2), A_3 = \text{tridiag}(-2, -2, 3), \]
\[ A_4 = \text{tridiag}(-3, 2, -1) \text{ and } E = \text{heptadiag}(1, -2, 1, -2, 1, -3). \]

Here, \( E \) is a heptadiagonal matrix—i.e., a band matrix with bandwidth 3. Choose an initial matrix \( X(0) = \text{zeros}(10) \), where \( \text{zeros}(n) \) is an \( n \times n \) matrix that contains 0 for every position. We compare Algorithm 1 with the direct method, LSI and GI algorithms. Table 1 shows the errors at the final step of iteration as well as the computation time after 75 iterations. Figure 2 illustrates that the approximated solutions via LSI diverge, while those via GI and MJGI converge. Table 1 and Figure 2 imply that our algorithm takes significantly less computational time and error than others.

**Table 1.** Computational time and error for Example 2.

| Method | IT | CT          | Error: \( \delta(75) \) |
|--------|----|-------------|--------------------------|
| Direct | -  | 0.0364 -    | -                        |
| LSI    | 75 | 0.0125 1.1296 \( \times 10^5 \) |
| GI     | 75 | 0.0049 1.4185 |
| MJGI   | 75 | 0.0022 0.5251 |

**Figure 2.** Error of Example 2.

**Example 3.** We consider the equation \( A_1 X A_2 + A_3 X A_4 = E \) in which \( A_1, A_2, A_3, A_4 \) and \( E \) are \( 100 \times 100 \) matrices determined by

\[ A_1 = \text{tridiag}(1, 1, -1), A_2 = \text{tridiag}(1, 2, -2), A_3 = \text{tridiag}(-1, -2, 3), \]
\[ A_4 = \text{tridiag}(-2, 1, -1) \text{ and } E = \text{heptadiag}(1, 2, -4, 1, -2, 2, -3). \]

The initial matrix is given by \( X(0) = \text{zeros}(100) \). We run LSI, GI and MJGI algorithms by using

\[ \mu = 0.1, \quad \mu = (\|A_1\|^2 \|A_2\|^2 + \|A_3\|^2 \|A_4\|^2)^{-1}, \]
\[ \mu = 2(\|A_1\|^2 \|A_2\|^2 + \|A_3\|^2 \|A_4\|^2)^{-1}, \]

respectively. The reported result in Table 2 and Figure 3 illustrate that the approximated solution generated from LSI diverges, while those from GI or MJGI converge. Both computational time and the error \( \delta(100) \) from MJGI are less than those from GI.
### Table 2. Computational time and error for Example 3.

| Method | IT  | CT            | Error: $\delta(100)$ |
|--------|-----|---------------|-----------------------|
| Direct | -   | 34.6026       | -                     |
| LSI    | 100 | 0.1920        | $2.7572 \times 10^4$ |
| GI     | 100 | 0.0849        | 4.7395                |
| MJGI   | 100 | 0.0298        | 1.8844                |

#### Figure 3. Comparison of Example 3.

4.2. Numerical Simulation for Sylvester Matrix Equation

Assume that the Sylvester equation

$$A_1X + XA_4 = E$$  \hspace{1cm} (26)

has a unique solution. This condition is equivalent to that the Kronecker sum $A_4^T \oplus A_1$ is invertible, or all possible sums between eigenvalues of $A_1$ and $A_4$ are nonzero. To solve (26), the Algorithm 2 is proposed:

$$T(k) := E - A_1X(k) - X(k)A_4.$$

**Algorithm 2:** Modified Jacobi-gradient based iterative (MJGI) algorithm for Sylvester equation

1. Choose $\mu \in \mathbb{R}, \epsilon > 0$ and set $k = 1$;
2. for $k = 1, \ldots, n$ do
   1. $T(k) = E - A_1X(k) - X(k)A_4$;
   2. $x_{ij}(k) = x_{ij}(k-1) + \mu(d_{ii}^{(1)} + d_{jj}^{(4)})t_{ij}(k-1)$ where $T(k) = [t_{ij}(k)]$;
   3. if $\|T(k-1)\|_F < \epsilon$ then
      1. break;
   else
      1. $k = k + 1$;
end
Example 4. Consider the equation $A_1X + XA_4 = E$, in which $E$ is the same matrix as in the previous example, $A_1 = \text{tridiag}(2, -1, 1) \in \mathbb{R}^{10\times10}$ and $A_4 = \text{tridiag}(-1, 1, -2) \in \mathbb{R}^{10\times10}$.

