CONJUGATE GRADIENT ALGORITHM FOR A GENERALIZED SYLVESTER-TRANSPOSE MATRIX EQUATION

SARAWANEE CHOOMKLANG

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENT
FOR THE DEGREE OF MASTER OF SCIENCE IN APPLIED MATHEMATICS
DEPARTMENT OF MATHEMATICS SCHOOL OF SCIENCE
KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG
2021

This material is reserved for educational use only, not allowed for commercial use.
Forbidden to modify the content, and cite the document when use.

KMITL-2021-SC-M-001-038

COPYRIGHT 2021
SCHOOL OF SCIENCE
KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG
This thesis is for educational use only, not allowed for commercial use.
Forbidden to modify the content, and cite the document when use.

Thesis Title	Conjugate Gradient Algorithm for a Generalized Sylvester-Transpose Matrix Equation
Student Name	Miss Sarawanee Choomklang
Student ID	62605023
Degree	Master of Science in Applied Mathematics
Department	Mathematics
Year	2021
Thesis Advisor	Assoc. Prof. Dr. Pattrawut Chansangiam

Abstract

In this research, we propose a conjugate gradient algorithm to solve the generalized Sylvester-transpose matrix equation. This algorithm is usable for the matrix equation whose matrix coefficients constitute a symmetric matrix. the proposed algorithm aims to construct a finite sequence of approximated solutions for the matrix equation, based on orthogonality of associated residual matrices. The last iteration step will provide the exact solution of the equation.

Keywords : generalized Sylvester-transpose matrix equation, Kronecker product, matrix norm, orthogonality

This material is reserved for educational use only, not allowed for commercial use.
Forbidden to modify the content, and cite the document when use.

Acknowledgements

I would like to express my deepest gratitude to my thesis advisor, Assoc. Prof. Dr. Pattrawut Chansangiam, for his precious assistance and advice throughout the course of this research. This thesis would not have been completed without all the support and guidance that I have always received from him.

In addition, I would also like to acknowledge my thesis committee members, Prof. Dr. Jessada Tariboon and Asst. Prof. Dr. Kanchana Kumnungkit for their invaluable comments and suggestion.

Finally, I would like to thank my family for always being supportive of my education, and thank my friends for their encouragement throughout academic years.

Sarawanee Choomklang

This material is reserved for educational use only, not allowed for commercial use.
Forbidden to modify the content, and cite the document when use.

Table of Contents

This material is reserved for educational use only, not allowed for commercial use.
Forbidden to modify the content, and cite the document when use.

List of Tables

Table **Page**

This material is reserved for educational use only, not allowed for commercial use.
Forbidden to modify the content, and cite the document when use.

Chapter 1

Introduction

1.1 Inception and significance of research

Linear matrix equations are famous problems often show up in mathematics and engineering, especially control theory, system theory, see e.g. [11], [18]. There are many well-known types of matrix equations such as the following:

$$AX - XA^T = C, \quad \text{Lyapunov matrix equation} \tag{1.1}$$

$$E^TAX + AXE^T = -C, \quad \text{a generalized Lyapunov matrix equation} \tag{1.2}$$

$$X + AXB = C, \quad \text{Stein matrix equation} \tag{1.3}$$

$$AX + XB = C, \quad \text{Sylvester matrix equation} \tag{1.4}$$

$$AXB + CXD = E, \quad \text{a generalized Sylvester matrix equation} \tag{1.5}$$

$$AX + X^TB = C, \quad \text{Sylvester-transpose matrix equation} \tag{1.6}$$

$$AXB + CX^TD = E, \quad \text{a generalized Sylvester-transpose matrix equation.} \tag{1.7}$$

Here, A, B, C, D, and E are given matrices and X is an unknown matrix to be found. An ordinary method for solving a linear matrix equation is to reduce it to an equivalent linear system $Ux = v$ by taking the vector operator. If U is a nonsingular matrix, then the matrix equation has a unique solution. Since U is forming by the Kronecker multiplication, the size of U may be enormous. For instance

$$(B^T \otimes A + D^T \otimes C)\operatorname{vec} X = \operatorname{vec} E$$

is equivalent to (1.5) by taking the vector operator. Here, the symbol $\otimes$ denotes the Kronecker product and the operation vec$(\cdot)$ is the vector operator. Whenever the matrix has enormous size, the computation will consume a long time and many memories.

In the recent decade, many researchers developed several iterative methods for solving the matrix equation (1.1)-(1.7). The obviously advantage of iteration is not necessary use memories as massive as the regular method in a big dimension case. We can classify such iterative method into two groups. The first group are stationary iterative methods; see e.g. [13, 6, 9, 17, 2, 16]. These methods aim to construct a sequence of approximated solution converging to the exact solution. The second group, known as Krylov subspace methods, are semi-direct methods aiming to construct an orthogonal basis for the associated vector space. The basis consists of (residual) vectors pointing in direction so that the approximated solutions fastest approach to the exact solution.

In fact, the equations (1.1)-(1.7) are special cases of the following generalized

This material is reserved for educational use only, not allowed for commercial use.
Forbidden to modify the content, and cite the document when use.

Sylvester-transpose matrix equation:

$$\sum_{i=1}^{p} A_i X B_i + \sum_{j=1}^{q} C_j X^T D_j = E \tag{1.8}$$

Recently, the matrix equation has been discussed extensively. In 2014, Hajarian presented a bi-conjugate gradient and bi-conjugate residual methods for the generalized Sylvester-transpose matrix equation [3]. The Sylvester matrix equation and the generalized Sylvester matrix equation has been solved by Yiqin Lin in 2006 [12]. Yi-Fen Ke and Chang-Feng Ma propose a nested splitting conjugate gradient method and a preconditioned nested splitting conjugate gradient method for a large sparse coefficient of the equation (1.5) [7]. To Solve a pair of matrix equations over generalized centrosymmetric matrix, Hajarian and Dehghan propose iterative algorithm in 2008 [1]. Liang and Liu propose two iterative algorithms for a minimum-norm and a least square solution of a pair of generalized Sylvester-conjugate matrix equation [10]. Wang, Cheng, and Wei propose two iterative algorithms for the equation (1.7) in 2007 by applying conjugate gradient method [15]. In [8], Lei and Liao propose a minimal residual algorithm for the inconsistent matrix equation $AXB = C$ over symmetric matrices in 2007. The CGS algorithm and the extended CGS algorithm for the general couple matrix equation has been proposed by Hajarian [4].

In this paper, we intend to present a new choice of iterative algorithm to solve the matrix equation (1.8) when the equation is consistent and has a unique solution. Our method is based on conjugate gradient method idea. The propose method is applicable as long as the coefficient matrix must be symmetric and invertible. The method aims to construct a finite sequence of approximated solutions for the matrix equation, based on orthogonality of associated residual matrices. The final step of iteration comes out with the exact solution of the equation.

1.2 Objectives

1) Propose a conjugate gradient iterative algorithm for solving the generalized Sylvester-transpose matrix equation

$$\sum_{i=1}^{p} A_i X B_i + \sum_{j=1}^{q} C_j X^T D_j = E. \tag{1.9}$$

2) Investigate the convergent property of the proposed algorithm.

3) Provide some numerical examples to show the capability and performance of the proposed algorithm.

1.3 Scope

This material is reserved for educational use only, not allowed for commercial use.
Forbidden to modify the content, and cite the document when use.

We consider the matrix equation (1.9) when all coefficient matrices $A_1, \ldots, A_p$, $B_1, \ldots, B_p, C_1, \ldots, C_q, D_1, \ldots, D_q$ and E are $n \times n$ real matrices. We eould like to find

an unknown matrix $X \in \mathbb{R}^{n\times n}$. We assume that $\sum_{i=1}^{p}\left(B_i^T \otimes A_i\right) + \sum_{j=1}^{q}\left(D_j^T \otimes C_j\right) P(n,n)$ is symmetric.

1.4 Benefits

Obtain a new algorithm for finding a solution of the generalized Sylvester-transpose matrix equation within finite iteration step.

1.5 Research methodology

1) Study fundamentals of matrix algebra and matrix analysis.
2) Study the Kronecker product of matrices, the vector operator, and related topics from textbooks.
3) Study the direct method for solving linear matrix equations from textbooks and research papers.
4) Study conjugate gradient algorithms for solving linear matrix equations from research paper.
5) Propose a new conjugate gradient iterative algorithm for solving the generalized Sylvester-transpose matrix equation.
6) Analyze and verify the convergent property of the proposed algorithm.
7) Provide some numerical examples.
8) Discuss and conclude the result.
9) Write the thesis.

Table 1.1: The research schedule

Activity	Time frame							
	2019	2020				2021		
	Aug. - Dec.	Jan. - Mar.	Apr. - Jun.	Jul. - Sep.	Oct. - Dec.	Jan. - Feb.	Mar. - Apr.	May. - Jul.
Step 1	←——→							
Step 2	←————→							
Step 3		←————→						
Step 4			←————→					
Step 5					←————→			
Step 6						←————→		
Step 7							←————→	
Step 8							←————→	
Step 9								←——→

This material is reserved for educational use only, not allowed for commercial use.
Forbidden to modify the content, and cite the document when use.

Chapter 2

Preliminaries

In this chapter, we recall fundamental tools for solving linear matrix equations and analysing matrix iterative algorithms. For each $m, n \in \mathbb{N}$, we denote the set of $m \times n$ real matrices by $\mathbb{R}^{m\times n}$. When $n = 1$, the set $\mathbb{R}^m = \mathbb{R}^{m\times 1}$ is the set of m-dimensional (column) vectors.

2.1 Fundamental tools from matrix analysis

Recall that the Kronecker product is a matrix product defined for two matrices of arbitrary dimensions. Indeed, the Kronecker product of $A \in \mathbb{R}^{m\times n}$ and $B \in \mathbb{R}^{p\times q}$ is defined to be

$$A \otimes B = \begin{pmatrix} a_{11}B & \cdots & a_{1n}B \\ \vdots & \ddots & \vdots \\ a_{m1}B & \cdots & a_{mn}B \end{pmatrix} \in \mathbb{R}^{mp\times nq}.$$

Lemma 2.1. (see e.g. [5]) The following properties hold for any compatible matrices A, B, C:

1. $(A \otimes B)^T = A^T \otimes B^T$,
2. $(A + B) \otimes C = A \otimes C + B \otimes C$,
3. $A \otimes (B + C) = A \otimes B + A \otimes C$.

Recall that the commutation matrix $P(m, n)$ is a permutation matrix defined by

$$P(m,n) = \sum_{i=1}^{m}\sum_{j=1}^{n} E_{ij} \otimes E_{ij}^T \in \mathbb{R}^{mn\times mn} \tag{2.1}$$

where each $E_{ij} \in \mathbb{R}^{m\times n}$ has entry 1 in position (i, j) and all other entries are 0. Moreover, $P(m, n) = P(n, m)^T = P(n, m)^{-1}$. An important property of $P(m, n)$ is as follows:

Lemma 2.2. (see e.g. [5]) For any $A \in \mathbb{R}^{m\times n}$ and $B \in \mathbb{R}^{p\times q}$, we have

$$B \otimes A = P(m,p)^T (A \otimes B) P(n,q). \tag{2.2}$$

The vector operator vec$(\cdot)$ is a linear operator that maps any $A = [a_{ij}] \in \mathbb{R}^{m\times n}$ to the column vector

$$\text{vec}(A) = [a_{11} \ldots a_{m1}\, a_{12} \ldots a_{m2} \ldots a_{1n} \ldots a_{mn}]^T \in \mathbb{R}^{mn}.$$

Lemma 2.3. (see e.g. [5]) For any matrices A, B, C of compatible dimensions, we have

$$\text{vec}(ABC) = (C^T \otimes A)\,\text{vec}(B). \tag{2.3}$$

This material is reserved for educational use only, not allowed for commercial use.
Forbidden to modify the content, and cite the document when use.

Lemma 2.4. (see e.g. [5]) For any matrix $X \in \mathbb{R}^{m\times n}$, we have

$$\text{vec}(X^T) \;=\; P(m,n)\,\text{vec}(X). \tag{2.4}$$

Recall that the trace of a matrix $A = [a_{ij}] \in \mathbb{R}^{n\times n}$ is defined to be the sum of all elements on the main diagonal of A, that is,

$$\text{tr}\, A \;=\; \sum_{i=1}^{n} a_{ii}.$$

Lemma 2.5. (see e.g. [5]) For any matrices A, B, X, Y of compatible dimensions, we have

$$\text{tr}\left(A^T Y^T B X\right) \;=\; \left(\text{vec}\,(Y)\right)^T (A \otimes B)\,\text{vec}\,(X)\,. \tag{2.5}$$

Definition 2.6. The Frobenius norm of $A \in \mathbb{R}^{m\times n}$ is defined by

$$\|A\| \;=\; \left(\sum_{i=1}^{m}\sum_{j=1}^{n} a_{ij}^2\right)^{\frac{1}{2}} \;=\; \left(\text{tr}\left(A^T A\right)\right)^{\frac{1}{2}}.$$

2.2 Conjugate gradient method for linear systems

The conjugate gradient method is the most famous iterative method in Krylov subspace. Invented by Magnus Hestenes and Eduard Stiefel in 1952.

Consider linear system

$$Ax \;=\; b \tag{2.6}$$

in which the coefficient matrix A is positive definite as occurs in many applications. The orthogonality with respect to the induced inner product

$$\langle u, v\rangle = u^T A v \tag{2.7}$$

is very natural. Vectors that are orthogonal with respect to (2.7) are called conjugate vector. That explain the half name of this method. 'Gradient' from the minimization principle. The solution x^* is the unique minimizer of the quadratic function

$$p(x) = \frac{1}{2}x^T A x - x^T b \tag{2.8}$$

The gradient vector $\nabla p(x)$ of the function points in the direction of its steepest increase at the point and $-\nabla p(x)$ points in the direction of steepest decrease. Thus

$$-\nabla p(x) \;=\; b - Ax \;=\; r,$$

where r is known as the residual vector. Note that $r = 0$ if and only if $x = x^*$ such that x^* is a solution of linear system. Thus, each successive approximation x_k was obtained by the equation

$$x_{k+1} \;=\; x_k + t_k r_k, \qquad \text{where} \qquad r_k \;=\; b - Ax_k.$$

This material is reserved for educational use only, not allowed for commercial use.
Forbidden to modify the content, and cite the document when use.

