Numer. Math. Theor. Meth. Appl.
doi: 10.4208/nmtma.OA-2017-0010

Vol. **11**, No. 1, pp. 140-159
February 2018

Extending GCR Algorithm for the Least Squares Solutions on a Class of Sylvester Matrix Equations

Baohua Huang and Changfeng Ma[*]

College of Mathematics and Informatics, Fujian Key Laborotary of Mathematical Analysis and Applications, Fujian Normal University, Fuzhou 350117, China

Received 1 January 2017; Accepted (in revised version) 2 August 2017

Abstract. The purpose of this paper is to derive the generalized conjugate residual (GCR) algorithm for finding the least squares solution on a class of Sylvester matrix equations. We prove that if the system is inconsistent, the least squares solution can be obtained within finite iterative steps in the absence of round-off errors. Furthermore, we provide a method for choosing the initial matrix to obtain the minimum norm least squares solution of the problem. Finally, we give some numerical examples to illustrate the performance of GCR algorithm.

AMS subject classifications: 65F10

Key words: Sylvester matrix equation, Least squares solution, Generalized conjugate residual algorithm, Numerical experiments

1. Introduction

Matrix equations appear frequently in many areas of applied mathematics and play important roles in many applications, such as control theory and system theory [25–27]. For example, the descriptor linear system

$$A_1\dot{x}+A_1\dot{x}+B_0u=0 \tag{1.1}$$

captures the dynamic behavior of many physical systems in practice [29–31] and the second order linear system

$$A_2\ddot{x}+A_1\dot{x}+A_1\dot{x}+B_0u=0 \tag{1.2}$$

has wide applications in vibration and structural analysis, robotics control and spacecraft control [32, 33]. It is known that certain control problems, such as pole/eigenstructure assignment and observer design are closely related to the generalized Sylvester matrix equations (1.1) and (1.2). To solve the additive decomposition problem of a transfer

[*]Corresponding author. *Email address:* macf@fjnu.edu.cn (C. F. Ma)

matrix [34], we need to find the solution pair (X,Y) of the generalized coupled Sylvester matrix equations

$$\begin{cases} AX - YB = C, \\ DX - YE = F. \end{cases} \tag{1.3}$$

In this paper, we consider the solution of the following matrix equations

$$\begin{cases} A_1XB_1 = C_1, \\ A_2XB_2 = C_2, \end{cases} \tag{1.4}$$

where $A_1 \in R^{p\times m}$, $B_1 \in R^{n\times q}$, $C_1 \in R^{p\times q}$ and $A_2 \in R^{r\times m}$, $B_2 \in R^{n\times s}$, $C_2 \in R^{r\times s}$ are given matrices, and $X \in R^{m\times n}$ is an unknown matrix to be determined.

There have been many papers considering various solutions of the matrix equations(1.4). For instance, Mitra [1, 2] gave conditions for the existence of a solution and a representation of the general common solution to the system (1.4). Navarra et al. [3] derived the sufficient and necessary conditions for the existence of a common solution to the system (1.4). Yuan [4] obtained an analytical expression of the least squares solution of the system (1.4) by using the generalized singular value decomposition (GSVD) of matrices. Sheng and Chen [5] presented a finite iterative method when the system (1.4) is consistent. Cai and Chen [6] constructed an iterative algorithm for the least squares bisymmetric solution of the matrix equations (1.4) by applying the theory of convex analysis. In [24], Dehghan and Hajarian presented an algorithm for solving matrix equations (1.4) in order to obtain (R,S)-symmetric and (R,S)-skew symmetric solution. An efficient iterative method was proposed for finding the generalized centro-symmetric solution of the matrix equations (1.4) by Dehghan and Hajarian [43]. Chen et al. [38] obtained common symmetric least squares solutions of the matrix equations (1.4) by using the LSQR iterative method. Wang et al. [42] presented a direct method to solve the least squares Hermitian problem of the complex matrix equations $(AXB, CXD) = (E,F)$ with the help of matrix-vector product and the Moore-Penrose generalized inverse.

In the past decades, most of the proposed iterative algorithms for solving linear matrix equations were obtained from the extension of algorithms which were previously introduced for solving the linear system of equations $Ax = b$. See for [7–13, 39, 40]. For example, Bai proposed a Hermitian and skew-Hermitian splitting (HSS) iteration algorithm to solve the Sylvester matrix equation

$$AX + XB = F, \tag{1.5}$$

with non-Hermitian and positive definite/semi-definite matrices [44]. A nested splitting conjugate gradient (NSCG) iteration method [45] was proposed for solving the matrix equation

$$AXB = C. \tag{1.6}$$

Based on the conjugate gradient algorithm, several iterative algorithms were proposed for solving the (coupled) linear matrix equations [14–17]. The matrix forms of CGS, GPBiCG, QMRCGSTAB, BiCOR, Bi-CGSTAB, CORS, BiCG, Bi-CR and CGLS algorithms were given to solve linear matrix equations [18–22, 41].

Inspired by the previous works, in this paper, we are devoted to obtain the generalized conjugate residual (GCR) algorithm for finding the least squares solution of the matrix equations (1.4). When the system (1.4) is inconsistent, we prove that the least squares solution of the system (1.4) can be obtained within finite iterative steps in the absence of round-off errors. Moreover, the least Frobenius norm least squares solution can be derived by finding the special type of the initial matrix.

For convenience, we use the following notations throughout this paper. Let $R^{m\times n}$ be the sets of all real $m\times n$ matrices. We abbreviate $R^{n\times 1}$ as R^n. For $A\in R^{m\times n}$, we write $A^T, ||A||_F, tr(A)$ and A^{-1} to denote transpose, Frobenius norm, the trace and the inverse of matrix A, respectively. For any matrix $A=(a_{ij})$, $B=(b_{ij})$, matrix $A\otimes B$ denotes the Kronecker product defined as $A\otimes B=(a_{ij}B)$. For the matrix $X=(x_1,x_2,\cdots,x_n)\in R^{m\times n}$, $vec(X)$ denotes the vec operator defined as $vec(X)=(x_1^T,x_2^T,\cdots,x_n^T)^T\in R^{mn}$.

The rest of this paper is organized as follows. In Section 2, we introduce a matrix form of generalized conjugate residual (GCR) algorithm to solve Sylvester matrix equations (1.4) over the least squares solution. Then we prove that if the system is inconsistent, the least squares solution can be obtained within finite iterative steps in the absence of round-off errors. In Section 3, we provide a method for choosing the initial matrix to obtain the least Frobenius norm least squares solution of the system (1.4). In Section 4, we present some numerical experiments. The paper ends up with conclusions in Section 5.

2. The generalized conjugate residual algorithm for solving the matrix equations (1.4)

First, we give the definition of the inner product. In the space $R^{m\times n}$ over the field R, the inner product can be defined as

$$\langle A,\ B\rangle = tr(B^TA). \tag{2.1}$$

The norm of a matrix generated by this inner product space is denoted by $||\cdot||$. Then, for $A\in R^{m\times n}$, we have

$$\begin{aligned}&||A||^2=\langle A,\ A\rangle = tr(A^TA)=||A||_F^2,\\ &\langle A,\ B\rangle=\langle vec(A),\ vec(B)\rangle.\end{aligned} \tag{2.2}$$

In addition, from the definition of the inner product and the properties of matrix trace, we have the following results:

$$\langle A,\ B\rangle = tr\big(B^TA\big)=tr\big(AB^T\big)=tr\big(BA^T\big)=tr\big(A^TB\big)=\langle B,\ A\rangle.$$

So far, many iterative methods have been proposed for solving the linear equations

$$Ax=b,\quad A\in R^{n\times n},\quad b\in R^n, \tag{2.3}$$

Algorithm 2.1 GCR Algorithm.

```
1 Initial vector: $x_0 \in R^n$ arbitrary, $r_0 = b - Ax_0$, $p_0 = r_0$, $k := 0$.
2 Recursions:
    $\alpha_k = \frac{(r_k, Ap_k)}{(Ap_k, Ap_k)}$,   $x_{k+1} = x_k + \alpha_k p_k$,   $r_{k+1} = r_k - \alpha_k Ap_k$,
    $\beta_i^k = -\frac{(Ar_{k+1}, Ap_i)}{(Ap_i, Ap_i)}$,   for $i = 0, 1, \cdots, k$,   $p_{k+1} = r_{k+1} + \sum_{i=0}^{k} \beta_i^k p_i$.
```

where A is positive definite matrix, i.e., the symmetric part of A, $H = (A + A^T)/2$ is symmetric positive definite matrix. Generalized conjugate residual method is one of the iterative methods. The ordinary generalized conjugate residual (GCR) to solve Eq. (2.3) is as follows [37].