In this case, all eigenvalues of the iteration matrix have positive real parts, so that we can apply our algorithm. We compare our algorithm with GI, RGI, AGBI and JGI algorithms. The results after running 100 iterations are shown in Figure 4 and Table 3. According to the error and CT in Table 3 and Figure 4, our algorithm uses less computational time and has smaller errors than others.

| Method | IT | CT  | Error: $\delta(100)$ |
|--------|----|-----|---------------------|
| Direct | -  | 0.0118 | -                  |
| GI     | 100 | 0.0051 | 2.5981             |
| RGI    | 100 | 0.0061 | 3.4741             |
| AGBI   | 100 | 0.0051 | 7.3306             |
| JGI    | 100 | 0.0038 | 17.2652            |
| MJGI   | 100 | 0.0028 | 0.4281             |

Figure 4. Errors of Example 4.

5. Conclusions and Suggestion

A modified Jacobi-gradient (MJGI) algorithm (Algorithm 1) is proposed for solving the generalized Sylvester matrix Equation (3). In order to have MJGI algorithm applicable for any sizes of matrix system and any initial matrices, the convergence factor $\mu$ must be chosen properly according to Theorem 1. In this case, the iteration matrix $I_{mn} - \mu H$ has a spectral radius less than 1. When the iteration matrix is symmetric, we determine the optimal convergent factor $\mu_{opt}$ which enhances the algorithm reaching the fastest rate of convergence. The asymptotic convergence rate of the algorithm is governed by the spectral radius of $I_{mn} - \mu H$. So, if the eigenvalue $H$ is close to 1, then the algorithm converges faster in the long run. The numerical examples reveal that our algorithm is suitable for small ($2 \times 2$), medium ($10 \times 10$) and large ($100 \times 100$) sizes of matrix systems. In addition, the MJGI algorithm performs well compared to recent gradient iterative algorithms. For future works, we may add another
parameter for an updating step to make the algorithm converge faster—see [25]. Another possible way is to apply the idea in this paper to derive an iterative algorithm for nonlinear matrix equations.

Author Contributions: Supervision, P.C.; software, N.S.; writing—original draft preparation, N.S.; writing—review and editing, P.C. All authors contributed equally and significantly in writing this article. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Acknowledgments: The first author received financial support from the RA-TA graduate scholarship from the faculty of Science, King Mongkut’s Institute of Technology Ladkrabang, Grant. No. RA/TA-2562-M-001 during his Master’s study.

Conflicts of Interest: The authors declare no conflict of interest.