The scalar t_k is represent the step length in direction as mentioned above. That is the solution x^* by successive approximation, with kth iteration step can be written

$$x_{k+1} = x_k + t_{k+1}u^{k+1}, \qquad \text{where} \qquad x_k = t_1u_1 + \cdots + t_ku_k$$

where, the conjugate vectors $u_1, \ldots, u_n$ form a A-orthogonal basis.

Assume that initial guess x_0 for the solution. The residual vector $r_0 = b - Ax_0 = b$ present the direction of steepest decrease of $p(x)$ at the point x_0. Next, original guess was updated by moving in this direction, taking $u_1 = r_0 = b$. The next iteration step is $x_1 = x_0 + t_1u_1 = t_1u_1$, and choose the parameter t_1 so that the corresponding residual vector r_1 which orthogonal to r_0 that is

$$\begin{aligned} 0 = r_0^T r_1 &= \|r_0\|^2 - t_1 r_0^T A u_1 \\ &= \|r_0\|^2 - t_1\langle r_0, u_1\rangle \\ &= \|r_0\|^2 - t_1\langle u_1, u_1\rangle \end{aligned}$$

Notice that Euclidean norm is defined by $\|\cdot\|$, and $\langle u, v\rangle$ is inner product defined by (2.7). Hence,

$$t_1 = \frac{\|r_0\|^2}{\langle u_1, u_1\rangle_A}, \qquad \text{and} \qquad x_1 = x_0 + \frac{\|r_0\|^2}{\langle u_1, u_1\rangle_A}u_1$$

is a new approximation solution. Assume that $t_1 \neq 0$, since otherwise the residual $r_0 = 0$. It induce $x_0 = 0$ is the exact solution of the system, then there no reason to continue the procedure.

Next step, update x_1 by moving in the residual direction r_1, choose a direction u_2 which is conjugate (meaning A-orthogonal) to the first direction $u_1 = r_0$. So, setting $u_2 = r_1 + s_1u_1$, where s_1 is the parameter that determined by the orthogonality requirement

$$\begin{aligned} 0 &= \langle u_2, u_1\rangle_A \\ &= \langle r_1 + s_1u_1, u_1\rangle_A \\ &= \langle r_1, u_1\rangle_A + s_1\langle u_1, u_1\rangle_A. \end{aligned}$$

Therefore,

$$s_1 = -\frac{\langle r_1, u_1\rangle_A}{\langle u_1, u_1\rangle_A}.$$

Now, consider

$$\begin{aligned} \langle r_1, u_1\rangle_A &= r_1^T A u_1 \\ &= r_1^T\left(\frac{r_0 - r_1}{t_1}\right) \\ &= -\frac{1}{t_1}\|r_1\|^2, \end{aligned}$$

and

$$\langle u_1, u_1\rangle_A = u_1^T A u_1^T = \frac{1}{t_1}\|r_0\|^2.$$

This material is reserved for educational use only, not allowed for commercial use.
Forbidden to modify the content, and cite the document when use.

Then, the second conjugate direction is

$$u_2 = r_1 + s_1 u_1, \qquad \text{where} \qquad s_1 = \frac{\|r_1\|^2}{\|r_0\|^2}.$$

Next, update the new approximation solution

$$\begin{aligned} x_2 &= x_1 + t_2 u_2 \\ &= x_0 + t_1 u_1 + t_2 u_2 \\ &= t_1 u_1 + t_2 u_2 \end{aligned}$$

so as to make the corresponding residual vector

$$r_2 = b - Ax_2 = r_1 - t_2 A u_2.$$

Then, using the A-orthogonality of r_1 and r_2,

$$\begin{aligned} 0 &= r_1^T r_2 \\ &= \|r_1\|^2 - t_2 r_1^T A u_2 \\ &= \|r_1\|^2 - t_2 \langle u_2, u_2 \rangle. \end{aligned}$$

Therefore,

$$t_2 = \frac{\|r_1\|^2}{\langle u_2, u_2 \rangle}.$$

Again, assume that $t_2 \neq 0$, as otherwise $r_1 = 0$ and x_1 would be the exact solution, so the procedure would be discontinue.

Repeating this process, at the kth stage, there are already constructed the conjugate vector $u_1, \ldots, u_k$, and the approximate solution x_k. Then the next conjugate direction is given by

$$u_{k+1} = r_k + s_k u_k, \qquad \text{where} \qquad s_k = \frac{\|r_k\|^2}{\|r_{k-1}\|^2}$$

results from the A-orthogonality requirement: $\langle u_i, u_k \rangle = 0$ for $i < k$. The approximate solution update as follows:

$$x_{k+1} = x_k + t_{k+1} u_{k+1}$$

where

$$t_{k+1} = \frac{\|r_k\|^2}{\langle u_{k+1}, u_{k+1} \rangle} \qquad \text{and} \qquad r_{k+1} = b - Ax_{k+1} = r_k - t_{k+1} A u_{k+1}$$

as small as possible and orthogonal to r_k.

This material is reserved for educational use only, not allowed for commercial use.
Forbidden to modify the content, and cite the document when use.

Algorithm 1: A conjugate gradient algorithm for the linear system (2.6)

Start: Choose x_0, e.g. $x_0 = 0$
set $u_0 = 0$
for $k = 0$ **to** $n-1$ **do**
  set $r_k = b - Ax_k$;
  if $r_k = 0$ **then**
    print 'x_k is the exact solution' ; break
  else
    if $k = 0$ **then**
      set $u_1 = r_0$
      else
        set $u_{k+1} = r_k + \frac{\|r_k\|^2}{\|r_{k-1}\|^2} u_k$
      end
    end
  end
  set $x_{k+1} = x_k + \frac{\|r_k\|^2}{u_{k+1}^T A u_{k+1}} u_{k+1}$
  next k
end

2.3 Conjugate gradient algorithms for solving a generalized Sylvester-transpose equation

In 2007 [15], Minghui Wang, Xuehan Cheng, and Musheng Wei proposed two algorithms for solving the matrix equation

$$AXB + CX^TD = E. \tag{2.9}$$

They established two algorithms by conjugate gradient method. The first algorithm for the equation is consistent and a solution can be obtain within finite step for any initial matrix. The second algorithm for inconsistent equation. a least square solution can be obtain within finite step for any initial matrix.

The first algorithm for the consistent matrix equation as follows:

This material is reserved for educational use only, not allowed for commercial use.
Forbidden to modify the content, and cite the document when use.

Algorithm 2: A conjugate gradient algorithm for consistent matrix equation

For arbitrary initial matrix $X_1 \in R_{l\times n}$

Start: Set $R_1 = E - AX_1B - CX_1^TD$

$P_1 = A^TR_1B^T + DR_1^TC$

$Q_1 = P_1$;

for $k = 1, 2, \ldots$ **do**

  if $R_k = 0$ **then**

    X_k is the exact solution; break

  else

    $X_{k+1} = X_k + \frac{\|R_k\|^2}{\|Q_k\|^2}Q_k$

    $R_{k+1} = E - AX_{k+1}B - CX_{k+1}^TD$

    $P_{k+1} = A^TR_{k+1}B^T + DR_{k+1}^TC$

    $Q_{k+1} = P_{k+1} - \frac{\mathrm{tr}\left(P_{k+1}^TQ_k\right)}{\|Q_k\|^2}Q_k$

    next k

  end

end

To prove the convergent property, they show some properties as follow:

Lemma 2.7. ([15]) Suppose that the matrix equation (2.9) is consistent and the sequence $\{R_i\}$ and $\{Q_i\}$ are generated by Algorithm 2. For a natural number $k > 1$, such that $R_i \neq 0$ for all $i = 1, \ldots, k$, then $\{R_i\}_{i=1}^k$ and $\{Q_i\}_{i=1}^k$ are two sets of orthogonal matrices.

Theorem 2.8. ([15]) Suppose that the matrix equation (2.9) is consistent. The sequence $\{X_k\}$ generated by Algorithm 2. Then for any initial matrix $X_1 \in \mathbb{R}^{l\times n}$ converges to a exact solution of (2.9) in at most $ln + 1$ iteration steps.

When the matrix equation (2.9) is inconsistent, they consider the least square problem

$$\min_{X\in\mathbb{R}^{n\times p}} \|E - AXB - CX^TD\|. \tag{2.10}$$

To find a least square solution, they consider the equation equivalent to (2.10) that has the form

$$A^TAXBB^T + A^TCX^TDB^T + DB^TX^TA^TC + DD^TXC^TC = A^TEB^T + DE^TC. \tag{2.11}$$

Adopting the same idea of Algorithm 2 for equation (2.11), they propose algorithm as follows:

This material is reserved for educational use only, not allowed for commercial use.
Forbidden to modify the content, and cite the document when use.

Algorithm 3: A conjugate gradient algorithm for inconsistent matrix equation

For arbitrary initial matrix $X_1 \in R_{l\times n}$

Start: Set

$R_1 = A^TEB^T + DE^TC - A^T(AX_1B + CX_1^TD)B^T - D(B^TX_1^TA^T + D^TX_1C^T)C,$

$P_1 = A^T(AR_1B + CR_1^TD)B^T + D(B^TR_1^TA^T + D^TR_1C^T)C,$

$Q_1 = P_1;$

for $k = 1, 2, \ldots$ **do**

  if $R_k = 0$ **then**

    X_k is the exact solution; break

  else

$$X_{k+1} = X_k + \frac{\|R_k\|^2}{\|Q_k\|^2}Q_k$$

$$R_{k+1} = A^TEB^T + DE^TC - A^T(AX_{k+1}B + CX_{k+1}^TD)B^T - D(B^TX_{k+1}^TA^T + D^TX_{k+1}C^T)C,$$

$$P_{k+1} = A^T(AR_{k+1}B + CR_{k+1}^TD)B^T + D(B^TR_{k+1}^TA^T + D^TR_{k+1}C^T)C$$

$$Q_{k+1} = P_{k+1} - \frac{\mathrm{tr}\,(P_{k+1}^TQ_k)}{\|Q_k\|^2}Q_k$$

    next k

  end

end

To Prove the convergent property of Algorithm 3 is the same way that prove convergent property of Algorithm 2.

This material is reserved for educational use only, not allowed for commercial use.
Forbidden to modify the content, and cite the document when use.

Chapter 3

Conjugate Gradient Algorithm for a Generalized Sylvester-Transpose Matrix Equation

3.1 A direct method to solve a generalized Sylvester-transpose matrix equation

Consider the matrix equation (1.8) when A_i, B_i, C_j, D_j, $E \in \mathbb{R}^{n\times n}$ for any $i, j = 1, 2, \ldots, p$. We can apply the vector operator to Eq. (1.8), and we get

$$\begin{aligned}
\text{vec}\,(E) &= \text{vec}\left(\sum_{i=1}^{p} A_i X B_i + \sum_{j=1}^{q} C_j X^T D_j\right) \\
&= \sum_{i=1}^{p} \left(B_i^T \otimes A_i\right) \text{vec}(X) + \sum_{j=1}^{q} \left(D_j^T \otimes C_j\right) \text{vec}(X^T) \\
&= \sum_{i=1}^{p} \left(B_i^T \otimes A_i\right) \text{vec}(X) + \sum_{j=1}^{q} \left(D_j^T \otimes C_j\right) P\,(n,n)\,\text{vec}(X) \\
&= \left(\sum_{i=1}^{p} \left(B_i^T \otimes A_i\right) + \sum_{j=1}^{q} \left(D_j^T \otimes C_j\right) P(n,n)\right) \text{vec}(X)
\end{aligned} \tag{3.1}$$

Let us denote

$$M = \sum_{i=1}^{p} \left(B_i^T \otimes A_i\right) + \sum_{j=1}^{q} \left(D_j^T \otimes C_j\right) P(n,n), x = \text{vec}(X), b = \text{vec}(E). \tag{3.2}$$

Thus, Eq. (1.8) is equivalently transformed to a linear algebraic system $Mx = b$. Hence, Eq. (1.8) has a unique solution if and only if the matrix M is invertible. In this case, the solution is $x = M^{-1}b$, and we can get the unknown matrix using the injectively of the vector operator.

3.2 A conjugate gradient algorithm

In this section, we propose an iterative algorithm for solving the matrix equation (1.8). We suppose that the matrix M in invertible, so that Eq. (1.8) has a unique solution. To derive an iterative procedure, we apply the idea of a conjugate gradient algorithm. Indeed, for a consistent linear system

$$Ku = f \tag{3.3}$$

with $n \times n$ positive definite coefficient matrix K, we have Algorithm 1.

From now on, suppose that the matrix M is invertible and symmetric. We propose the following algorithm:

This material is reserved for educational use only, not allowed for commercial use.
Forbidden to modify the content, and cite the document when use.