For the convenience of discussion in what follows, we adopt the following notation:

$$G(Y) = A_1^T A_1 Y B_1 B_1^T + A_2^T A_2 Y B_2 B_2^T. \tag{2.4}$$

It is easy to prove that G is a linear operator.

Now, we construct the following generalized conjugate residual algorithm (GCR) with the matrix form for finding the least squares solution of the system (1.4).

Remark 2.1. In the implementation of the Algorithm 2.2, in order to save memory requirements, we can calculate Q_{k+1} from Eq. (2.5). Obviously, we have

$$Q_{k+1} = S_{k+1} + \sum_{s=0}^{k} \beta_s^{(k)} Q_s. \tag{2.7}$$

To prove the convergence property of Algorithm 2.2, we first establish the following lemmas.

Lemma 2.1. *Let the sequence $\{Q_k\}$ be generated by Algorithm 2.2. Then we have*

$$\langle Q_i, Q_j \rangle = 0, \quad i, j = 0, 1, \cdots, k, \ i \neq j. \tag{2.8}$$

Proof. First, we prove

$$\langle Q_i, Q_j \rangle = 0, \quad 0 \le i < j \le k. \tag{2.9}$$

By mathematical induction, for $k = 1$, by using update rules of Q_k and P_k, the definition of $\beta_s^{(k)}$, we obtain

$$\begin{aligned}\langle Q_0, Q_1 \rangle &= \langle Q_0, G(P_1) \rangle = \langle Q_0, G(\tilde{R}_1 + \beta_0^{(0)} P_0) \rangle \\ &= \langle Q_0, G(\tilde{R}_1) \rangle + \beta_0^{(0)} \langle Q_0, G(P_0) \rangle = \langle Q_0, S_1 \rangle + \beta_0^{(0)} \langle Q_0, Q_0 \rangle \\ &= \langle Q_0, S_1 \rangle - \frac{\langle S_1, Q_0 \rangle}{\langle Q_0, Q_0 \rangle} \langle Q_0, Q_0 \rangle = 0.\end{aligned}$$

Algorithm 2.2 Generalized Conjugate Residual Algorithm with matrix form.

```
Input appropriate dimensionality matrices $A_1 \in R^{p\times m}$, $B_1 \in R^{n\times q}$, $C_1 \in R^{p\times q}$ and $A_2 \in R^{r\times m}$, $B_2 \in R^{n\times s}$, $C_2 \in R^{r\times s}$ in (1.4). Choose the initial matrix $X_0 \in R^{m\times n}$.
  1 Compute
      $R_0^{(1)} = C_1 - A_1X_0B_1, \quad R_0^{(2)} = C_2 - A_2X_0B_2, \quad \tilde{R}_0 = A_1^TR_0^{(1)}B_1^T + A_2^TR_0^{(2)}B_2^T.$
    Set $P_0 = \tilde{R}_0$ and $k := 0$.
  2 If $\|\tilde{R}_k\| = 0$, stop; otherwise, go to Step 3.
  3 Compute
      $Q_k = A_1^TA_1P_kB_1B_1^T + A_2^TA_2P_kB_2B_2^T \triangleq G(P_k).$
    Update the sequence $X_{k+1} = X_k + \alpha_kP_k$. Compute
      $R_{k+1}^{(1)} = C_1 - A_1X_{k+1}B_1, \quad R_{k+1}^{(2)} = C_2 - A_2X_{k+1}B_2.$
    Update the sequences
      $\tilde{R}_{k+1} = A_1^TR_{k+1}^{(1)}B_1^T + A_2^TR_{k+1}^{(2)}B_2^T = \tilde{R}_k - \alpha_kQ_k,$
      $S_{k+1} = A_1^TA_1\tilde{R}_{k+1}B_1B_1^T + A_2^TA_2\tilde{R}_{k+1}B_2B_2^T \triangleq G(\tilde{R}_{k+1})$
      $P_{k+1} = \tilde{R}_{k+1} + \sum_{s=0}^{k}\beta_s^{(k)}P_s,$    (2.5)
    where
      $\alpha_k = \frac{\langle \tilde{R}_k, Q_k\rangle}{\langle Q_k, Q_k\rangle}, \quad \beta_s^{(k)} = \frac{\langle S_{k+1}, Q_s\rangle}{\langle Q_s, Q_s\rangle}, \quad s = 0, 1, \cdots, k.$    (2.6)
  4 Set $k := k + 1$, return to Step 2.
```

Suppose that (2.9) holds for $k = l$. For $k = l + 1$, according to the update rules of Q_k, P_k, the relation (2.6) and induction principle, we have

$$\begin{aligned}
\langle Q_i, Q_{l+1}\rangle =&\langle Q_i, G(P_{l+1})\rangle = \langle Q_i, G(\tilde{R}_{l+1} + \sum_{s=0}^{l}\beta_s^{(l)}P_s)\rangle \\
=&\langle Q_i, G(\tilde{R}_{l+1})\rangle + \sum_{s=0}^{l}\beta_s^{(l)}\langle Q_i, G(P_s)\rangle = \langle Q_i, S_{l+1}\rangle + \sum_{s=0}^{l}\beta_s^{(l)}\langle Q_i, Q_s\rangle \\
=&\langle Q_i, S_{l+1}\rangle + \beta_i^{(l)}\langle Q_i, Q_i\rangle = \langle Q_i, S_{l+1}\rangle - \frac{\langle S_{l+1}, Q_i\rangle}{\langle Q_i, Q_i\rangle}\langle Q_i, Q_i\rangle = 0.
\end{aligned}$$

So the relation (2.9) holds for $k=l+1$. By the induction principle, the relation (2.9) holds for all $0\le i<j\le k$. For $i>j$, from the symmetry of inner product, we have

$$\langle Q_i,Q_j\rangle=\langle Q_j,Q_i\rangle=0,$$

which completes the proof. □

Lemma 2.2. *Let the sequences $\{Q_k\}$, $\{\tilde{R}_k\}$ and $\{S_k\}$ be generated by Algorithm 2.2. Then the following results hold:*

$$\begin{aligned}
&(1)\quad \langle \tilde{R}_i,Q_j\rangle=0,\quad i,j=0,1,\cdots,k,\ i>j;\\
&(2)\quad \langle \tilde{R}_i,Q_j\rangle=\langle \tilde{R}_0,Q_j\rangle,\quad i,j=0,1,\cdots,k,\ i\le j;\\
&(3)\quad \langle \tilde{R}_i,Q_i\rangle=\langle \tilde{R}_i,S_i\rangle;\\
&(4)\quad \langle \tilde{R}_i,S_j\rangle=0,\quad i,j=0,1,\cdots,k,\ i>j.
\end{aligned}$$