References
1. Shang, Y. Consensus seeking over Markovian switching networks with time-varying delays and uncertain topologies. *Appl. Math. Comput.* 2016, 273, 1234–1245. [CrossRef]
2. Shang, Y. Average consensus in multi-agent systems with uncertain topologies and multiple time-varying delays. *Linear Algebra Appl.* 2014, 459, 411–429. [CrossRef]
3. Golub, G.H.; Nash, S.; Van Loan, C.F. A Hessenberg-Schur method for the matrix AX + XB = C. *IEEE Trans. Automat. Control.* 1979, 24, 909–913. [CrossRef]
4. Ding, F.; Chen, T. Hierarchical least squares identification methods for multivariable systems. *IEEE Trans. Automat. Control* 1997, 42, 408–411. [CrossRef]
5. Benner, P.; Quintana-Orti, E.S. Solving stable generalized Lyapunov equations with the matrix sign function. *Numer. Algorithms* 1999, 20, 75–100. [CrossRef]
6. Starke, G.; Niethammer, W. SOR for AX − XB = C. *Linear Algebra Appl.* 1991, 154–156, 355–375. [CrossRef]
7. Jonsson, I.; Kagstrom, B. Recursive blocked algorithms for solving triangular systems—Part I: One-sided and coupled Sylvester-type matrix equations. *ACM Trans. Math. Softw.* 2002, 28, 392–415. [CrossRef]
8. Jonsson, I.; Kagstrom, B. Recursive blocked algorithms for solving triangular systems—Part II: Two-sided and generalized Sylvester and Lyapunov matrix equations. *ACM Trans. Math. Softw.* 2002, 28, 416–435. [CrossRef]
9. Kaabi, A.; Kerayechian, A.; Toutounian, F. A new version of successive approximations method for solving Sylvester matrix equations. *Appl. Math. Comput.* 2007, 186, 638–648. [CrossRef]
10. Lin, Y.Q. Implicitly restarted global FOM and GMRES for nonsymmetric matrix equations and Sylvester equations. *Appl. Math. Comput.* 2005, 167, 1004–1025. [CrossRef]
11. Kressner, D.; Sirkovic, P. Truncated low-rank methods for solving general linear matrix equations. *Numer. Linear Algebra Appl.* 2015, 22, 564–583. [CrossRef]
12. Dehghan, M.; Shirilord, A. A generalized modified Hermitian and skew-Hermitian splitting (GMHSS) method for solving complex Sylvester matrix equation. *Appl. Math. Comput.* 2019, 348, 632–651. [CrossRef]
13. Dehghan, M.; Shirilord, A. Solving complex Sylvester matrix equation by accelerated double-step scale splitting (ADSS) method. *Eng. Comput.* 2019. [CrossRef]
14. Li, S.Y.; Shen, H.L.; Shao, X.H. PHSS iterative method for solving generalized Lyapunov equations. *Mathematics* 2019, 7, 38. [CrossRef]
15. Shen, H.L.; Li, Y.R.; Shao, X.H. The four-parameter PSS method for solving the Sylvester equation. *Mathematics* 2019, 7, 105. [CrossRef]
16. Hajarian, M. Generalized conjugate direction algorithm for solving the general coupled matrix equations over symmetric matrices. *Numer. Algorithms* 2016, 73, 591–609. [CrossRef]
17. Hajarian, M. Extending the CGLS algorithm for least squares solutions of the generalized Sylvester-transpose matrix equations. *J. Frankl. Inst.* 2016, 353, 1168–1185. [CrossRef]
18. Dehghan, M.; Mohammad-Arani, R. Generalized product-type methods based on Bi-conjugate gradient (GPBiCG) for solving shifted linear systems. *Comput. Appl. Math.* 2017, 36, 1591–1606. [CrossRef]
19. Ding, F.; Chen, T. Gradient based iterative algorithms for solving a class of matrix equations. *IEEE Trans. Automat. Control* 2005, 50, 1216–1221. [CrossRef]
20. Niu, Q.; Wang, X.; Lu, L.-Z. A relaxed gradient based algorithm for solving Sylvester equation. *Asian J. Control* **2011**, *13*, 461–464. [CrossRef]
21. Zhang, X.D.; Sheng, X.P. The relaxed gradient based iterative algorithm for the symmetric (skew symmetric) solution of the Sylvester equation $AX + XB = C$. *Math. Probl. Eng.* **2017**, *2017*, 1624969. [CrossRef]
22. Xie, Y.J.; Ma, C.F. The accelerated gradient based iterative algorithm for solving a class of generalized Sylvester-transpose matrix equation. *Appl. Math. Comput.* **2012**, 218, 5620–5628. [CrossRef]
23. Ding, F.; Chen, T. Iterative least-squares solutions of coupled Sylvester matrix equations. *Syst. Control Lett.* **2005**, *54*, 95–107. [CrossRef]
24. Fan, W.; Gu, C.; Tian, Z. Jacobi-gradient iterative algorithms for Sylvester matrix equations. In Proceedings of the 14th Conference of the International Linear Algebra Society, Shanghai University, Shanghai, China, 16–20 July 2007.
25. Tian, Z.; Tian, M.; Gu, C.; Hao, X. An accelerated Jacobi-gradient based iterative algorithm for solving Sylvester matrix equations. *Filomat* **2017**, *31*, 2381–2390. [CrossRef]
26. Ding, F.; Liu, P.X.; Chen, T. Iterative solutions of the generalized Sylvester matrix equations by using the hierarchical identification principle. *Appl. Math. Comput.* **2008**, *197*, 41–50. [CrossRef]
27. Horn, R.A.; Johnson, C.R. *Topics in Matrix Analysis*; Cambridge University Press: New York, NY, USA, 1991.

**Publisher’s Note:** MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.