Algorithm 4: A conjugate gradient algorithm for Eq. (1.8)

Start: Set $k = 0$

Choose X_0 e.g. $X_0 = 0$

$R_0 = E - \sum_{i=1}^{p} A_i X_0 B_i - \sum_{j=1}^{q} C_j X_0^T D_j$

for $k = 0$ **to** $n^2 - 1$ **do**

  if $R_k = 0$ **then**

    X_k is the exact solution; break

  else

    if $k = 0$ **then**

      $P_1 = R_0$

    else

      $P_{k+1} = R_k + \frac{\|R_k\|^2}{\|R_{k-1}\|^2} P_k$

    end

    $Q_{k+1} = \sum_{i=1}^{p} A_i P_{k+1} B_i + \sum_{j=1}^{q} C_j P_{k+1}^T D_j$

    $\alpha_{k+1} = \text{tr}\left(P_{k+1}^T Q_{k+1}\right)$

    $X_{k+1} = X_k + \frac{\|R_k\|^2}{\alpha_{k+1}} P_{k+1}$

    $R_{k+1} = E - \sum_{i=1}^{p} A_i X_{k+1} B_i - \sum_{j=1}^{q} C_j X_{k+1}^T D_j$

    next k

  end

end

From the main computation

$$X_{k+1} = X_k + \frac{\|R_k\|^2}{\alpha_{k+1}} P_{k+1},$$

we call X_{k+1} the next point, X_k the current point, and P_{k+1} the search direction. The step size $\|R_k\|^2/\alpha_{k+1}$ is chosen so that the set $\{R_i\}$ of residual matrices is orthogonal, as we shall see later. To avoid roundoff errors from computations, we may introduce a small tolerance error $\varepsilon > 0$ and use $\|R_k\| < \varepsilon$ to be the stopping rule, instead of the condition $R_k = 0$.

Next, we shall show that Algorithm 4 leads to the exact solution in at most $n^2 + 1$ iterations. We divide the proof into several lemmas as follows.

Lemma 3.1. Suppose that the sequence $\{R_i\}$ is generated by Algorithm 4. We have

$$R_{k+1} = R_k - \frac{\|R_k\|^2}{\alpha_{k+1}} Q_{k+1} \qquad \textit{for} \quad k = 1, 2, \ldots \tag{3.4}$$

This material is reserved for educational use only, not allowed for commercial use.
Forbidden to modify the content, and cite the document when use.

Proof. From Algorithm 4 , we have that for any $k \in \mathbb{N}$,

$$\begin{aligned}
R_{k+1} &= E - \sum_{i=1}^{p} A_i X_{k+1} B_i \sum_{j=1}^{q} C_j X_{k+1}^T D_j \\
&= E - \sum_{i=1}^{p} A_i \left(X_k + \frac{\|R_k\|^2}{\alpha_{k+1}} P_{k+1} \right) B_i - \sum_{j=1}^{q} C_j \left(X_k + \frac{\|R_k\|^2}{\alpha_{k+1}} P_{k+1} \right)^T D_j \\
&= E - \sum_{i=1}^{p} A_i X_k B_i - \sum_{j=1}^{q} C_j X_k^T D_j - \frac{\|R_k\|^2}{\alpha_{k+1}} \left(\sum_{i=1}^{p} A_i P_{k+1} B_i + \sum_{j=1}^{q} C_j P_{k+1}^T D_j \right) \\
&= R_k - \frac{\|R_k\|^2}{\alpha_{k+1}} Q_{k+1}.
\end{aligned}$$

Thus, Eq. (3.4) holds. □

Lemma 3.2. The sequences $\{P_m\}$ and $\{Q_m\}$ generated by Algorithm 4 satisfy

$$\operatorname{tr}\left(P_s^T Q_t\right) = \operatorname{tr}\left(Q_s^T P_t\right) \tag{3.5}$$

for any $s, t \in \mathbb{N}$.

Proof. We apply Lemmas 2.1-2.5. Indeed, we have

$$\begin{aligned}
\operatorname{tr}\left(P_s^T Q_t\right) &= \operatorname{tr}\left[P_s^T \left(\sum_{i=1}^{p} A_i P_t B_i + \sum_{j=1}^{q} C_j P_t^T D_j \right) \right] \\
&= \operatorname{vec}(P_s) \left(\sum_{i=1}^{p} \left(B_i^T \otimes A_i\right) + \sum_{j=1}^{q} \left(D_j^T \otimes C_j\right) P(n,n) \right) \operatorname{vec}(P_t) \\
&= \operatorname{vec}(P_s) \left(\sum_{i=1}^{p} \left(B_i \otimes A_i^T\right) + P(n,n) \sum_{j=1}^{q} \left(D_j \otimes C_j^T\right) \right) \operatorname{vec}(P_t) \\
&= \operatorname{vec}(P_s) \left(\sum_{i=1}^{p} \left(B_i \otimes A_i^T\right) \right) \operatorname{vec}(P_t) + \operatorname{vec}(P_s) \left(P(n,n) \sum_{j=1}^{q} \left(D_j \otimes C_j^T\right) \right) \operatorname{vec}(P_t) \\
&= \operatorname{vec}(P_s) \left(\sum_{i=1}^{p} \left(B_i \otimes A_i^T\right) \right) \operatorname{vec}(P_t) \\
&\quad + \operatorname{vec}(P_s) \left(P(n,n) \sum_{j=1}^{q} \left(D_j \otimes C_j^T\right) \right) \left(P(n,n) \operatorname{vec}(P_t^T) \right) \\
&= \operatorname{vec}(P_s) \left(\sum_{i=1}^{p} \left(B_i \otimes A_i^T\right) \right) \operatorname{vec}(P_t) + \operatorname{vec}(P_s) \sum_{j=1}^{q} \left(C_j^T \otimes D_j\right) \operatorname{vec}(P_t^T) \\
&= \operatorname{tr}\left(P_s^T \left(\sum_{i=1}^{p} A_i^T P_t B_i^T \right) \right) + \operatorname{tr}\left(P_s^T \left(\sum_{i=1}^{p} D_i P_t^T C_j \right) \right) \\
&= \operatorname{tr}\left(\sum_{i=1}^{p} P_s^T A_i^T P_t B_i^T \right) + \operatorname{tr}\left(\sum_{i=1}^{p} P_s^T D_i P_t^T C_j \right) \\
&= \operatorname{tr}\left(\sum_{i=1}^{p} P_t^T A_i P_s B_i \right) + \operatorname{tr}\left(\sum_{i=1}^{p} P_t^T C_j P_s^T D_i \right) \\
&= \operatorname{tr}\left(P_t^T Q_s\right) \\
&= \operatorname{tr}\left(Q_s^T P_t\right).
\end{aligned}$$

This material is reserved for educational use only, not allowed for commercial use.
Forbidden to modify the content, and cite the document when use.

Hence, Eq. (3.5) holds. □

Lemma 3.3. The sequences $\{R_m\}$, $\{P_m\}$ and $\{Q_m\}$ are generated by Algorithm 4 satisfy

$$\text{tr}\left(R_m^T R_{m-1}\right) = 0, \qquad \text{tr}\left(P_{m+1}^T Q_m\right) = 0 \tag{3.6}$$

for any m.

Proof. To prove this conclusion, we use induction principle. In order to compute related terms, we use Lemmas 3.1 and 3.2.
First, for the initial step $m = 1$, we have

$$\begin{aligned}
\text{tr}\left(R_1^T R_0\right) &= \text{tr}\left[\left(R_0 - \frac{\|R_0\|^2}{\alpha_1} Q_1\right)^T R_0\right] \\
&= \text{tr}\left(R_0^T R_0\right) - \frac{\|R_0\|^2}{\alpha_1} \text{tr}\left(Q_1^T R_0\right) \\
&= \text{tr}\left(R_0^T R_0\right) - \frac{\|R_0\|^2}{\alpha_1} \text{tr}\left(Q_1^T P_1\right) \\
&= \|R_0\|^2 - \|R_0\|^2 \\
&= 0,
\end{aligned}$$

and

$$\begin{aligned}
\text{tr}\left(P_2^T Q_1\right) &= \text{tr}\left[\left(R_1 + \frac{\|R_1\|^2}{\|R_0\|^2} P_1\right)^T Q_1\right] \\
&= \text{tr}\left(R_1^T Q_1\right) + \frac{\|R_1\|^2}{\|R_0\|^2} \text{tr}\left(P_1^T Q_1\right) \\
&= \text{tr}\left[R_1^T \left(\frac{\alpha_1}{\|R_0\|^2}\left(R_0 - R_1\right)\right)\right] + \frac{\|R_1\|^2}{\|R_0\|^2}\alpha_1 \\
&= -\frac{\alpha_1}{\|R_0\|^2} \text{tr}\left(R_1^T R_1\right) + \frac{\|R_1\|^2}{\|R_0\|^2}\alpha_1 \\
&= 0.
\end{aligned}$$

Second, suppose that (3.6) holds for $m = k$, we have $(R_k^T R_{k-1}) = 0$ and $(P_{k+1}^T Q_k) = 0$. Then for $m = k+1$,

$$\begin{aligned}
\text{tr}\left(R_{k+1}^T R_k\right) &= \text{tr}\left[\left(R_k - \frac{\|R_k\|^2}{\alpha_{k+1}} Q_{k+1}\right)^T R_k\right] \\
&= \text{tr}\left(R_k^T R_k\right) - \frac{\|R_k\|^2}{\alpha_{k+1}} \text{tr}\left(Q_{k+1}^T R_k\right) \\
&= \|R_k\|^2 - \frac{\|R_k\|^2}{\alpha_{k+1}} \text{tr}\left[Q_{k+1}^T \left(P_{k+1} - \frac{\|R_k\|^2}{\|R_{k-1}\|^2} P_k\right)\right] \\
&= \|R_k\|^2 - \frac{\|R_k\|^2}{\alpha_{k+1}} \text{tr}\left(Q_{k+1}^T P_{k+1}\right) \\
&= \|R_k\|^2 - \|R_k\|^2 \\
&= 0,
\end{aligned}$$

This material is reserved for educational use only, not allowed for commercial use.
Forbidden to modify the content, and cite the document when use.

and we have

$$\begin{aligned}
\text{tr}\left(P_{k+2}^T Q_{k+1}\right) &= \text{tr}\left[\left(R_{k+1} + \frac{\|R_{k+1}\|^2}{\|R_k\|^2} P_{k+1}\right)^T Q_{k+1}\right] \\
&= \text{tr}\left(R_{k+1}^T Q_{k+1}\right) + \frac{\|R_{k+1}\|^2}{\|R_k\|^2} \text{tr}\left(P_{k+1}^T Q_{k+1}\right) \\
&= \text{tr}\left[R_{k+1}\left(\frac{\alpha_{k+1}}{\|R_k\|^2}\left(R_k - R_{k+1}\right)\right)\right] + \frac{\|R_{k+1}\|^2}{\|R_k\|^2}\alpha_{k+1} \text{tr}\left(P_{k+1}^T Q_{k+1}\right) \\
&= \frac{\alpha_{k+1}}{\|R_k\|^2} \text{tr}\left[\left(R_{k+1}^T R_k\right) - \left(R_{k+1}^T R_{k+1}\right)\right] + \frac{\|R_{k+1}\|^2}{\|R_k\|^2}\alpha_{k+1} \\
&= -\frac{\alpha_{k+1}}{\|R_k\|^2}\|R_{k+1}\|^2 + \frac{\|R_{k+1}\|^2}{\|R_k\|^2}\alpha_{k+1} \\
&= 0.
\end{aligned}$$

Hence, the conclusion (3.6) holds. □

Lemma 3.4. Suppose the sequences $\{R_n\}$, $\{P_n\}$ and $\{Q_n\}$ are generated by Algorithm 4.Then

$$\text{tr}\left(R_n^T R_0\right) = 0, \qquad \text{tr}\left(P_{n+1}^T Q_1\right) = 0 \tag{3.7}$$

for any n.

Proof. We prove by induction. For $n = 1$, we have $\text{tr}\left(R_1^T R_0\right) = 0$ and $\text{tr}\left(P_2^T Q_1\right) = 0$ by Lemma 3.3. Now, suppose that Eq. (3.7) is true for all $n = 1, \ldots, m$. From Lemmas 3.1 and 3.2, for $n = m+1$, we have

$$\begin{aligned}
\text{tr}\left(R_{m+1}^T R_0\right) &= \text{tr}\left(R_m^T R_0\right) - \frac{\|R_m\|^2}{\alpha_{m+1}} \text{tr}(Q_{m+1}^T R_0) \\
&= -\frac{\|R_m\|^2}{\alpha_{m+1}} \text{tr}(Q_{m+1}^T R_0) \\
&= -\frac{\|R_m\|^2}{\alpha_{m+1}} \text{tr}(Q_{m+1}^T P_1) \\
&= -\frac{\|R_m\|^2}{\alpha_{m+1}} \text{tr}(P_{m+1}^T Q_1) \\
&= 0,
\end{aligned}$$

and

$$\begin{aligned}
\text{tr}\left(P_{m+2}^T Q_1\right) &= \text{tr}(R_{m+1}^T Q_1) + \frac{\|R_{m+1}\|^2}{\|R_m\|^2} \text{tr}(P_{m+1}^T Q_1) \\
&= \text{tr}(R_{m+1}^T Q_1) \\
&= \text{tr}(R_m^T Q_1) - \frac{\|R_m\|^2}{\alpha_{m+1}} \text{tr}(P_m^T Q_1) \\
&= \text{tr}(P_{m+1}^T Q_1) - \frac{\|R_m\|^2}{\|R_{m-1}\|^2} \text{tr}(P_m^T Q_1) \\
&= 0.
\end{aligned}$$

Therefore, the conclusion (3.7) holds for any n. □

This material is reserved for educational use only, not allowed for commercial use.
Forbidden to modify the content, and cite the document when use.