Proof. (1) We apply mathematical induction. For $i=1$, together the update rule of $\tilde{R}_k$ with the relation (2.6) yields

$$\begin{aligned}
\langle \tilde{R}_1,Q_0\rangle=&\langle \tilde{R}_0-\alpha_0Q_0,Q_0\rangle=\langle \tilde{R}_0,Q_0\rangle-\alpha_0\langle Q_0,Q_0\rangle\\
=&\langle \tilde{R}_0,Q_0\rangle-\frac{\langle \tilde{R}_0,Q_0\rangle}{\langle Q_0,Q_0\rangle}\langle Q_0,Q_0\rangle=0.
\end{aligned}$$

Assume that $\langle \tilde{R}_l,Q_j\rangle=0$ for all $l>j$, we shall show that $\langle \tilde{R}_{l+1},Q_j\rangle=0$ for all $l+1>j$. According to the update rule of $\tilde{R}_k$, we get

$$\langle \tilde{R}_{l+1},Q_j\rangle=\langle \tilde{R}_l-\alpha_lQ_l,Q_j\rangle=\langle \tilde{R}_l,Q_j\rangle-\alpha_l\langle Q_l,Q_j\rangle.$$

For $j<l$, combining the induction principle with Lemma 2.1 yields $\langle \tilde{R}_{l+1},Q_j\rangle=0$. For $j=l$, using the relation (2.6), we obtain

$$\langle \tilde{R}_{l+1},Q_l\rangle=\langle \tilde{R}_l,Q_l\rangle-\alpha_l\langle Q_l,Q_l\rangle=\langle \tilde{R}_l,Q_l\rangle-\frac{\langle \tilde{R}_l,Q_l\rangle}{\langle Q_l,Q_l\rangle}\langle Q_l,Q_l\rangle=0.$$

Thus we draw the conclusion by induction.

(2) By mathematical induction, for $i=0$, the result is trivial. Assume that $\langle \tilde{R}_i,Q_j\rangle=\langle \tilde{R}_0,Q_j\rangle$ for $i=l<j$, then using Algorithm 2.1 and Lemma 2.1, we have

$$\langle \tilde{R}_{l+1},Q_j\rangle=\langle \tilde{R}_l-\alpha_lQ_l,Q_j\rangle=\langle \tilde{R}_l,Q_j\rangle-\alpha_l\langle Q_l,Q_j\rangle=\langle \tilde{R}_l,Q_j\rangle=\cdots=\langle \tilde{R}_0,Q_j\rangle.$$

By the induction principle, we obtain this result.

(3) By the first claim of Lemma 2.2 and the relation (2.7), we get

$$\begin{aligned}
\langle \tilde{R}_i,Q_i\rangle=&\left\langle \tilde{R}_i,S_i+\sum_{s=0}^{i-1}\beta_s^{(i)}Q_s\right\rangle=\langle \tilde{R}_i,S_i\rangle+\left\langle \tilde{R}_i,\sum_{s=0}^{i-1}\beta_s^{(i)}Q_s\right\rangle\\
=&\langle \tilde{R}_i,S_i\rangle+\sum_{s=0}^{i-1}\beta_s^{(i)}\langle \tilde{R}_i,Q_s\rangle=\langle \tilde{R}_i,S_i\rangle.
\end{aligned}$$

(4) From the update rule of $\tilde{R}_k$ and the definition of S_j, we have

$$\begin{aligned}\langle \tilde{R}_i, S_j\rangle =& \langle \tilde{R}_i, G(\tilde{R}_j)\rangle = \left\langle \tilde{R}_i, G(P_j - \sum_{s=0}^{j-1}\beta_s^{j-1}P_s)\right\rangle \\ =& \langle \tilde{R}_i, G(P_j)\rangle - \left\langle \tilde{R}_i, \sum_{s=0}^{j-1}\beta_s^{j-1}G(P_s)\right\rangle \\ =& \langle \tilde{R}_i, Q_j\rangle - \sum_{s=0}^{j-1}\beta_s^{j-1}\langle \tilde{R}_i, Q_s\rangle.\end{aligned}$$

Combining this with the first claim of Lemma 2.2 yields

$$\langle \tilde{R}_i, S_j\rangle = \langle \tilde{R}_i, Q_j\rangle - \sum_{s=0}^{j-1}\beta_s^{j-1}\langle \tilde{R}_i, Q_s\rangle = 0, \quad \forall i > j.$$

This completes the proof. □

Lemma 2.3. *Let the sequences* $\{Q_k\}$, $\{\tilde{R}_k\}$ *and* $\{P_k\}$ *be generated by Algorithm* 2.2. *If* $B_1B_1^T \otimes A_1^TA_1 + B_2B_2^T \otimes A_2^TA_2$ *is an positive definite matrix and* $\tilde{R}_k \neq 0$*, then* $Q_k \neq 0$ *and* $P_k \neq 0$.

Proof. Using the third claim of Lemma 2.2, the update rule of S_k and the relation (2.4), we obtain

$$\begin{aligned}\langle \tilde{R}_k, Q_k\rangle =& \langle \tilde{R}_k, S_k\rangle = \langle \tilde{R}_k, G(\tilde{R}_k)\rangle \\ =& \langle \tilde{R}_k, A_1^TA_1\tilde{R}_kB_1B_1^T + A_2^TA_2\tilde{R}_kB_2B_2^T\rangle \\ =& \langle \tilde{R}_k, A_1^TA_1\tilde{R}_kB_1B_1^T\rangle + \langle \tilde{R}_k, A_2^TA_2\tilde{R}_kB_2B_2^T\rangle \\ =& \left\langle vec(\tilde{R}_k), (B_1B_1^T \otimes A_1^TA_1)vec(\tilde{R}_k)\right\rangle + \left\langle vec(\tilde{R}_k), (B_2B_2^T \otimes A_2^TA_2)vec(\tilde{R}_k)\right\rangle \\ =& \left\langle vec(\tilde{R}_k), \big(B_1B_1^T \otimes A_1^TA_1 + B_2B_2^T \otimes A_2^TA_2\big)vec(\tilde{R}_k)\right\rangle. \end{aligned} \tag{2.10}$$

As $\tilde{R}_k \neq 0$, we have $vec(\tilde{R}_k) \neq 0$. Since $B_1B_1^T \otimes A_1^TA_1 + B_2B_2^T \otimes A_2^TA_2$ is an positive definite matrix, we immediately have

$$\langle \tilde{R}_k, Q_k\rangle = \left\langle vec(\tilde{R}_k), \big(B_1B_1^T \otimes A_1^TA_1 + B_2B_2^T \otimes A_2^TA_2\big)vec(\tilde{R}_k)\right\rangle > 0.$$

Hence $Q_k \neq 0$.

On the other hand, notice that

$$Q_k = G(P_k) = A_1^TA_1P_kB_1B_1^T + A_2^TA_2P_kB_2B_2^T,$$

from the definition of the Kronecker product and vec operator, we have

$$vec(Q_k) = \Big(B_1B_1^T \otimes A_1^TA_1 + B_2B_2^T \otimes A_2^TA_2\Big)\, vec(P_k).$$

Therefore

$$vec(P_k) \neq 0,$$

due to $Q_k \neq 0$. Hence $P_k \neq 0$. The proof is completed. $\square$

Remark 2.2. $B_1B_1^T \otimes A_1^T A_1 + B_2 B_2^T \otimes A_2^T A_2$ is an positive definite matrix if one of the following conditions is satisfied:
(1) A_1 is full column rank matrix, B_1 is full row rank matrix.
(2) A_2 is full column rank matrix, B_2 is full row rank matrix.