Theorem 3.5. Suppose the sequences $\{R_s\}$, $\{P_s\}$ and $\{Q_s\}$ are generated by Algorithm 4. Then for any s, t such that $s \neq t$, we have

$$\text{tr}\left(R_{s-1}^T R_{t-1}\right) = 0, \qquad \text{tr}\left(P_s^T Q_t\right) = 0. \tag{3.8}$$

Proof. We prove by using induction. Lemma 3.3 assures that the two equalities in Eq. (3.8) holds for $s = t+1$. Before we process induction step, we consider

$$\begin{aligned}
\text{tr}\left(R_{t+2}^T R_t\right) &= \text{tr}(R_{t+1}^T R_t) - \frac{\|R_{t+1}\|^2}{\alpha_{t+2}\|^2}\,\text{tr}(Q_{t+2}^T R_t) \\
&= -\frac{\|R_{t+1}\|^2}{\alpha_{t+2}}\left[\text{tr}\left(Q_{t+2}^T P_{t+1}\right) - \frac{\|R_t\|^2}{\|R_{t-1}\|^2}\,\text{tr}\left(Q_{t+2}^T P_t\right)\right] \\
&= \frac{\|R_{t+1}\|^2}{\alpha_{t+2}}\frac{\|R_t\|^2}{\|R_{t-1}\|^2}\left[\text{tr}\left(R_{t+1}^T Q_t\right) - \frac{\|R_{t+1}\|^2}{\|R_t\|^2}\,\text{tr}\left(P_{t+1}^T Q_t\right)\right] \\
&= \frac{\|R_{t+1}\|^2}{\alpha_{t+2}}\frac{\|R_t\|^2}{\|R_{t-1}\|^2}\frac{\alpha_t}{\|R_{t-1}\|^2}\left[\text{tr}(R_{t+1}^T R_{t-1}) - \text{tr}(R_{t+1}^T R_t)\right] \\
&= \frac{\|R_{t+1}\|^2}{\alpha_{t+2}}\frac{\|R_t\|^2}{\|R_{t-1}\|^2}\frac{\alpha_t}{\|R_{t-1}\|^2}\,\text{tr}(R_{t+1}^T R_{t-1}),
\end{aligned}$$

$$\begin{aligned}
\text{tr}\left(P_{t+2}^T Q_t\right) &= \text{tr}(R_{t+1}^T Q_t) + \frac{\|R_{t+1}\|^2}{\|R_t\|^2}\,\text{tr}(P_{t+1}^T Q_t) \\
&= \frac{\alpha_t}{\|R_{t-1}\|^2}\left[\text{tr}\left(R_{t+1}^T R_{t-1}\right) - \text{tr}\left(R_{t+1}^T R_t\right)\right] \\
&= \frac{\alpha_t}{\|R_{t-1}\|^2}\left[\text{tr}\left(R_t^T R_{t-1}\right) - \frac{\|R_t\|^2}{\alpha_{t+1}}\,\text{tr}\left(Q_{t+1}^T R_{t-1}\right)\right] \\
&= -\frac{\alpha_t}{\|R_{t-1}\|^2}\frac{\|R_t\|^2}{\alpha_{t+1}}\left[\text{tr}\left(Q_{t+1}^T P_t\right) - \frac{\|R_{t-1}\|^2}{\|R_{t-2}\|^2}\,\text{tr}\left(Q_{t+1}^T P_{t-1}\right)\right] \\
&= \frac{\alpha_t}{\|R_{t-1}\|^2}\frac{\|R_t\|^2}{\alpha_{t+1}}\frac{\|R_{t-1}\|^2}{\|R_{t-2}\|^2}\,\text{tr}\left(Q_{t+1}^T P_{t-1}\right) \\
&= \frac{\alpha_t}{\|R_{t-1}\|^2}\frac{\|R_t\|^2}{\alpha_{t+1}}\frac{\|R_{t-1}\|^2}{\|R_{t-2}\|^2}\,\text{tr}\left(P_{t+1}^T Q_{t-1}\right),
\end{aligned}$$

$$\begin{aligned}
\text{tr}\left(R_{t+1}^T R_{t-1}\right) &= \text{tr}(R_t^T R_{t-1}) - \frac{\|R_t\|^2}{\alpha_{t+1}\|^2}\,\text{tr}(Q_{t+1}^T R_{t-1}) \\
&= -\frac{\|R_t\|^2}{\alpha_{t+1}}\left[\text{tr}\left(Q_{t+1}^T P_t\right) - \frac{\|R_{t-1}\|^2}{\|R_{t-2}\|^2}\,\text{tr}\left(Q_{t+1}^T P_{t-1}\right)\right] \\
&= \frac{\|R_t\|^2}{\alpha_{t+1}}\frac{\|R_{t-1}\|^2}{\|R_{t-2}\|^2}\left[\text{tr}\left(R_t^T Q_{t-1}\right) - \frac{\|R_t\|^2}{\|R_{t-1}\|^2}\,\text{tr}\left(P_t^T Q_{t-1}\right)\right] \\
&= \frac{\|R_t\|^2}{\alpha_{t+1}}\frac{\|R_{t-1}\|^2}{\|R_{t-2}\|^2}\frac{\alpha_{t-1}}{\|R_{t-2}\|^2}\left[\text{tr}(R_t^T R_{t-2}) - \text{tr}(R_t^T R_{t-1})\right] \\
&= \frac{\|R_t\|^2}{\alpha_{t+1}}\frac{\|R_{t-1}\|^2}{\|R_{t-2}\|^2}\frac{\alpha_{t-1}}{\|R_{t-2}\|^2}\,\text{tr}(R_t^T R_{t-2}),
\end{aligned}$$

This material is reserved for educational use only, not allowed for commercial use.
Forbidden to modify the content, and cite the document when use.

and

$$\begin{aligned}
\mathrm{tr}\left(P_{t+1}^T Q_{t-1}\right) &= \mathrm{tr}(R_t^T Q_{t-1}) + \frac{\|R_t\|^2}{\|R_{t-1}\|^2}\,\mathrm{tr}(P_t^T Q_{t-1}) \\
&= \frac{\alpha_{t-1}}{\|R_{t-2}\|^2}\left[\mathrm{tr}\left(R_t^T R_{t-2}\right) - \mathrm{tr}\left(R_t^T R_{t-1}\right)\right] \\
&= \frac{\alpha_{t-1}}{\|R_{t-2}\|^2}\left[\mathrm{tr}\left(R_{t-1}^T R_{t-2}\right) - \frac{\|R_{t-1}\|^2}{\alpha_t}\,\mathrm{tr}\left(Q_t^T R_{t-2}\right)\right] \\
&= -\frac{\alpha_{t-1}}{\|R_{t-2}\|^2}\frac{\|R_{t-1}\|^2}{\alpha_t}\left[\mathrm{tr}\left(Q_t^T P_{t-1}\right) - \frac{\|R_{t-2}\|^2}{\|R_{t-3}\|^2}\,\mathrm{tr}\left(Q_t^T P_{t-2}\right)\right] \\
&= \frac{\alpha_{t-1}}{\|R_{t-2}\|^2}\frac{\|R_{t-1}\|^2}{\alpha_t}\frac{\|R_{t-2}\|^2}{\|R_{t-3}\|^2}\,\mathrm{tr}\left(Q_t^T P_{t-2}\right) \\
&= \frac{\alpha_{t-1}}{\|R_{t-2}\|^2}\frac{\|R_{t-1}\|^2}{\alpha_t}\frac{\|R_{t-2}\|^2}{\|R_{t-3}\|^2}\,\mathrm{tr}\left(P_t^T Q_{t-2}\right).
\end{aligned}$$

Similarly, we can write $\mathrm{tr}\left(R_{t+2}^T R_t\right)$ and $\mathrm{tr}\left(P_{t+2}^T Q_t\right)$ in terms of $\mathrm{tr}\left(R_{t+1}^T R_{t-1}\right)$ and $\mathrm{tr}\left(P_{t+1}^T Q_{t-1}\right)$, respectively. Repeating this process until the terms $\mathrm{tr}\left(P_3^T Q_1\right)$ and $\mathrm{tr}\left(R_2^T R_0\right)$ show up. By Lemma 3.4, we get $\mathrm{tr}\left(R_{t+1}^T R_{t-1}\right) = 0$ and $\mathrm{tr}\left(P_{t+1}^T Q_{t-1}\right) = 0$.

Next, we compute

$$\begin{aligned}
\mathrm{tr}\left(R_{t+3}^T R_t\right) &= \mathrm{tr}(R_{t+2}^T R_t) - \frac{\|R_{t+2}\|^2}{\alpha_{t+3}}\,\mathrm{tr}(Q_{t+3}^T R_t) \\
&= -\frac{\|R_{t+2}\|^2}{\alpha_{t+3}}\left[\mathrm{tr}\left(Q_{t+3}^T P_{t+1}\right) - \frac{\|R_t\|^2}{\|R_{t-1}\|^2}\left(Q_{t+3}^T P_t\right)\right] \\
&= \frac{\|R_{t+2}\|^2}{\alpha_{t+3}}\frac{\|R_t\|^2}{\|R_{t-1}\|^2}\left(Q_{t+3}^T P_t\right) \\
&= \frac{\|R_{t+2}\|^2}{\alpha_{t+3}}\frac{\|R_t\|^2}{\|R_{t-1}\|^2}\left(P_{t+3}^T Q_t\right) \\
&= -\frac{\|R_{t+2}\|^2}{\alpha_{t+3}}\frac{\|R_t\|^2}{\|R_{t-1}\|^2}\left[\mathrm{tr}\left(R_{t+2}^T Q_t\right) + \frac{\|R_{t+2}\|^2}{\|R_{t+1}\|^2}\,\mathrm{tr}\left(P_{t+2}^T Q_t\right)\right] \\
&= -\frac{\|R_{t+2}\|^2}{\alpha_{t+3}}\frac{\|R_t\|^2}{\|R_{t-1}\|^2}\frac{\alpha_t}{\|R_{t-1}\|^2}\left[\mathrm{tr}\left(R_{t+2}^T R_{t-1}\right) - \mathrm{tr}\left(R_{t+2}^T R_t\right)\right] \\
&= -\frac{\|R_{t+2}\|^2}{\alpha_{t+3}}\frac{\|R_t\|^2}{\|R_{t-1}\|^2}\frac{\alpha_t}{\|R_{t-1}\|^2}\,\mathrm{tr}\left(R_{t+2}^T R_{t-1}\right),
\end{aligned}$$

and

$$\begin{aligned}
\mathrm{tr}\left(P_{t+3}^T Q_t\right) &= \mathrm{tr}(R_{t+2}^T Q_t) + \frac{\|R_{t+2}\|^2}{\|R_{t+1}\|^2}\,\mathrm{tr}(P_{t+2}^T Q_t) \\
&= \frac{\alpha_t}{\|R_{t-1}\|^2}\left[\mathrm{tr}\left(R_{t+2}^T R_{t-1}\right) - \mathrm{tr}\left(R_{t+2}^T R_t\right)\right] \\
&= \frac{\alpha_t}{\|R_{t-1}\|^2}\left[\mathrm{tr}\left(R_{t+1}^T R_{t-1}\right) - \frac{\|R_{t+1}\|^2}{\alpha_{t+2}}\,\mathrm{tr}\left(Q_{t+2}^T R_{t-1}\right)\right] \\
&= -\frac{\alpha_t}{\|R_{t-1}\|^2}\frac{\|R_{t+1}\|^2}{\alpha_t}\left[\mathrm{tr}\left(Q_{t+2}^T P_t\right) - \frac{\|R_{t-1}\|^2}{\|R_{t-2}\|^2}\,\mathrm{tr}\left(Q_{t+2}^T P_{t-1}\right)\right] \\
&= \frac{\alpha_t}{\|R_{t-1}\|^2}\frac{\|R_{t+1}\|^2}{\alpha_{t-1}}\frac{\|R_{t-1}\|^2}{\|R_{t-2}\|^2}\,\mathrm{tr}\left(Q_{t+2}^T P_{t-1}\right) \\
&\quad + \frac{\alpha_t}{\|R_{t-1}\|^2}\frac{\|R_{t+1}\|^2}{\alpha_{t-1}}\frac{\|R_{t-1}\|^2}{\|R_{t-2}\|^2}\,\mathrm{tr}\left(P_{t+2}^T Q_{t-1}\right).
\end{aligned}$$

This material is reserved for educational use only. Not allowed for commercial use.
Forbidden to modify the content, and cite the document when use.