Hence, the assumption is reasonable.

For convenience, we introduce the following notations.

$$\Gamma = \begin{pmatrix} B_1^T \otimes A_1 \\ B_2^T \otimes A_2 \end{pmatrix}, \quad b = \begin{pmatrix} vec(C_1) \\ vec(C_2) \end{pmatrix}. \tag{2.11}$$

Remark 2.3. If the condition (1) or the condition (2) of Remark 2.2 satisfies, then $\text{rank}(\Gamma) = mn$. Moreover, if $\text{rank}(\Gamma, b) \neq \text{rank}(\Gamma)$, then the system (1.4) is inconsistent.

Lemma 2.4. *For any initial matrix X_0, we have*

$$\begin{aligned} \|\tilde{R}_{k+1}\| &= \left\| A_1^T (C_1 - A_1 X_{k+1} B_1) B_1^T + A_2^T (C_2 - A_2 X_{k+1} B_2) B_2^T \right\| \\ &= \min_{X \in \mathscr{T}_k} \left\| A_1^T (C_1 - A_1 X B_1) B_1^T + A_2^T (C_2 - A_2 X B_2) B_2^T \right\|, \end{aligned} \tag{2.12}$$

where X_{k+1} is generated by Algorithm 2.2 *at the (k+1)-th iteration step and $\mathscr{T}_k$ denotes an affine space which has the following form*

$$\mathscr{T}_k = X_0 + \text{span}\{P_0, P_1, \cdots, P_k\}, \tag{2.13}$$

where P_k is generated by Algorithm 2.2 *at the k-th iteration step.*

Proof. According to the expression in Eq. (2.13), for any $X \in \mathscr{T}_k$, there exists scalars $\beta_0, \beta_1, \cdots, \beta_k$ such that

$$X = X_0 + \sum_{s=0}^{k} \beta_s P_s. \tag{2.14}$$

We introduce a scalar function $f(\beta_0, \beta_1, \cdots, \beta_k)$ by formula

$$f(\beta_0, \beta_1, \cdots, \beta_k) = \left\| A_1^T (C_1 - A_1 X B_1) B_1^T + A_2^T (C_2 - A_2 X B_2) B_2^T \right\|^2. \tag{2.15}$$

Substituting Eq. (2.14) into Eq. (2.15), we have

$$\begin{aligned}
f(\beta_0,\beta_1,\cdots,\beta_k) &= \left\|A_1^T(C_1-A_1XB_1)B_1^T+A_2^T(C_2-A_2XB_2)B_2^T\right\|^2\\
&=\left\|A_1^T\Big[C_1-A_1(X_0+\sum_{s=0}^{k}\beta_sP_s)B_1\Big]B_1^T+A_2^T\Big[C_2-A_2(X_0+\sum_{s=0}^{k}\beta_sP_s)B_2\Big]B_2^T\right\|^2\\
&=\left\|A_1^T(C_1-A_1X_0B_1)B_1^T+A_2^T(C_2-A_2X_0B_2)B_2^T-\sum_{s=0}^{k}\beta_s(A_1^TA_1P_sB_1B_1^T+A_2^TA_2P_sB_2B_2^T)\right\|^2\\
&=\left\|\tilde{R}_0-\sum_{s=0}^{k}\beta_sQ_s\right\|^2\\
&=\|\tilde{R}_0\|^2+\sum_{s=0}^{k}\beta_s^2\|Q_s\|^2-2\sum_{s=0}^{k}\beta_s\langle\tilde{R}_0,Q_s\rangle,
\end{aligned}$$

where the last equality use Lemma 2.1. Clearly, the function $f(\beta_0,\beta_1,\cdots,\beta_k)$ is continuous and differentiable with respect to the variables $\beta_0,\beta_1,\cdots,\beta_k$. Next, we minimize the function $f(\beta_0,\beta_1,\cdots,\beta_k)$. Obviously, the minimum of this function occurs when

$$\frac{\partial f(\beta_0,\beta_1,\cdots,\beta_k)}{\beta_s}=2\beta_s\|Q_s\|^2-2\langle\tilde{R}_0,Q_s\rangle=0. \tag{2.16}$$

Solving Eq. (2.16) gives

$$\beta_s=\frac{\langle\tilde{R}_0,Q_s\rangle}{\langle Q_s,Q_s\rangle}=\frac{\langle\tilde{R}_s,Q_s\rangle}{\langle Q_s,Q_s\rangle}=\alpha_s,\quad s=0,1,2,\cdots,k, \tag{2.17}$$

where the second equality use the second claim of Lemma 2.2. This implies that when

$$X_{k+1}=X_0+\sum_{s=0}^{k}\alpha_sP_s,$$

the matrix X_{k+1} minimizes the residual $\|\tilde{R}_{k+1}\|$ in the affine space $\mathscr{T}_k$. This completes the proof. □

According to Algorithm 2.2, if matrix equations (1.4) is inconsistent, then we will obtain the least squares solution. That is, we have the following conclusion.

Theorem 2.1. *Consider Algorithm* 2.2*. If* $\|\tilde{R}_k\|=0$*, then the matrix* X_k *is the least squares solution of the matrix equations* (1.4).

Proof. If $\|\tilde{R}_k\|=0$, then $\tilde{R}_k=0$. According to Algorithm 2.2, we have

$$R_k^{(1)}=C_1-A_1X_kB_1,\qquad R_k^{(2)}=C_2-A_2X_kB_2, \tag{2.18a}$$
$$\tilde{R}_k=A_1^TR_k^{(1)}B_1^T+A_2^TR_k^{(2)}B_2^T=0. \tag{2.18b}$$

Substituting Eq. (2.18a) into Eq. (2.18b) gives

$$\tilde{R}_k = A_1^T(C_1 - A_1X_kB_1)B_1^T + A_2^T(C_2 - A_2X_kB_2)B_2^T = 0. \tag{2.19}$$

Simplifying Eq. (2.19) yields

$$A_1^TA_1X_kB_1B_1^T + A_2^TA_2X_kB_2B_2^T = A_1^TC_1B_1^T + A_2^TC_2B_2^T. \tag{2.20}$$

By using vector operator and the Kronecker product, the relation (2.20) can be written as

$$\begin{aligned}
&\big(B_1 \otimes A_1^T\big)vec(C_1) + \big(B_2 \otimes A_2^T\big)vec(C_2)\\
=&\big(B_1B_1^T \otimes A_1^TA_1\big)vec(X_k) + \big(B_2B_2^T \otimes A_2^TA_2\big)vec(X_k)\\
=&\big(B_1 \otimes A_1^T\big)\big(B_1^T \otimes A_1\big)vec(X_k) + \big(B_2 \otimes A_2^T\big)\big(B_2^T \otimes A_2\big)vec(X_k).
\end{aligned} \tag{2.21}$$

With these preparations, Eq. (2.21) can be rewritten in detail as

$$\begin{aligned}
&\begin{pmatrix} B_1 \otimes A_1^T & B_2 \otimes A_2^T \end{pmatrix} \begin{pmatrix} B_1^T \otimes A_1 \\ B_2^T \otimes A_2 \end{pmatrix} vec(X_k)\\
=&\begin{pmatrix} B_1 \otimes A_1^T & B_2 \otimes A_2^T \end{pmatrix} \begin{pmatrix} vec(C_1) \\ vec(C_2) \end{pmatrix}.
\end{aligned}$$

Denote

$$\Gamma = \begin{pmatrix} B_1^T \otimes A_1 \\ B_2^T \otimes A_2 \end{pmatrix}, \quad b = \begin{pmatrix} vec(C_1) \\ vec(C_2) \end{pmatrix}, \quad x(k) = vec(X_k).$$

Then

$$\Gamma^T\Gamma x(k) = \Gamma^T b. \tag{2.22}$$

Since the equations $\Gamma x = b$ is equivalent to the matrix equations (1.4) and Eq. (2.22) is the normal equations of $\Gamma x = b$, the matrix X_k is the least squares solution of the matrix equations (1.4). Thus the proof is completed. □

Theorem 2.2. *If* $B_1B_1^T \otimes A_1^TA_1 + B_2B_2^T \otimes A_2^TA_2$ *is an positive definite matrix, then for any initial iterative matrix* X_0*, the least squares solution of the system* (1.4) *can be derived in at most mn iteration steps by Algorithm* 2.2.