Also, we have

$$\begin{aligned}
\mathrm{tr}\left(R_{t+2}^T R_{t-1}\right) &= \mathrm{tr}(R_{t+1}^T R_{t-1}) - \frac{\|R_{t+1}\|^2}{\alpha_{t+2}}\,\mathrm{tr}(Q_{t+2}^T R_{t-1}) \\
&= -\frac{\|R_{t+1}\|^2}{\alpha_{t+2}}\left[\mathrm{tr}\left(Q_{t+2}^T P_t\right) - \frac{\|R_{t-1}\|^2}{\|R_{t-2}\|^2}\left(Q_{t+2}^T P_t\right)\right] \\
&= \frac{\|R_{t+1}\|^2}{\alpha_{t+2}}\frac{\|R_{t-1}\|^2}{\|R_{t-2}\|^2}\left(Q_{t+2}^T P_{t-1}\right) \\
&= \frac{\|R_{t+1}\|^2}{\alpha_{t+2}}\frac{\|R_{t-1}\|^2}{\|R_{t-2}\|^2}\left(P_{t+2}^T Q_{t-1}\right) \\
&= -\frac{\|R_{t+1}\|^2}{\alpha_{t+2}}\frac{\|R_{t-1}\|^2}{\|R_{t-2}\|^2}\left[\mathrm{tr}\left(R_{t+1}^T Q_t\right) + \frac{\|R_{t+1}\|^2}{\|R_t\|^2}\,\mathrm{tr}\left(P_{t+1}^T Q_{t-1}\right)\right] \\
&= -\frac{\|R_{t+1}\|^2}{\alpha_{t+2}}\frac{\|R_{t-1}\|^2}{\|R_{t-2}\|^2}\frac{\alpha_{t-1}}{\|R_{t-2}\|^2}\left[\mathrm{tr}\left(R_{t+1}^T R_{t-2}\right) - \mathrm{tr}\left(R_{t+1}^T R_{t-1}\right)\right] \\
&= -\frac{\|R_{t+1}\|^2}{\alpha_{t+2}}\frac{\|R_{t-1}\|^2}{\|R_{t-2}\|^2}\frac{\alpha_{t-1}}{\|R_{t-2}\|^2}\,\mathrm{tr}\left(R_{t+1}^T R_{t-2}\right),
\end{aligned}$$

$$\begin{aligned}
\mathrm{tr}\left(P_{t+2}^T Q_{t-1}\right) &= \mathrm{tr}(R_{t+1}^T Q_{t-1}) + \frac{\|R_{t+1}\|^2}{\|R_t\|^2}\,\mathrm{tr}(P_{t+1}^T Q_{t-1}) \\
&= \frac{\alpha_{t-1}}{\|R_{t-2}\|^2}\left[\mathrm{tr}\left(R_{t+1}^T R_{t-2}\right) - \mathrm{tr}\left(R_{t+1}^T R_{t-1}\right)\right] \\
&= \frac{\alpha_{t-1}}{\|R_{t-2}\|^2}\left[\mathrm{tr}\left(R_t^T R_{t-2}\right) - \frac{\|R_t\|^2}{\alpha_{t+1}}\,\mathrm{tr}\left(Q_{t+1}^T R_{t-2}\right)\right] \\
&= -\frac{\alpha_{t-1}}{\|R_{t-2}\|^2}\frac{\|R_t\|^2}{\alpha_{t+1}}\left[\mathrm{tr}\left(Q_{t+1}^T P_{t-1}\right) - \frac{\|R_{t-2}\|^2}{\|R_{t-3}\|^2}\,\mathrm{tr}\left(Q_{t+1}^T P_{t-2}\right)\right] \\
&= \frac{\alpha_{t-1}}{\|R_{t-2}\|^2}\frac{\|R_t\|^2}{\alpha_{t+1}}\frac{\|R_{t-2}\|^2}{\|R_{t-3}\|^2}\,\mathrm{tr}\left(Q_{t+1}^T P_{t-2}\right) \\
&= \frac{\alpha_{t-1}}{\|R_{t-2}\|^2}\frac{\|R_t\|^2}{\alpha_{t+1}}\frac{\|R_{t-2}\|^2}{\|R_{t-3}\|^2}\,\mathrm{tr}\left(P_{t+1}^T Q_{t-2}\right).
\end{aligned}$$

Hence, we can write $\mathrm{tr}\left(R_{t+3}^T R_t\right)$ and $\mathrm{tr}\left(P_{t+3}^T Q_t\right)$ in terms of $\mathrm{tr}\left(R_{t+2}^T R_{t-1}\right)$ and $\mathrm{tr}\left(P_{t+2}^T Q_{t-1}\right)$. We use this idea to prove the conclusion (3.8)

Suppose that $\mathrm{tr}\left(R_{s-1}^T R_{t-1}^T\right) = 0$ and $\mathrm{tr}\left(P_s^T Q_t^T\right) = 0$ for $s = t+1, \ldots, m$. Then for $s = m+1$, we get

$$\begin{aligned}
\mathrm{tr}\left(R_m^T R_{t-1}\right) &= \mathrm{tr}\left[\left(R_{m-1} - \frac{\|R_{m-1}\|^2}{\alpha_{m-1}} Q_{m-1}\right)^T R_{t-1}\right] \\
&= \mathrm{tr}\left(R_{m-1}^T R_{t-1}\right) - \frac{\|R_{m-1}\|^2}{\alpha_{m-1}}\,\mathrm{tr}\left(Q_{m-1}^T R_{t-1}\right) \\
&= -\frac{\|R_{m-1}\|^2}{\alpha_{m-1}}\,\mathrm{tr}\left(Q_{m-1}^T R_{t-1}\right) \\
&= -\frac{\|R_{m-1}\|^2}{\alpha_{m-1}}\,\mathrm{tr}\left[Q_{m-1}^T\left(P_t - \frac{\|R_{t-1}\|^2}{\alpha_{t-2}} P_{t-1}\right)\right] \\
&= -\frac{\|R_{m-1}\|^2}{\alpha_{m-1}}\left[\mathrm{tr}\left(Q_{m-1}^T P_t\right) - \frac{\|R_{t-1}\|^2}{\alpha_{t-2}}\left(Q_{m-1}^T P_{t-1}\right)\right] \\
&= 0,
\end{aligned}$$

This material is reserved for educational use only, not allowed for commercial use.
Forbidden to modify the content, and cite the document when use.

and thus

$$\begin{aligned}
\text{tr}\left(P_{m+1}^T Q_t\right) &= \text{tr}\left[\left(R_m + \frac{\|R_m\|^2}{m-1} P_m\right)^T Q_t\right] \\
&= \text{tr}\left(R_m^T Q_t\right) + \frac{\|R_m\|^2}{\alpha_m - 1} \text{tr}\left(P_m^T Q_t\right) \\
&= \text{tr}\left(R_m^T Q_t\right) \\
&= \frac{\alpha_t}{\|R_{t-1}\|^2} \left[\text{tr}\left(R_m^T R_{t-1}\right) - \text{tr}\left(R_m^T R_t\right)\right] \\
&= 0.
\end{aligned}$$

Hence, by induction principle, the conclusion (3.8) holds for any s, t such that $s \neq t$. □

Theorem 3.6. Suppose that the sequence $\{X_k\}$ is generated by Algorithm 4. Then for any initial matrix X_0, the exact solution X can be obtained in at most n^2 iteration steps.

Proof. Suppose that $R_m \neq 0$ for $m = 0, 1, \ldots, n^2 - 1$. Then we compute X_{n^2}. Assume that $R_{n^2} \neq 0$. By Theorem 3.5, the set $\{R_0, R_1, ..., R_{n^2}\}$ is orthogonal in $\mathbb{R}^{n \times n}$. So, $\{R_0, R_1, ..., R_{n^2}\}$ is linearly independent. Since the dimension of $\mathbb{R}^{n \times n}$ is n^2, any linearly independent subset of $\mathbb{R}^{n \times n}$ must have at most n^2 elements. So this is false because $\{R_0, R_1, ..., R_{n^2}\}$ has $n^2 + 1$ elements. Thus $R_{n^2} = 0$. Hence X_{n^2} is a solution of the equation. □

This material is reserved for educational use only, not allowed for commercial use.
Forbidden to modify the content, and cite the document when use.

Chapter 4

Numerical Experiment

In this chapter, we provide numerical examples to show the capability and performance of Algorithm 4. This algorithm has been carried out by MATLAB R2013a, Intel(R) Pentium(R) CPU N3540 @ 2.16GHz 2.16 GHz, RAM 4.00 GB.

Example 4.1. Consider the matrix equation

$$A_1XB_1 + A_2XB_2 + CX^TD = E$$

where

$$A_1 = \begin{bmatrix} 12 & 7 & 9 & 11 \\ 7 & 3 & 16 & 13 \\ 9 & 16 & 17 & 14 \\ 11 & 13 & 14 & 2 \end{bmatrix}, A_2 = \begin{bmatrix} 7 & 4 & 0 & 9 \\ 4 & 7 & 11 & 5 \\ 0 & 11 & 8 & 12 \\ 9 & 5 & 12 & 14 \end{bmatrix}, B_2 = \begin{bmatrix} 5 & 2 & 0 & 9 \\ 2 & 8 & 2 & 11 \\ 0 & 2 & 9 & 0 \\ 9 & 11 & 0 & 5 \end{bmatrix},$$

$$E = \begin{bmatrix} 2522 & 2781 & 711 & 2880 \\ 2143 & 3191 & 742 & 3368 \\ 3157 & 2454 & 565 & 3887 \\ 3721 & 5013 & 1172 & 5389 \end{bmatrix}, \text{ and } B_1 = C = D = I_4.$$

We can verify that the matrix equation has a unique solution

$$X = \begin{bmatrix} 12 & 2 & 7 & 3 \\ 3 & 0 & 2 & 9 \\ 0 & 11 & 0 & 0 \\ 5 & 4 & 0 & 12 \end{bmatrix}.$$

By using Algorithm 4, we choose initial matrix $X_1 = 0$ and use $\varepsilon = 10^{-8}$ for stopping this process. In fact, we arrive at

$$X^* = \begin{bmatrix} 11.9999 & 1.9999 & 6.9999 & 2.9999 \\ 2.9999 & 0.0000 & 2.0000 & 8.9999 \\ 0.0000 & 10.9999 & 0.0000 & 0.0000 \\ 4.9999 & 3.9999 & 0.0000 & 11.9999 \end{bmatrix}$$

in 21 iterations. Figure 4.1 shows the errors of solutions converge to zero.

This material is reserved for educational use only, not allowed for commercial use.
Forbidden to modify the content, and cite the document when use.

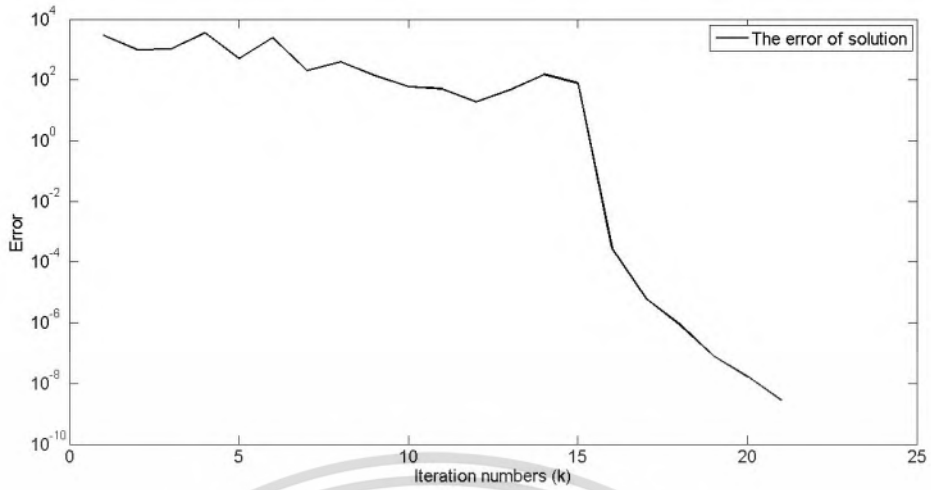

Figure 4.1: The errors of solution for Example 4.1

Example 4.2. Consider the matrix equation

$$A_1XB_1 + A_2XB_2 + A_3XB_3 + CX^TD = E$$

with

$$A_1 = \begin{bmatrix} 0 & 6 & 3 \\ 6 & 2 & 8 \\ 3 & 8 & 9 \end{bmatrix}, A_2 = \begin{bmatrix} 6 & 5 & 7 \\ 5 & 10 & 7 \\ 7 & 7 & 1 \end{bmatrix}, A_3 = \begin{bmatrix} 8 & 6 & 5 \\ 6 & 3 & 5 \\ 5 & 5 & 8 \end{bmatrix}, B_1 = \begin{bmatrix} 4 & 10 & 7 \\ 10 & 6 & 6 \\ 7 & 6 & 6 \end{bmatrix},$$

$$B_2 = \begin{bmatrix} 7 & 2 & 3 \\ 2 & 9 & 2 \\ 3 & 2 & 4 \end{bmatrix}, B_3 = \begin{bmatrix} 8 & 6 & 4 \\ 6 & 10 & 1 \\ 4 & 1 & 8 \end{bmatrix}, C = D = \begin{bmatrix} 3 & 9 & 4 \\ 9 & 10 & 4 \\ 4 & 4 & 10 \end{bmatrix}, \text{and } E = \begin{bmatrix} 38 & 21 & 61 \\ 23 & 32 & 25 \\ 15 & 38 & 63 \end{bmatrix}.$$

We applied Algorithm 4 to compute X_k with stopping rule $\varepsilon = 10^{-8}$. We provide three different initial matrices, such as the identity matrix, the zero matrix, and a random matrix

$$X_0 = \begin{bmatrix} 38 & 21 & 61 \\ 23 & 32 & 25 \\ 15 & 38 & 63 \end{bmatrix}.$$

Table 4.1 shows the errors of solution with three different initial matrices within 9 iteration steps. We see that any initial matrix construct approximated solutions which converge to an exact solution.

Table 4.1: The errors of solution with different initial matrices

k	5	6	7	8	9
Identity matrix	2.28e+1	1.84e+1	3.61e-1	8.05e-3	1.84e-11
Zero matrix	3.42e+0	2.80e+0	7.43e-1	5.25e-4	3.73e-12
Random matrix	4.40e+1	1.59e+1	4.29e-1	1.25e-4	4.44e-12

This material is reserved for educational use only, not allowed for commercial use.
Forbidden to modify the content, and cite the document when use.

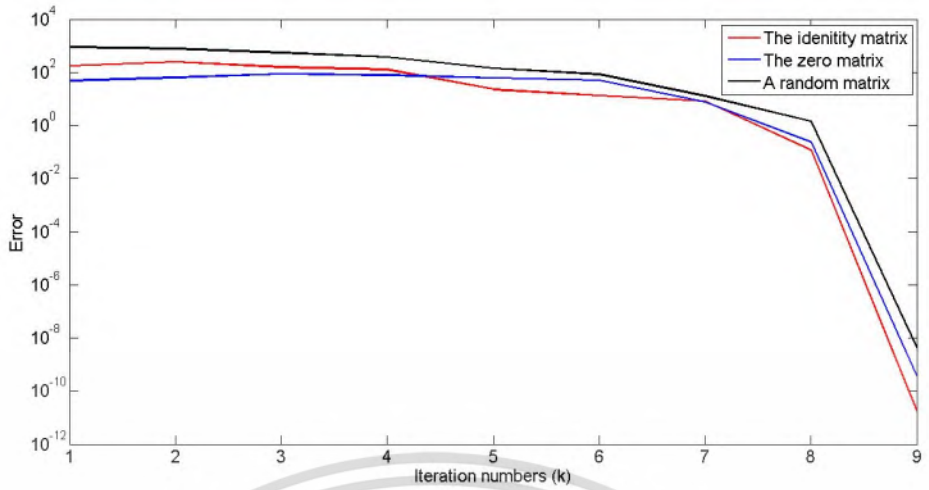

Figure 4.2: The errors of solution with different initial matrices for Example 4.2

Next, we provide an example to show that the computational time of Algorithm 4 is less than the time consumed by the direct method when the matrix has enormous size.

Example 4.3. Consider the equation (1.8) where the matrix coefficients A_i, B_i, C_j, D_j, E such that $i = 1, \ldots, 3$ and $j = 1, \ldots, 4$, are 40×40 real tridiagonal matrices as follows:

$$A_1 = \text{trid}(1, -3, 1), \quad A_2 = \text{trid}(-1, -2, -1), \quad A_3 = \text{trid}(-1, 3, -1),$$
$$B_1 = \text{trid}(2, 1, 2), \quad B_2 = \text{trid}(1, 3, 1), \quad B_3 = \text{trid}(0, -3, 0),$$
$$C_1 = D_1 = \text{trid}(2, 0, 2), \quad C_2 = D_2 = \text{trid}(1, -1, 1), \quad C_3 = D_3 = \text{trid}(-1, 0, -1),$$
$$C_4 = D_4 = \text{trid}(0, 2, 0), \quad \text{and} \quad E = I_4 .$$

We choose initial matrix $X_0 = 0$ and a small tolerance $\varepsilon = 10^{-12}$.

The direct method takes around 122 seconds to obtain the exact solution. However, solving this problem via Algorithm 4 consumes only 0.568547 seconds within 103 iterations and the error norm $\|R_{103}\| = 8.8733 \times 10^{-13}$.

Table 4.2: The errors of solution in case of enormous size

k	20	40	60	80	100	103
Algorithm 4	5.62+0	3.45e-4	1.22e-6	2.44e-9	4.17e-12	8.87e-13

This material is reserved for educational use only, not allowed for commercial use.
Forbidden to modify the content, and cite the document when use.

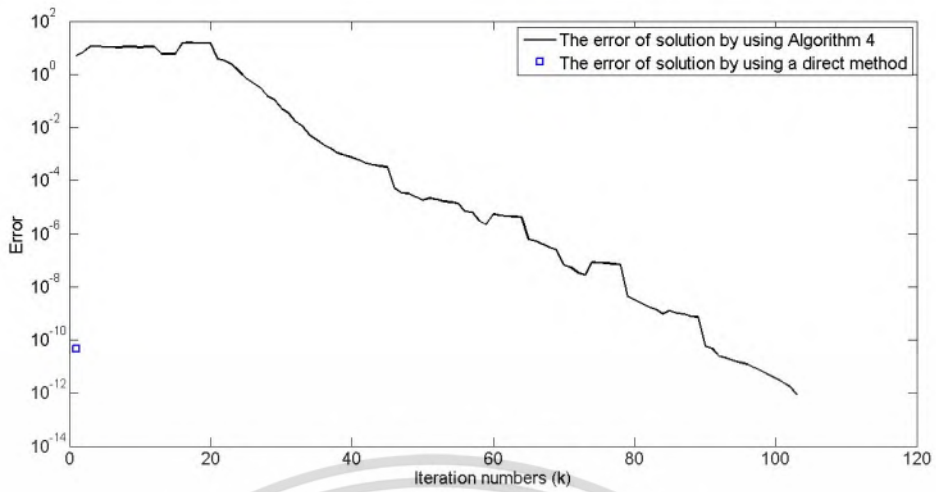

Figure 4.3*:* The errors of solution in case of enormous size for Example 4.3

This material is reserved for educational use only, not allowed for commercial use.
Forbidden to modify the content, and cite the document when use.

Chapter 5

Conclusion and Suggestion

5.1 Conclusion

We propose an iterative procedure (Algorithm 4) to construct a sequence of approximate solutions for the generalized Sylvester-transpose matrix equation (1.8). The algorithm is applicable whenever the matrix M, defined by Eq. (3.2), is invertible and symmetric, not necessarily positive definite. We get the exact solution within n^2 iterations. Moreover, Algorithm 4 can be applied to any special case of Eq. (1.8), e.g. Eqs. (1.1)-(1.7).

5.2 Suggestion

This algorithm provides an exact solution within n^2 iterations but the error of solution has been fluctuated of computation. The further research may include other condition for decrease the error of solution steadily.

This material is reserved for educational use only, not allowed for commercial use.
Forbidden to modify the content, and cite the document when use.

References

[1] M. Dehghan, M Hajarian, *An iterative algorithm for solving a pair of matrix equations* $AYB = E, CYD = F$ *over generalized centro-symmetric matrices*, Comput. Math. Appl., **12** (2008), 3246–3260.

[2] F. Ding, T. Chen, *Iterative least squares solutions of coupled Sylvester matrix equations*, Systems Control Lett., **5** (2005), 95–107.

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

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

[5] R.A. Horn, C.R. Johnson, *Topics in matrix Analysis*, Cambridge University Press, Cambridge, UK, 1985.

[6] Y.J. Xie ,C.F. Ma , *The accelerated gradient based iterative algorithm for solving a class of generalized Sylvester-transpose matrix equation*, Appl. Math. Comp., **273** (2016), 1257–1269.

[7] Y.F. Ke, C.F. Ma, *A preconditioned nested splitting conjugate gradient iterative method for the large sparse generalized Sylvester equation*, Appl. Math. Comput., **10** (2014), 1409–1420.

[8] Y. Lei, A. Liao, *A minimal residual algorithm for the inconsistent matrix equation* $AXB = C$ *over symmetric matrices*, Appl. Math. Comput, **188** (2007), 499–513.

[9] S. Li, T. Huang, *A shift-splitting Jacobi-gradient algorithm for Lyapunov matrix equation arising form control theory*, J.Comput. Anal. Appl., **13** (2011), 1246–1257.

[10] K. Liang, J. Liu, *Iterative algorithms for the minimum-norm solution and the least-squares solution of the linear matrix equations*,Appl. Math. Comput., **218** (2011), 3166–3175.

[11] A. Liao, Z.Z. Bai, Y. Lei, *best approximate solution of matrix equation* $AXB+CXD = E$, SIAM Journal on Matrix Analysis and Applications,**27** (2005), 675–688.

[12] Y. Lin, *Minimal residual methods augmented with eigenvectors for solving Sylvester equations and generalized Sylvester equatios*, Appl. Math. Comput., **181** (2006), 487–499.

[13] Q. Niu, X. Wang, L. Lu, *A relaxed gradient based algorithm for solving Sylvester equation*, Asian J. Control, **13** (2011), 461–464.

This material is reserved for educational use only, not allowed for commercial use.
Forbidden to modify the content, and cite the document when use.

[14] P.J. Olver, C. Shakiban, *Applied Linear Algebra*, Pearson Education, 2006.

[15] M. Wang, X. Cheng, M. Wei, *Iterative algorithms for solving the matrix equation* $AXB + CX^TD = E$, Appl. Math. Comput., **187** (2007), 622–629.

[16] Y.J. Xie, *C.F. Ma, Gradient based and least square based iterative algorithms for matrix equation* $AXB + CX^TB = F$, Appl. Math. Comp, **217** (2010), 2191–2199.

[17] Y.J. Xie, C.F. Ma, *The accelerated gradient based iterative algorithm for solving a class of generalized Sylvester-transpose matrix equation*,Appl. Math. Comp., **218** (2012), 5620–5628.

[18] B. Zhou, G.R. Duan, Z.Y. Li, *Gradient based iterative algorithm for solving coupled matrix equations*, Systems Control Lett, **58** (2009), 327–333.

This material is reserved for educational use only, not allowed for commercial use.
Forbidden to modify the content, and cite the document when use.

Appendix

This material is reserved for educational use only, not allowed for commercial use.
Forbidden to modify the content, and cite the document when use.

Appendix A

The research paper

This material is reserved for educational use only, not allowed for commercial use.
Forbidden to modify the content, and cite the document when use.

APAM 2021
Annual Pure and Applied
Mathematics Conference 2021

Conjugate gradient algorithm for a generalized Sylvester-transpose matrix equation

Sarawanee Choomklang[1] and Pattrawut Chansangiam*[2]

[1,2]Department of Mathematics, Faculty of Science, King Mongkut's Institute of Technology Ladkrabang, Bangkok 10520, Thailand

[1]sarawanich@gmail.com and [2]pattrawut.ch@kmitl.ac.th

Abstract

In the present paper, we propose a semi-direct method to find the exact solution of a generalized Sylvester-transpose matrix equation, namely a conjugate gradient method. The proposed method is applicable for the matrix equation whose matrix coefficients constitute an associated symmetric matrix. The method aims to construct a finite sequence of approximated solutions for the matrix equation, based on orthogonality of associated residual matrices. The final step of iteration comes out with the exact solution of the equation.

Mathematics Subject Classification: 15A12, 15A60, 15A69, 65F45

Keywords: generalized Sylvester-transpose matrix equation, Kronecker product, matrix norm, orthogonality

1 Introduction

Linear matrix equations are famous problems often shown up in mathematics and engineering, especially control theory, system theory, see e.g. [10], [17]. There are many well-

*Corresponding author

This material is reserved for educational use only, not allowed for commercial use.
Forbidden to modify the content, and cite the document when use.

known types of matrix equations such as the following:

$$AX - XA^T = C, \quad \text{Lyapunov matrix equation} \tag{1.1}$$
$$E^TAX + AXE^T = -C, \quad \text{a generalized Lyapunov matrix equation} \tag{1.2}$$
$$X + AXB = C, \quad \text{Stein matrix equation} \tag{1.3}$$
$$AX + XB = C, \quad \text{Sylvester matrix equation} \tag{1.4}$$
$$AXB + CXD = E, \quad \text{a generalized Sylvester matrix equation} \tag{1.5}$$
$$AX + X^TB = C, \quad \text{Sylvester-transpose matrix equation} \tag{1.6}$$
$$AXB + CX^TD = E, \quad \text{a generalized Sylvester-transpose matrix equation.} \tag{1.7}$$

Here, A, B, C, D, and E are given matrices and X is an unknown matrix to be found. An ordinary method for solving a linear matrix equation is to reduce it to an equivalent linear system $Ux = v$ by taking the vector operator. If U is a nonsingular matrix, then the matrix equation has a unique solution. Since U is forming by the Kronecker multiplication, the size of U may be enormous. For instance

$$(B^T \otimes A + D^T \otimes C)\,\mathrm{vec}\,X = \mathrm{vec}\,E$$

is equivalent to (1.5) by taking the vector operator. Here, the symbol $\otimes$ denotes the Kronecker product and the operation $\mathrm{vec}(\cdot)$ is the vector operator. Whenever the matrix has enormous size, the computation will consume a long time and many memories.

In the recent decade, many researchers developed several iterative methods for solving the matrix equation (1.1)-(1.7). The obviously advantage of iteration is not necessary use memories as massive as the regular method in a big dimension case. We can classify such iterative method into two groups. The first group are stationary iterative methods; see e.g. [2, 5, 8, 12, 15, 16]. These methods aim to construct a sequence of approximated solution converging to the exact solution. The second group, known as Krylov subspace methods, are semi-direct methods aiming to construct an orthogonal basis for the associated vector space. The basis consists of (residual) vectors pointing in direction so that the approximated solutions fastest approach to the exact solution.

In fact, the equations (1.1)-(1.7) are special cases of the following generalized Sylvester-

This material is reserved for educational use only, not allowed for commercial use.
Forbidden to modify the content, and cite the document when use.

transpose matrix equation:

$$\sum_{i=1}^{p} A_i X B_i + \sum_{j=1}^{q} C_j X^T D_j = E \tag{1.8}$$

Recently, the matrix equation has been discussed extensively. In 2014, Hajarian presented a bi-conjugate gradient and bi-conjugate residual methods for the generalized Sylvester-transpose matrix equation [3]. The Sylvester matrix equation and the generalized Sylvester matrix equation has been solved by Yiqin Lin in 2006 [11]. Yi-Fen Ke and Chang-Feng Ma propose a nested splitting conjugate gradient method and a preconditioned nested splitting conjugate gradient method for a large sparse coefficient of the equation (1.5) [6]. To Solve a pair of matrix equations over generalized centro-symmetric matrix, Hajarian and Dehghan propose iterative algorithm in 2008 [1]. Liang and Liu propose two iterative algorithms for a minimum-norm and a least square solution of a pair of generalized Sylvester-conjugate matrix equation [9]. Wang, Cheng, and Wei propose two iterative algorithms for the equation (1.7) in 2007 by applying conjugate gradient method [14]. In [7], Lei and Liao propose a minimal residual algorithm for the inconsistent matrix equation $AXB = C$ over symmetric matrices in 2007. The CGS algorithm and the extended CGS algorithm for the general couple matrix equation has been proposed by Hajarian [4].

In this paper, we intend to present a new choice of iterative algorithm to solve the matrix equation (1.8) when the equation is consistent and has a unique solution. Our method is based on conjugate gradient method idea. The propose method is applicable as long as the coefficient matrix must be symmetric and invertible. The method aims to construct a finite sequence of approximated solutions for the matrix equation, based on orthogonality of associated residual matrices. The final step of iteration comes out with the exact solution of the equation.

2 Preliminaries

In this section, we recall fundamental tools for solving linear matrix equations and analysing matrix iterative algorithms. For each $m, n \in \mathbb{N}$, we denote the set of $m \times n$ real matrices by $\mathbb{R}^{m\times n}$. When $n = 1$, the set $\mathbb{R}^m = \mathbb{R}^{m\times 1}$ is the set of m-dimensional (column) vectors.