Proof. Assume that $\tilde{R}_k \neq 0$ for $k = 0,1,2,\cdots,mn-1$. It follows from Lemma 2.3 that $Q_k \neq 0$, $P_k \neq 0$, $k = 0,1,\cdots,mn-1$. Then Q_{mn}, P_{mn} can be derived by Algorithm 2.2. According to Lemma 2.1, we know that $\langle Q_i, Q_j\rangle = 0$ for all $i,j = 0,1,\cdots,mn-1$, $i \neq j$. So the matrix sequence of $Q_0, Q_1,\cdots,Q_{mn-1}$ is an orthogonal basis of the linear space $R^{m\times n}$.

Moreover, according to Lemma 2.4, the matrix X_{mn} minimizes the residual $\|\tilde{R}_{mn}\|$ in the affine space $\mathscr{T}_{mn-1}$, that is

$$\begin{aligned}\|\tilde{R}_{mn}\|^2 =& \min_{X\in\mathscr{T}_{mn-1}} \left\|A_1^T(C_1 - A_1XB_1)B_1^T + A_2^T(C_2 - A_2XB_2)B_2^T\right\|^2 \\ =& \min_{\beta_s} \left\|\tilde{R}_0 - \sum_{s=0}^{mn-1} \beta_s Q_s\right\|^2.\end{aligned}$$

Hence $\|\tilde{R}_{mn}\| = 0$, which completes the proof. □

3. The least Frobenius norm least squares solution

When the system (1.4) is inconsistent, its least squares solution is not unique. Therefore, we need to find the unique least Frobenius norm least squares solution. First, we introduce the following lemmas.

Lemma 3.1. ([35]) *Suppose that $A \in R^{m\times n}$, $c \in R^m$ and the consistent system of linear equations $Ax = c$ has a solution $x^* \in \mathscr{R}(A^T)$, then x^* is the unique least Frobenius norm solution of $Ax = c$.*

Lemma 3.2. ([35]) *Suppose that $A \in R^{m\times n}$, $c \in R^m$ and the system of linear equations $Ax = c$ is inconsistent. If $x^* \in \mathscr{R}(A^T)$, then x^* is the unique least Frobenius norm least squares solution of $Ax = c$.*

Theorem 3.1. *Suppose that the system* (1.4) *is inconsistent, if we choose the initial matrix*

$$X_0 = A_1^T U_0 B_1^T + A_2^T V_0 B_2^T, \tag{3.1}$$

where $U_0 \in R^{p\times q}$, $V_0 \in R^{r\times s}$ are arbitrary matrices (especially, take $X_0 = 0 \in R^{m\times n}$), then the solution X^* given by Algorithm 2.2 *is the unique least Frobenius norm least squares solution of the system* (1.4).

Proof. If X_0 has the form of the relation (3.1), then by Algorithm 2.2, we have

$$\begin{aligned}X_1 =& X_0 + \alpha_0 P_0 = X_0 + \alpha_0 \tilde{R}_0 \\ =& A_1^T U_0 B_1^T + A_2^T V_0 B_2^T + \alpha_0\left(A_1^T R_0^{(1)} B_1^T + A_2^T R_0^{(2)} B_2^T\right) \\ =& A_1^T\left[U_0 + \alpha_0 R_0^{(1)}\right]B_1^T + A_2^T\left[V_0 + \alpha_0 R_0^{(2)}\right]B_2^T.\end{aligned}$$

Let $U_1 = U_0 + \alpha_0 R_0^{(1)}$ and $V_1 = V_0 + \alpha_0 R_0^{(2)}$. Then $X_1 = A_1^T U_1 B_1^T + A_2^T V_1 B_2^T$. Note that

$$\begin{aligned}P_1 =& \tilde{R}_1 + \beta_0^{(0)} P_0 \\ =& A_1^T R_1^{(1)} B_1^T + A_2^T R_1^{(2)} B_2^T + \beta_0^{(0)}\left(A_1^T R_0^{(1)} B_1^T + A_2^T R_0^{(2)} B_2^T\right) \\ =& A_1^T\left[R_1^{(1)} + \beta_0^{(0)} R_0^{(1)}\right]B_1^T + A_2^T\left[R_1^{(2)} + \beta_0^{(0)} R_0^{(2)}\right]B_2^T.\end{aligned}$$

Let $L_1 = R_1^{(1)} + \beta_0^{(0)} R_0^{(1)}$, $T_1 = R_1^{(2)} + \beta_0^{(0)} R_0^{(2)}$. Then $P_1 = A_1^T L_1 B_1^T + A_2^T T_1 B_2^T$. Hence, we have

$$\begin{aligned} X_2 =& X_1 + \alpha_1 P_1 \\ =& A_1^T U_1 B_1^T + A_2^T V_1 B_2^T + \alpha_1 \left[A_1^T L_1 B_1^T + A_2^T T_1 B_2^T\right] \\ =& A_1^T \left[U_1 + \alpha_1 L_1\right] B_1^T + A_2^T \left[V_1 + \alpha_1 T_1\right] B_2^T \\ =& A_1^T U_2 B_1^T + A_2^T V_2 B_2^T, \end{aligned}$$

where $U_2 = U_1 + \alpha_1 L_1$ and $V_2 = V_1 + \alpha_1 T_1$. By parity of reasoning, we can prove that

$$X_k = A_1^T U_k B_1^T + A_2^T V_k B_2^T,$$

where $U_k \in R^{p\times q}$, $V_k \in R^{r\times s}$. This fact together with Theorem 2.2 yields

$$X^* = A_1^T U B_1^T + A_2^T V B_2^T, \tag{3.2}$$

where $U \in R^{p\times q}$, $V \in R^{r\times s}$. The above equality together with the definition of Kronecker product yields

$$\begin{aligned} vec(X^*) =& (B_1 \otimes A_1^T) vec(U) + (B_2 \otimes A_2^T) vec(V) \\ =& \begin{pmatrix} B_1 \otimes A_1^T & B_2 \otimes A_2^T \end{pmatrix} \begin{pmatrix} vec(U) \\ vec(V) \end{pmatrix}. \end{aligned}$$

Hence

$$vec(X^*) \in \mathscr{R}\left(\begin{pmatrix} B_1^T \otimes A_1 \\ B_2^T \otimes A_2 \end{pmatrix}^T \right).$$

This together with Lemma 3.2 that X^* is the unique least Frobenius norm least squares solution of the system (1.4). The proof is completed. □

4. Numerical experiments

In this section, we report some numerical results to support our Algorithm 2.2. All of the tests were run on the Intel (R) Core (TM), where the CPU is 2.40 GHz and the memory is 8.0 GB, the programming language was MATLAB R2015a. In view of the influence of round-off errors, we regard a matrix T as the zero matrix if $\langle T, T\rangle < 10^{-9}$, where $\langle\cdot,\cdot\rangle$ denotes the inner product defined by (2.1).

For convenience, we demonstrate the effectiveness of Algorithm 2.2 from the residual of matrix equations (1.4) (denoted by 'Err(k)') and the residual of normal equations (2.22) (denoted by 'Frr(k)'). Here, 'Err(k)' and 'Frr(k)' are defined as

$$\begin{aligned} &Err(k) = \|C_1 - A_1 X_k B_1\|^2 + \|C_2 - A_2 X_k B_2\|^2, \\ &Frr(k) = \|\tilde{R}_k\| = \|A_1^T C_1 B_1^T + A_2^T C_2 B_2^T - A_1^T A_1 X_k B_1 B_1^T - A_2^T A_2 X_k B_2 B_2^T\|, \end{aligned} \tag{4.1}$$

where $\tilde{R}_k$ is generated by Algorithm 2.2.

Example 4.1. We consider Eq. (1.4) with the following matrices:

$$A_1 = \begin{pmatrix} 12.1577 & 8.9748 & 5.8313 \\ 1.3548 & 7.9965 & 6.9825 \\ 3.3212 & 6.1529 & 3.2933 \end{pmatrix}, \quad B_1 = \begin{pmatrix} 8.2788 & 3.3999 & 5.8149 \\ 0.3207 & 11.4671 & 9.3768 \\ 8.2714 & 2.4607 & 3.4779 \end{pmatrix},$$

$$A_2 = \begin{pmatrix} 4.3601 & 2.3787 & 8.5835 \\ 7.8889 & 5.4365 & 6.9820 \\ 0.9240 & 1.0482 & 10.3374 \end{pmatrix}, \quad B_2 = \begin{pmatrix} 9.5053 & 6.6178 & 0.1976 \\ 5.1627 & 4.1757 & 9.6429 \\ 3.2639 & 1.4782 & 12.7037 \end{pmatrix},$$

$$C_1 = \begin{pmatrix} 3.5398 & 5.9863 & 6.1785 \\ 0.2062 & 4.1403 & 0.7021 \\ 6.8148 & 7.9625 & 3.6928 \end{pmatrix}, \quad C_2 = \begin{pmatrix} 4.2386 & 2.9019 & 4.2779 \\ 4.6741 & 10.5454 & 2.6719 \\ 6.5669 & 5.5812 & 10.5374 \end{pmatrix}.$$

Choosing the arbitrary initial matrix X_0, such as

$$X_0 = \begin{pmatrix} 4.6157 & 4.3330 & 1.7898 \\ 7.1564 & 11.8424 & 6.3333 \\ 5.7774 & 3.9305 & 9.2400 \end{pmatrix}.$$

We obtain the least squares solution after 10 iterations by using Algorithm 2.2:

$$X_{10} = \begin{pmatrix} 0.1815 & 0.0004 & -0.1684 \\ -0.1652 & -0.0127 & 0.2015 \\ -0.0053 & 0.0905 & 0.0022 \end{pmatrix}.$$

At this moment, Err(k), Frr(k) and the norm of X_k are

$$\text{Err}(10) = 119.1892, \quad \text{Frr}(10) = 3.1895^{-10}, \quad \|X_{10}\| = 0.3709.$$

If we choose the initial matrices X_0 as the form of (3.1), we can get the least Frobenius norm least squares solution of Example 4.1. Especially, we can choose $X_0 = 0^{3\times 3}$. After 10 iteration steps, we obtain the least Frobenius norm least squares solution X_{10} and

$$\text{Err}(10) = 119.1892, \quad \text{Frr}(10) = 4.4236^{-13}, \quad \|X_{10}\| = 0.3709.$$

The relationship between the number of iterations and Frr(k) is shown in Fig. 1.

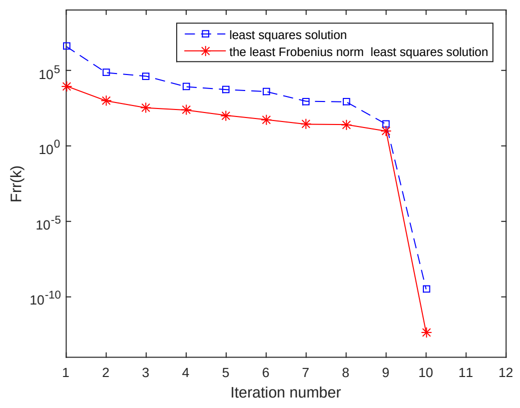

Figure 1: The relationship between the number of iterations and the norm of $\mathrm{Frr}(k)$ for Example 4.1.

Example 4.2. We consider Eq. (1.4) with the following matrices:

$$A_1 = \begin{pmatrix} 10.2594 & 0.4182 & 3.5446 & 6.7664 \\ 1.3787 & 5.0694 & 4.1063 & 9.8830 \\ 2.1780 & 6.1644 & 13.8435 & 7.6683 \\ 1.8214 & 9.3966 & 9.4558 & 7.3670 \end{pmatrix}, \quad B_1 = \begin{pmatrix} 9.6238 & 6.8018 & 6.0264 \\ 2.4417 & 8.2785 & 7.5052 \\ 2.9551 & 4.1159 & 8.8353 \end{pmatrix},$$

$$A_2 = \begin{pmatrix} 2.1773 & 3.0891 & 7.8287 & 0.0980 \\ 1.2565 & 7.2610 & 6.9379 & 8.4321 \end{pmatrix}, \quad B_2 = \begin{pmatrix} 12.2233 & 3.7819 & 2.2428 \\ 7.7095 & 10.0434 & 2.6905 \\ 0.4266 & 7.2951 & 9.7303 \end{pmatrix},$$

$$C_1 = \begin{pmatrix} 5.5179 & 7.1957 & 3.4645 \\ 5.8357 & 9.9616 & 8.8654 \\ 5.1182 & 3.5453 & 4.5469 \\ 0.8259 & 9.7126 & 4.1343 \end{pmatrix}, \quad C_2 = \begin{pmatrix} 4.7749 & 2.3644 & 8.2964 \\ 6.2372 & 1.7712 & 7.6692 \end{pmatrix}.$$

Choosing the arbitrary initial matrices X_0, such as

$$X_0 = \begin{pmatrix} 9.3448 & 4.8976 & 0.4417 \\ 1.0789 & 1.9325 & 5.5730 \\ 1.8223 & 8.9589 & 7.7250 \\ 0.9910 & 0.9909 & 3.1194 \end{pmatrix}.$$

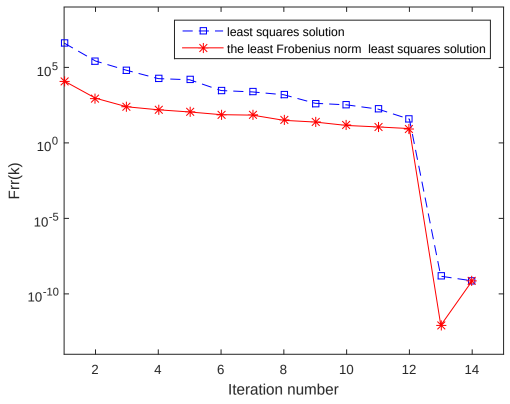

Figure 2: The relationship between the number of iterations and the norm of $\mathrm{Frr}(k)$ for Example 4.2.