Recall that the Kronecker product is a matrix product defined for two matrices of arbitrary dimensions. Indeed, the Kronecker product of $A = [a_{ij}] \in \mathbb{R}^{m\times n}$ and $B \in \mathbb{R}^{p\times q}$ is defined to

This material is reserved for educational use only, not allowed for commercial use.
Forbidden to modify the content, and cite the document when use.

be

$$A \otimes B \;=\; \begin{bmatrix} a_{11}B & \dots & a_{1n}B \\ \vdots & \ddots & \vdots \\ a_{m1}B & \dots & a_{mn}B \end{bmatrix} \in \mathbb{R}^{mp \times nq}.$$

Lemma 2.1. *The following properties hold for any compatible matrices A, B, C:*

1. $(A \otimes B)^T \;=\; A^T \otimes B^T$,
2. $(A + B) \otimes C \;=\; A \otimes C + B \otimes C$,
3. $A \otimes (B + C) \;=\; A \otimes B + A \otimes C$.

Recall that the commutation matrix $P(m, n)$ is a permutation matrix defined by

$$P(m,n) \;=\; \sum_{i=1}^{m} \sum_{j=1}^{n} E_{ij} \otimes E_{ij}^T \;\in\; \mathbb{R}^{mn \times mn} \tag{2.1}$$

where each $E_{ij} \in \mathbb{R}^{m \times n}$ has entry 1 in position (i, j) and all other entries are 0. Moreover, $P(m, n) \;=\; P(n, m)^T \;=\; P(n, m)^{-1}$. An important property of $P(m, n)$ is as follows:

Lemma 2.2. *For any $A \in \mathbb{R}^{m \times n}$ and $B \in \mathbb{R}^{p \times q}$, we have*

$$B \otimes A \;=\; P(n,p)^T \, (A \otimes B) \, P(n,q). \tag{2.2}$$

The vector operator $\mathrm{vec}(\cdot)$ is a linear operator that maps any $A = [a_{ij}] \in \mathbb{R}^{m \times n}$ to the column vector

$$\mathrm{vec}(A) \;=\; [a_{11} \; \dots \; a_{m1} \, a_{12} \; \dots \; a_{m2} \; \dots \; a_{1n} \; \dots \; a_{mn}]^T \;\in\; \mathbb{R}^{mn}.$$

Lemma 2.3. *For any matrices A, B, C of compatible dimensions, we have*

$$\mathrm{vec}\,(ABC) \;=\; \left(C^T \otimes A\right) \mathrm{vec}(B). \tag{2.3}$$

Lemma 2.4. *For any matrix $X \in \mathbb{R}^{m \times n}$, we have*

$$\mathrm{vec}(X^T) \;=\; P(m,n)\,\mathrm{vec}(X). \tag{2.4}$$

Lemma 2.5. *For any matrices A, B, X, Y of compatible dimensions, we have*

$$(\mathrm{vec}\,(Y))^T \,(A \otimes B)\,\mathrm{vec}\,(X) \;=\; \mathrm{tr}\left(A^T Y^T B X\right). \tag{2.5}$$

This material is reserved for educational use only, not allowed for commercial use.
Forbidden to modify the content, and cite the document when use.

The Frobenius norm of $A \in \mathbb{R}^{m\times n}$ is defined by

$$\|A\| = \left(\sum_{i=1}^{m}\sum_{j=1}^{n} a_{ij}^2\right)^{\frac{1}{2}} = \left(\operatorname{tr}\left(A^T A\right)\right)^{\frac{1}{2}}.$$

3 A direct method to solve a generalized Sylvester-transpose matrix eqiation

Consider the matrix equation (1.8) when A_i, B_i, C_j, D_j, $E \in \mathbb{R}^{n\times n}$ for any $i, j = 1, 2, \ldots, p$. We apply the vector operator to Eq. (1.8) to get

$$\begin{aligned}
\operatorname{vec}(E) &= \operatorname{vec}\left(\sum_{i=1}^{p} A_i X B_i + \sum_{j=1}^{q} C_j X^T D_j\right) \\
&= \sum_{i=1}^{p} \left(B_i^T \otimes A_i\right) \operatorname{vec}(X) + \sum_{j=1}^{q} \left(D_j^T \otimes C_j\right) \operatorname{vec}(X^T) \\
&= \sum_{i=1}^{p} \left(B_i^T \otimes A_i\right) \operatorname{vec}(X) + \sum_{j=1}^{q} \left(D_j^T \otimes C_j\right) P(n,n) \operatorname{vec}(X) \\
&= \left(\sum_{i=1}^{p} \left(B_i^T \otimes A_i\right) + \sum_{j=1}^{q} \left(D_j^T \otimes C_j\right) P(n,n)\right) \operatorname{vec}(X). \qquad (3.1)
\end{aligned}$$

Let us denote

$$x = \operatorname{vec}(X), \quad b = \operatorname{vec}(E), \quad M = \sum_{i=1}^{p} \left(B_i^T \otimes A_i\right) + \sum_{j=1}^{q} \left(D_j^T \otimes C_j\right) P(n,n). \qquad (3.2)$$

Thus, Eq. (1.8) is equivalently transformed to a linear algebraic system $Mx = b$. Hence, Eq.(1.8) has a unique solution if and only if the matrix M is invertible. In this case, the solution is $x = M^{-1}b$, and we can get the unknown matrix using the injectivity of the vector operator.

4 A conjugate gradient algorithm for a generalized Sylvester-transpose matrix equation

In this section, we propose an iterative algorithm for solving the matrix equation (1.8). We suppose that the matrix M is invertible, so that Eq. (1.8) has a unique solution. To derive an iterative procedure, we apply the idea of a conjugate gradient algorithm. Indeed, for a consistent linear system

$$Ku = f \qquad (4.1)$$

This material is reserved for educational use only, not allowed for commercial use.
Forbidden to modify the content, and cite the document when use.

with $n \times n$ positive definite coefficient matrix K, we have Algorithm 1.

Algorithm 1: A conjugate gradient algorithm for the linear system (4.1)

Start: Choose u_0, e.g. $u_0 = 0$
set $v_0 = 0$
for $k = 0$ **to** $n-1$ **do**
  set $r_k = f - Ku_k$;
  if $r_k = 0$ **then**
    print 'u_k is the exact solution' ; break
  else
    if $k = 0$ **then**
      set $v_1 = r_0$
    else
      set $v_{k+1} = r_k + \frac{\|r_k\|^2}{\|r_{k-1}\|^2} v_k$
    end
  end
  set $u_{k+1} = u_k + \frac{\|r_k\|^2}{v_{k+1}^T K v_{k+1}} v_{k+1}$
  next k
end

From now on, suppose that the matrix M is invertible and symmetric. We propose the following algorithm:

Algorithm 2: A conjugate gradient algorithm for Eq. (1.8)

Start: Set $k = 0$
$X_0 = 0, R_0 = E, P_1 = R_0$;
for $k = 0$ **to** $n^2 - 1$ **do**
  if $R_k = 0$ **then**
    X_k is the exact solution; break
  else
    $P_{k+1} = R_k + \frac{\|R_k\|^2}{\|R_{k-1}\|^2} P_k$
    $Q_{k+1} = \sum_{i=1}^{p} A_i P_{k+1} B_i + \sum_{j=1}^{q} C_j P_{k+1}^T D_j$
    $\alpha_{k+1} = \text{tr}\left(P_{k+1}^T Q_{k+1}\right)$
    $X_{k+1} = X_k + \frac{\|R_k\|^2}{\alpha_{k+1}} P_{k+1}$
    $R_{k+1} = E - \sum_{i=1}^{p} A_i X_{k+1} B_i - \sum_{j=1}^{q} C_j X_{k+1}^T D_j$
    next k
  end
end

This material is reserved for educational use only, not allowed for commercial use.
Forbidden to modify the content, and cite the document when use.

From the main computation

$$X_{k+1} = X_k + \frac{\|R_k\|^2}{\alpha_{k+1}} P_{k+1},$$

we call X_{k+1} the next point, X_k the current point, and P_{k+1} the search direction. The step size $\|R_k\|^2/\alpha_{k+1}$ is chosen so that the set $\{R_i\}$ of residual matrices is orthogonal, as we shall see later. To avoid roundoff errors from computations, we may introduce a small tolerance error $\varepsilon > 0$ and use $\|R_k\| < \varepsilon$ to be the stopping rule, instead of the condition $R_k = 0$.

Next, we shall show that Algorithm 2 leads to the exact solution in at most $n^2 + 1$ iterations. We divide the proof into several lemmas as follows.

Lemma 4.1. *Suppose that the sequence $\{R_i\}$ is generated by Algorithm 2. We have*

$$R_{k+1} = R_k - \frac{\|R_k\|^2}{\alpha_{k+1}} Q_{k+1} \quad \text{for} \quad k = 1, 2, ... \tag{4.2}$$

Proof. From Algorithm 2 , we have that for any $k \in \mathbb{N}$,

$$\begin{aligned} R_{k+1} &= E - \sum_{i=1}^{p} A_i X_{k+1} B_i \sum_{j=1}^{q} C_j X_{k+1}^T D_j \\ &= E - \sum_{i=1}^{p} A_i \left(X_k + \frac{\|R_k\|^2}{\alpha_{k+1}} P_{k+1} \right) B_i - \sum_{j=1}^{q} C_j \left(X_k + \frac{\|R_k\|^2}{\alpha_{k+1}} P_{k+1} \right)^T D_j \\ &= E - \sum_{i=1}^{p} A_i X_k B_i - \sum_{j=1}^{q} C_j X_k^T D_j - \frac{\|R_k\|^2}{\alpha_{k+1}} \left(\sum_{i=1}^{p} A_i P_{k+1} B_i + \sum_{j=1}^{q} C_j P_{k+1}^T D_j \right) \\ &= R_k - \frac{\|R_k\|^2}{\alpha_{k+1}} Q_{k+1}. \end{aligned}$$

Thus, Eq (4.1) holds. □

Lemma 4.2. *The sequences $\{P_m\}$ and $\{Q_m\}$ generated by Algorithm 2 satisfy*

$$\text{tr}\left(P_s^T Q_t\right) = \text{tr}\left(Q_s^T P_t\right) \tag{4.3}$$

for any $s, t \in \mathbb{N}$.

This material is reserved for educational use only, not allowed for commercial use.
Forbidden to modify the content, and cite the document when use.

Proof. We apply Lemmas 2.1-2.5. Indeed, we have

$$\begin{aligned}
\text{tr}\left(P_s^T Q_t\right) &= \text{tr}\left[P_s^T\left(\sum_{i=1}^{p} A_i P_t B_i + \sum_{j=1}^{q} C_j P_t^T D_j\right)\right] \\
&= \text{vec}(P_s)\left(\sum_{i=1}^{p}\left(B_i^T \otimes A_i\right) + \sum_{j=1}^{q}\left(D_j^T \otimes C_j\right) P(n,n)\right)\text{vec}(P_t) \\
&= \text{vec}(P_s)\left(\sum_{i=1}^{p}\left(B_i \otimes A_i^T\right) + P(n,n)\sum_{j=1}^{q}\left(D_j \otimes C_j^T\right)\right)\text{vec}(P_t) \\
&= \text{vec}(P_s)\left(\sum_{i=1}^{p}\left(B_i \otimes A_i^T\right)\right)\text{vec}(P_t) \\
&\quad + \text{vec}(P_s)\left(P(n,n)\sum_{j=1}^{q}\left(D_j \otimes C_j^T\right)\right)\left(P(n,n)\,\text{vec}(P_t^T)\right) \\
&= \text{tr}\left(\sum_{i=1}^{p} P_t^T A_i P_s B_i\right) + \text{tr}\left(\sum_{i=1}^{p} P_t^T C_j P_s^T D_i\right) \\
&= \text{tr}\left(P_t^T Q_s\right) \\
&= \text{tr}\left(Q_s^T P_t\right).
\end{aligned}$$

Hence, Eq (4.2) holds. □

Lemma 4.3. *The sequences $\{R_m\}$, $\{P_m\}$ and $\{Q_m\}$ are generated by Algorithm 2 satisfy*

$$\text{tr}\left(R_m^T R_{m-1}\right) = 0, \qquad \text{tr}\left(P_{m+1}^T Q_m\right) = 0 \tag{4.4}$$

for any m.

Proof. To prove this conclusion, we use induction principle. In order to compute related terms, we use Lemmas 4.1 and 4.2.

First, for the initial step $m = 1$, we have

$$\begin{aligned}
\text{tr}\left(R_1^T R_0\right) &= \text{tr}\left(R_0^T R_0\right) - \frac{\|R_0\|^2}{\alpha_1}\text{tr}\left(Q_1^T R_0\right) \\
&= \|R_0\|^2 - \|R_0\|^2 \\
&= 0,
\end{aligned}$$

This material is reserved for educational use only, not allowed for commercial use.
Forbidden to modify the content, and cite the document when use.

and

$$\begin{aligned}\operatorname{tr}\left(P_2^TQ_1\right) &= \operatorname{tr}\left(R_1^TQ_1\right)+\frac{\|R_1\|^2}{\|R_0\|^2}\operatorname{tr}\left(P_1^TQ_1\right)\\ &= -\frac{\alpha_1}{\|R_0\|^2}\operatorname{tr}\left(R_1^TR_1\right)+\frac{\|R_1\|^2}{\|R_0\|^2}\alpha_1\\ &= 0.\end{aligned}$$

Second, suppose that (4.4) holds for $m = k$, that is $(R_k^TR_{k-1}) = 0$. Then for $m = k+1$,

$$\begin{aligned}\operatorname{tr}\left(R_{k+1}^TR_k\right) &= \operatorname{tr}\left(R_k^TR_k\right)-\frac{\|R_k\|^2}{\alpha_{k+1}}\operatorname{tr}\left(Q_{k+1}^TR_k\right)\\ &= \|R_k\|^2-\frac{\|R_k\|^2}{\alpha_{k+1}}\operatorname{tr}\left(Q_{k+1}^TP_{k+1}\right)\\ &= 0,\end{aligned}$$

and we have

$$\begin{aligned}\operatorname{tr}\left(P_{k+2}^TQ_{k+1}\right) &= \operatorname{tr}\left(R_{k+1}^TQ_{k+1}\right)+\frac{\|R_{k+1}\|^2}{\|R_k\|^2}\operatorname{tr}\left(P_{k+1}^TQ_{k+1}\right)\\ &= \frac{\alpha_{k+1}}{\|R_k\|^2}\operatorname{tr}\left[\left(R_{k+1}^TR_k\right)-\left(R_{k+1}^TR_{k+1}\right)\right]+\frac{\|R_{k+1}\|^2}{\|R_k\|^2}\alpha_{k+1}\\ &= 0.\end{aligned}$$

Hence, the conclusion (4.4) holds. □

Lemma 4.4. *Suppose the sequences $\{R_n\}$, $\{P_n\}$ and $\{Q_n\}$ are generated by Algorithm 2.Then*

$$\operatorname{tr}\left(R_n^TR_0\right) = 0, \qquad \operatorname{tr}\left(P_{n+1}^TQ_1\right) = 0 \tag{4.5}$$

for any n.