We use Algorithm 2.2 and after 14 iterations, we obtain the least squares solution:

$$X_{14} = \begin{pmatrix} 0.0079 & 0.1080 & -0.0831 \\ -0.0700 & 0.1450 & -0.0317 \\ 0.0362 & -0.0981 & 0.0743 \\ 0.0606 & -0.0195 & 0.0120 \end{pmatrix}.$$

At this moment, Err(k), Frr(k) and the norm of X_k are

$$\mathrm{Err}(14) = 147.5996, \quad \mathrm{Frr}(14) = 7.3182^{-10}, \quad \|X_{14}\| = 0.2573.$$

If we choose the initial matrix X_0 as the form of (3.1), we can get the least Frobenius norm least squares solution of Example 4.2. Especially, we can choose $X_0 = 0^{4\times 3}$. Using Algorithm 2.2 and only 13 iteration numbers, we obtain the least Frobenius norm least squares solution:

$$X_{13} = \begin{pmatrix} 0.0079 & 0.1080 & -0.0831 \\ -0.0700 & 0.1450 & -0.0317 \\ 0.0362 & -0.0981 & 0.0743 \\ 0.0606 & -0.0195 & 0.0120 \end{pmatrix}.$$

At this moment, Err(k), Frr(k) and the norm of X_k are

$$\mathrm{Err}(13) = 147.5996, \quad \mathrm{Frr}(13) = 8.2875^{-13}, \quad \|X_{13}\| = 0.2573.$$

The relationship between the number of iterations and Frr(k) is shown in Fig. 2.

Example 4.3. In this example, we compare our algorithm (Extended GCR) with the extended LSQR method (LSQR-M) [38] and the Algorithm 2.1 in [6]. We consider Eq. (1.4)

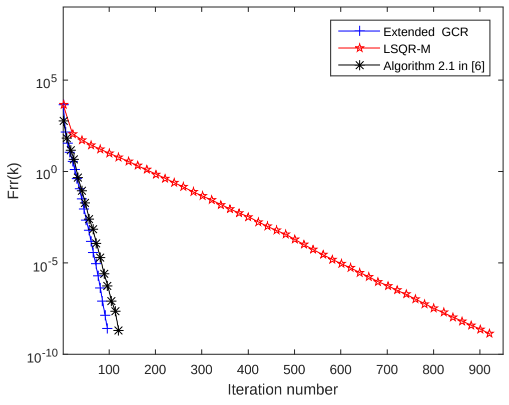

Figure 3: The residual norms Frr(k) for Example 4.3.

with the following matrices:

$$\begin{aligned}
A_1 &= triu(rand(n,n),1)+diag(2+diag(rand(n)))\in R^{n\times n},\\
B_1 &= tril(rand(n,n),1)+diag(3+diag(rand(n)))\in R^{n\times n},\\
A_2 &= tril(rand(n,n),1)-diag(4+diag(rand(n)))\in R^{n\times n},\\
B_2 &= triu(rand(n,n),n)+diag(2.5+diag(rand(n)))\in R^{n\times n},\\
C_1 &= C_2 = rand(n,n)\in R^{n\times n}.
\end{aligned}$$

When $n=40$, the convergence curves of mentioned algorithms with $X_0=0$ are obtained for the Frobenius norm of the residual

$$Frr(k)=\left\|A_1^TC_1B_1^T+A_2^TC_2B_2^T-A_1^TA_1X_kB_1B_1^T-A_2^TA_2X_kB_2B_2^T\right\|.$$

From Fig. 3, it is observe that the iteration number of the GCR algorithm is much less than the other tested methods. Moreover, we list the iteration steps, the CPU time and the residual norm (Frr) in Table 1. From Table 1, we know that the CPU time of our algorithm is much less than LSQR-M.

Example 4.4. In this example, we compare our algorithm (Extended GCR) with the extended LSQR method (LSQR-M) [38] and the Algorithm 2.1 in [6]. We consider Eq. (1.4)

Table 1: Numerical results for Example 4.3.

	Iteration#	CPU	Residual norms (Frr)
Extended GCR	98	0.1312	2.0340e-13
LSQR-M [38]	928	0.3309	9.7583e-10
Algorithm 2.1 [6]	124	0.0877	9.1908e-10

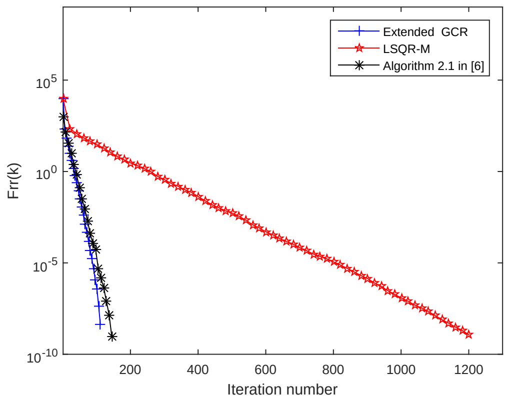

Figure 4: The residual norms Frr(k) for Example 4.4.

with the following matrices:

$$\begin{aligned}
A_1 &= triu(rand(n,n),2) - diag(6 + diag(rand(n))) \in R^{n\times n},\\
B_1 &= tril(rand(n,n),1) + diag(3 + diag(rand(n))) \in R^{n\times n},\\
A_2 &= rand(n,n) + diag(4 + diag(rand(n))) \in R^{n\times n},\\
B_2 &= rand(n,n) - diag(2.5 + diag(rand(n))) \in R^{n\times n},\\
C_1 &= C_2 = rand(n,n) \in R^{n\times n}.
\end{aligned}$$

When $n = 40$, the mentioned algorithms are applied with $X_0 = 0$ to obtain X_k. In Fig. 4, the convergence histories of mentioned methods are depicted where

$$Frr(k) = \left\|A_1^T C_1 B_1^T + A_2^T C_2 B_2^T - A_1^T A_1 X_k B_1 B_1^T - A_2^T A_2 X_k B_2 B_2^T\right\|.$$

From Fig. 4, it is easy to see that the iteration number of the GCR algorithm is much less than the other tested methods. Table 2 shows that the CPU time of our algorithm is much less than LSQR-M and our algorithm is efficient.

Table 2: Numerical results for Example 4.4.

	Iteration#	CPU	Residual norms (Frr)
Extended GCR	114	0.1735	6.3347e-10
LSQR-M [38]	1027	0.4226	9.9172e-10
Algorithm 2.1 [6]	144	0.0923	9.0964e-10

5. Conclusions

Based on the generalized conjugate residual (GCR) algorithm, we have constructed and analyzed Algorithm 2.2 for computing the least squares solution of the Sylvester matrix equations (1.4). When the system is inconsistent, we prove that the least squares solution can be obtained within finite iterative steps in the absence of round-off errors. Furthermore, we show that the least Frobenius norm least squares solution can be obtained by choosing a special kind of initial matrix. In addition, we present numerical examples, which demonstrate that Algorithm 2.2 is efficient.

Acknowledgments Supported by Fujian Natural Science Foundation (Grant No. 2016J01 005) and Strategic Priority Research Program of the Chinese Academy of Sciences (Grant No. XDB18010202).

References

[1] S. K. Mitra, *Common solutions to a pair of linear matrix equations $A_1XB_1 = C_1$, $A_2XB_2 = C_2$*, Proc. Cambridge Philos. Soc., 74 (1973), pp. 213–216.

[2] S. K. Mitra, *A pair of simultaneous linear matrix equations and a matrix programming problem*, Linear Algebra Appl., 131 (1990), pp. 97–123.

[3] A. Navarra, P. L. Odell and D. M. Young, *A representation of the general common solution to the matrix equations $A_1XB_1 = C_1$,$A_2XB_2 = C_2$ with applications*, Comput. Math. Appl., 41 (2001), pp. 929–935.

[4] Y. X. Yuan, *Least squares solutions of matrix equation $AXB = E$, $CXD = F$*, J. East China Shipbuilding Inst., 18 (2004), pp. 29–31.

[5] X. P. Sheng and G. L. Chen, *A finite iterative method for solving a pair of linear matrix equations $(AXB, CXD) = (E, F)$*, Appl. Math. Comput., 189 (2007) pp. 1350–1358.

[6] J. Cai and G. L. Chen, *An iterative algorithm for the least squares bisymmetric solutions of the matrix equations $A_1XB_1 = C_1$, $A_2XB_2 = C_2$*, Math. Comput. Model., 50 (2009), pp. 1237–1244.

[7] Z. Peng and H. Xin, *The reflexive least squares solutions of the general coupled matrix equations with a submatrix constraint*, Appl. Math. Comput., 225 (2013), pp. 425–445.

[8] M. Dehghan and M. Hajarian, *Efficient iterative method for solving the second-order Sylvester matrix equation $EVF^2 - AVF - CV = BW$*, IET Control Theory Appl., 3 (2009), pp. 1401–1408.

[9] M. Hajarian and M. Dehghan, *The generalized centro-symmetric and least squares generalized centro-symmetric solutions of the matrix equation $AYB + CY^TD = E$*, Math. Methods Appl. Sci., 34 (2011), pp. 1562–1579.

[10] M. Dehghan and M. Hajarian, *An iterative method for solving the generalized coupled Sylvester matrix equations over generalized bisymmetric matrices*, Appl. Math. Model., 34 (2010), pp. 639–654.

[11] M. Dehghan and M. Hajarian, *The general coupled matrix equations over generalized bisymmetric matrices*, Linear Algebra Appl., 432 (2010), pp. 1531–1552.

[12] M. Dehghan and M. Hajarian, *The generalized Sylvester matrix equations over the generalized bisymmetric and skew-symmetric matrices*, Int. J. Syst. Sci., 43 (2012), pp. 1580–1590.

[13] M. Dehghan and M. Hajarian, *On the generalized bisymmetric and skew-symmetric solutions of the system of generalized Sylvester matrix equations*, Linear Multilin. Algebra., 59 (2011), pp. 1281–1309.

[14] N. Huang and C. F. Ma, *Modified conjugate gradient method for obtaining the minimum-norm solution of the generalized coupled Sylvester-conjugate matrix equations*, Appl. Math. Modell., 40 (2016), pp. 1260–1275.

[15] Y. J. Xie, N. Huang and C. F. Ma, *Iterative method to solve the generalized coupled sylvester-transpose linear matrix equations over reflexive or anti-reflexive matrix*, Comput. Math. Appl., 67 (2014), pp. 2071–2084.

[16] N. Huang and C. F. Ma, *The modified conjugate gradient methods for solving a class of the generalized coupled Sylvester-transpose matrix equations*, Comput. Math. Appl., 67 (2014), pp. 1545–1558.

[17] Z. Peng, *The reflexive least squares solutions of the matrix equation $A_1X_1B_1+A_2X_2B_2+A_lX_lB_l = C$ with a submatrix constraint*, Numer. Algorithms., 64 (2013), pp. 455–480.

[18] M. Hajarian, *Matrix form of the CGS method for solving general coupled matrix equations*, Appl. Math. Lett., 34 (2014), pp. 37–42.

[19] M. Hajarian, *Developing Bi-CG and Bi-CR methods to solve generalized Sylvester-transpose matrix equations*, Int. J. Autom. Comput., 11 (2014), pp. 25–29.

[20] M. Hajarian, *Developing BiCOR and CORS methods for coupled Sylvester-transpose and periodic Sylvester matrix equations*, Appl. Math. Model., 39 (2015), pp. 6073–6084.

[21] M. Hajarian, *Matrix GPBiCG algorithms for solving the general coupled matrix equations*, IET Control Theory Appl., 9 (2015), pp. 74–81.

[22] M. Hajarian, *The generalized QMRCGSTAB algorithm for solving Sylvester-transpose matrix equations*, Appl. Math. Lett., 26 (2013), pp. 1013–1017.

[23] M. Dehghan and M. Hajarian, *An iterative algorithm for the reflexive solutions of the generalized coupled Sylvester matrix equations and its optimal approximation*, Appl. Math. Comput., 202 (2008), pp. 571–588.

[24] M. Dehghan and M. Hajarian, *The (R,S)-symmetric and (R,S)-skew symmetric solutions of the pair of matrix equations $A_1XB_1 = C_1$ and $A_2XB_2 = C_2$*, Bull. Iranian Math. Soc., 37 (2011), pp. 273–283.

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

[26] 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), pp. 41–50.

[27] F. Ding, System Identification-New Theory and Methods, Science Press, Beijing, 2013.

[28] M. Hajarian, *Extending the CGLS algorithm for least squares solutions of the generalized Sylvester-transpose matrix equations*, J. Franklin I., 353 (2016), pp. 1168–1185.

[29] L. Dai, Singular Control Systems, Lecture Notes in Control and Information Science, Springer, Berlin, 1989.

[30] G. R. Duan, *Eigenstructure assignment in descriptor linear systems via output feedback*, Int. J. Control., 72 (1999), pp. 345–364.

[31] F. L. Lewis, *Survey of linear singular systems*, Circuit Syst. Signal Process., 5 (1986), pp. 3–36.

[32] D. J. Inman and A. Kress, *Eigenstructure assignment algorithm for second order systems*, J. Guid. Control Dyn., 22 (1999), pp. 729–731.

[33] B. Zhou and G. R. Duan, *On the generalized Sylvester mapping and matrix equations*, Syst. Control Lett., 57 (2008), pp. 200–208.

[34] B. Kågström and P. van Dooren, *A generalized state-space approach for the additive decomposition of a transfer matrix*, J. Numer. Linear Algebra Appl., 1 (1992), pp. 165–181.

[35] A. Ben-Israel and T. N. E. Greville, Generalized Inverse: Theory and Applications, Wiley, New York, 2002.

[36] Q. W. Wang, J. H. Sun and S. Z. Li, *Consistency for bi(skew)symmetric solutions to systems of generalized Sylvester equations over a finite central algebra*, Linear Algebra Appl., 353 (2002), pp. 169–182.

[37] S. C. Eisenstat, H. C. Elman and M. H. Schultz, *Vriational iterative methods for nonsymmetric*

systems of linear equations, SIAM J. Numer. Anal. 20 (1983), pp. 345–357.

[38] *Y. B. Chen, Z. Y. Peng and T. J. Zhou, LSQR iterative common symmetric solutions to matrix equations $AXB = E$ and $CXD = F$,* Appl. Math. Comput., 217 (2010), pp. 230–236.

[39] M. HAJARIAN, *Solving the system of linear operator equations over generalized bisymmetric matrices,* T. I. Meas. Control., 36 (2014), pp. 541–550.

[40] M. HAJARIAN, *New finite algorithm for solving the generalized nonhomogeneous Yakubovich-Transpose matrix equations,* Asian. J. Control., 19 (2017), pp. 164–172.

[41] M. HAJARIAN, *Symmetric solutions of the coupled generalized Sylvester matrix equations via BCR algorithm,* J. Franklin I., 353 (2016), pp. 3233–3248.

[42] P. WANG, S. F. YUAN AND X. Y. XIE, *Least-squares Hermitian problem of complex matrix equation $(AXB, CXD) = (E, F)$,* Mathematische Annalen, 35(2) (2016), pp. 369–370.

[43] M. DEHGHAN AND M. HAJARIAN, *An iterative algorithm for solving a pair of matrix equations $AYB = E$, $CYD = F$ over generalized centro-symmetric matrices,* Comput. Math. Appl., 56 (2008), pp. 3246–3260.

[44] Z. Z. BAI, *On Hermitian and skew-Hermitian splitting iteration methods for continuous Sylvester equations,* J. Comput. Math., 29 (2011), pp. 185–198.

[45] M. K. ZAK AND F. TOUTOUNIAN, *Nested splitting conjugate gradient method for matrix equation $AXB = C$ and preconditioning,* Comput. Math. Appl., 66 (2013), pp. 269–178.