Proof. We prove by induction. For $n = 1$, we have $\operatorname{tr}\left(R_1^TR_0\right) = 0$ and $\operatorname{tr}\left(P_2^TQ_1\right) = 0$ by Lemma 4.3. Now, suppose that Eq. (4.5) is true for all $n = 1,\dots,m$. From Lemmas 4.1 and

This material is reserved for educational use only, not allowed for commercial use.
Forbidden to modify the content, and cite the document when use.

4.2, for $n = m+1$, we have

$$\begin{aligned}\operatorname{tr}\left(R_{m+1}^T R_0\right) &= \operatorname{tr}[R_m^T R_0] - \frac{\|R_m\|^2}{\alpha_{m+1}} \operatorname{tr}(Q_{m+1}^T R_0) \\ &= -\frac{\|R_m\|^2}{\alpha_{m+1}} \operatorname{tr}(Q_{m+1}^T P_1) \\ &= -\frac{\|R_m\|^2}{\alpha_{m+1}} \operatorname{tr}(P_{m+1}^T Q_1) \\ &= 0,\end{aligned}$$

and

$$\begin{aligned}\operatorname{tr}\left(P_{m+2}^T Q_1\right) &= \operatorname{tr}(R_{m+1}^T Q_1) + \frac{\|R_{m+1}\|^2}{\|R_m\|^2} \operatorname{tr}(P_{m+1}^T Q_1) \\ &= \operatorname{tr}(R_{m+1}^T Q_1) \\ &= \operatorname{tr}(R_m^T Q_1) - \frac{\|R_m\|^2}{\alpha_{m+1}} \operatorname{tr}(P_m^T Q_1) \\ &= \operatorname{tr}(P_{m+1}^T Q_1) - \frac{\|R_m\|^2}{\|R_{m-1}\|^2} \operatorname{tr}(P_m^T Q_1) \\ &= 0.\end{aligned}$$

Therefore, the conclusion (4.5) holds for any n. □

Theorem 4.5. *Suppose the sequences $\{R_s\}$, $\{P_s\}$ and $\{Q_s\}$ are generated by Algorithm 2. Then for any s, t such that $s \neq t$, we have*

$$\operatorname{tr}\left(R_{s-1}^T R_{t-1}\right) = 0 \quad \text{and} \quad \operatorname{tr}\left(P_s^T Q_t\right) = 0. \tag{4.6}$$

Proof. We prove by using induction. Lemma 4.3 assures that the two equalities in Eq. (4.6) hold for $s = t+1$. Before we process induction step, we consider

$$\begin{aligned}\operatorname{tr}\left(R_{t+2}^T R_t\right) &= \operatorname{tr}(R_{t+1}^T R_t) - \frac{\|R_{t+1}\|^2}{\alpha_{t+2}\|^2} \operatorname{tr}(Q_{t+2}^T R_t) \\ &= -\frac{\|R_{t+1}\|^2}{\alpha_{t+2}} \left[\operatorname{tr}\left(Q_{t+2}^T P_{t+1}\right) - \frac{\|R_t\|^2}{\|R_{t-1}\|^2} \operatorname{tr}\left(Q_{t+2}^T P_t\right)\right] \\ &= \frac{\|R_{t+1}\|^2}{\alpha_{t+2}} \frac{\|R_t\|^2}{\|R_{t-1}\|^2} \left[\operatorname{tr}\left(R_{t+1}^T Q_t\right) - \frac{\|R_{t+1}\|^2}{\|R_l\|^2} \operatorname{tr}\left(P_{t+1}^T Q_t\right)\right] \\ &= \frac{\|R_{t+1}\|^2}{\alpha_{t+2}} \frac{\|R_t\|^2}{\|R_{t-1}\|^2} \frac{\alpha_t}{\|R_{t-1}\|^2} \left[\operatorname{tr}(R_{t+1}^T R_{t-1}) - \operatorname{tr}(R_{t+1}^T R_t)\right] \\ &= \frac{\|R_{t+1}\|^2}{\alpha_{t+2}} \frac{\|R_t\|^2}{\|R_{t-1}\|^2} \frac{\alpha_t}{\|R_{t-1}\|^2} \operatorname{tr}(R_{t+1}^T R_{t-1}),\end{aligned}$$

This material is reserved for educational use only, not allowed for commercial use.
Forbidden to modify the content, and cite the document when use.

$$\begin{aligned}
\operatorname{tr}\left(P_{t+2}^T Q_t\right) &= \operatorname{tr}(R_{t+1}^T Q_t) + \frac{\|R_{t+1}\|^2}{\|R_t\|^2}\operatorname{tr}(P_{t+1}^T Q_t) \\
&= \frac{\alpha_t}{\|R_{t-1}\|^2}\left[\operatorname{tr}\left(R_{t+1}^T R_{t-1}\right) - \operatorname{tr}\left(R_{t+1}^T R_t\right)\right] \\
&= \frac{\alpha_t}{\|R_{t-1}\|^2}\left[\operatorname{tr}\left(R_t^T R_{t-1}\right) - \frac{\|R_t\|^2}{\alpha_{t+1}}\operatorname{tr}\left(Q_{t+1}^T R_{t-1}\right)\right] \\
&= -\frac{\alpha_t}{\|R_{t-1}\|^2}\frac{\|R_t\|^2}{\alpha_{t+1}}\left[\operatorname{tr}\left(Q_{t+1}^T P_t\right) - \frac{\|R_{t-1}\|^2}{\|R_{t-2}\|^2}\operatorname{tr}\left(Q_{t+1}^T P_{t-1}\right)\right] \\
&= \frac{\alpha_t}{\|R_{t-1}\|^2}\frac{\|R_t\|^2}{\alpha_{t+1}}\frac{\|R_{t-1}\|^2}{\|R_{t-2}\|^2}\operatorname{tr}\left(P_{t+1}^T Q_{t-1}\right),
\end{aligned}$$

$$\begin{aligned}
\operatorname{tr}\left(R_{t+1}^T R_{t-1}\right) &= \operatorname{tr}(R_t^T R_{t-1}) - \frac{\|R_t\|^2}{\alpha_{t+1}\|^2}\operatorname{tr}(Q_{t+1}^T R_{t-1}) \\
&= \frac{\|R_t\|^2}{\alpha_{t+1}}\frac{\|R_{t-1}\|^2}{\|R_{t-2}\|^2}\left[\operatorname{tr}\left(R_t^T Q_{t-1}\right) - \frac{\|R_t\|^2}{\|R_{t-1}\|^2}\operatorname{tr}\left(P_t^T Q_{t-1}\right)\right] \\
&= \frac{\|R_t\|^2}{\alpha_{t+1}}\frac{\|R_{t-1}\|^2}{\|R_{t-2}\|^2}\frac{\alpha_{t-1}}{\|R_{t-2}\|^2}\operatorname{tr}(R_t^T R_{t-2}),
\end{aligned}$$

and

$$\begin{aligned}
\operatorname{tr}\left(P_{t+1}^T Q_{t-1}\right) &= \operatorname{tr}(R_t^T Q_{t-1}) + \frac{\|R_t\|^2}{\|R_{t-1}\|^2}\operatorname{tr}(P_t^T Q_{t-1}) \\
&= \frac{\alpha_{t-1}}{\|R_{t-2}\|^2}\left[\operatorname{tr}\left(R_t^T R_{t-2}\right) - \operatorname{tr}\left(R_t^T R_{t-1}\right)\right] \\
&= -\frac{\alpha_{t-1}}{\|R_{t-2}\|^2}\frac{\|R_{t-1}\|^2}{\alpha_t}\left[\operatorname{tr}\left(Q_t^T P_{t-1}\right) - \frac{\|R_{t-2}\|^2}{\|R_{t-3}\|^2}\operatorname{tr}\left(Q_t^T P_{t-2}\right)\right] \\
&= \frac{\alpha_{t-1}}{\|R_{t-2}\|^2}\frac{\|R_{t-1}\|^2}{\alpha_t}\frac{\|R_{t-2}\|^2}{\|R_{t-3}\|^2}\operatorname{tr}\left(P_t^T Q_{t-2}\right).
\end{aligned}$$

Similarly, we can write $\operatorname{tr}\left(R_{t+2}^T R_t\right)$ and $\operatorname{tr}\left(P_{t+2}^T Q_t\right)$ in terms of $\operatorname{tr}\left(R_{t+1}^T R_{t-1}\right)$ and $\operatorname{tr}\left(P_{t+1}^T Q_{t-1}\right)$, respectively. Repeating this process until the terms $\operatorname{tr}\left(P_3^T Q_1\right)$ and $\operatorname{tr}\left(R_2^T R_0\right)$ show up. By Lemma 4.4, we get $\operatorname{tr}\left(R_{t+1}^T R_{t-1}\right) = 0$ and $\operatorname{tr}\left(P_{t+1}^T Q_{t-1}\right) = 0$.

This material is reserved for educational use only, not allowed for commercial use.
Forbidden to modify the content, and cite the document when use.

Next, we compute

$$\begin{aligned}
\operatorname{tr}\left(R_{t+3}^T R_t\right) &= \operatorname{tr}(R_{t+2}^T R_t) - \frac{\|R_{t+2}\|^2}{\alpha_{t+3}} \operatorname{tr}(Q_{t+3}^T R_t) \\
&= -\frac{\|R_{t+2}\|^2}{\alpha_{t+3}} \left[\operatorname{tr}\left(Q_{t+3}^T P_{t+1}\right) - \frac{\|R_t\|^2}{\|R_{t-1}\|^2} \left(Q_{t+3}^T P_t\right)\right] \\
&= \frac{\|R_{t+2}\|^2}{\alpha_{t+3}} \frac{\|R_t\|^2}{\|R_{t-1}\|^2} \left(P_{t+3}^T Q_t\right) \\
&= -\frac{\|R_{t+2}\|^2}{\alpha_{t+3}} \frac{\|R_t\|^2}{\|R_{t-1}\|^2} \left[\operatorname{tr}\left(R_{t+2}^T Q_t\right) + \frac{\|R_{t+2}\|^2}{\|R_{t+1}\|^2} \operatorname{tr}\left(P_{t+2}^T Q_t\right)\right] \\
&= -\frac{\|R_{t+2}\|^2}{\alpha_{t+3}} \frac{\|R_t\|^2}{\|R_{t-1}\|^2} \frac{\alpha_t}{\|R_{t-1}\|^2} \operatorname{tr}\left(R_{t+2}^T R_{t-1}\right),
\end{aligned}$$

and

$$\begin{aligned}
\operatorname{tr}\left(P_{t+3}^T Q_t\right) &= \operatorname{tr}(R_{t+2}^T Q_t) + \frac{\|R_{t+2}\|^2}{\|R_{t+1}\|^2} \operatorname{tr}(P_{t+2}^T Q_t) \\
&= \frac{\alpha_t}{\|R_{t-1}\|^2} \left[\operatorname{tr}\left(R_{t+2}^T R_{t-1}\right) - \operatorname{tr}\left(R_{t+2}^T R_t\right)\right] \\
&= \frac{\alpha_t}{\|R_{t-1}\|^2} \left[\operatorname{tr}\left(R_{t+1}^T R_{t-1}\right) - \frac{\|R_{t+1}\|^2}{\alpha_{t+2}} \operatorname{tr}\left(Q_{t+2}^T R_{t-1}\right)\right] \\
&= -\frac{\alpha_t}{\|R_{t-1}\|^2} \frac{\|R_{t+1}\|^2}{\alpha_t} \left[\operatorname{tr}\left(Q_{t+2}^T P_t\right) - \frac{\|R_{t-1}\|^2}{\|R_{t-2}\|^2} \operatorname{tr}\left(Q_{t+2}^T P_{t-1}\right)\right] \\
&= \frac{\alpha_t}{\|R_{t-1}\|^2} \frac{\|R_{t+1}\|^2}{\alpha_{t-1}} \frac{\|R_{t-1}\|^2}{\|R_{t-2}\|^2} \operatorname{tr}\left(P_{t+2}^T Q_{t-1}\right).
\end{aligned}$$

Also, we have

$$\begin{aligned}
\operatorname{tr}\left(R_{t+2}^T R_{t-1}\right) &= \operatorname{tr}(R_{t+1}^T R_{t-1}) - \frac{\|R_{t+1}\|^2}{\alpha_{t+2}} \operatorname{tr}(Q_{t+2}^T R_{t-1}) \\
&= -\frac{\|R_{t+1}\|^2}{\alpha_{t+2}} \left[\operatorname{tr}\left(Q_{t+2}^T P_t\right) - \frac{\|R_{t-1}\|^2}{\|R_{t-2}\|^2} \left(Q_{t+2}^T P_t\right)\right] \\
&= \frac{\|R_{t+1}\|^2}{\alpha_{t+2}} \frac{\|R_{t-1}\|^2}{\|R_{t-2}\|^2} \left(P_{t+2}^T Q_{t-1}\right) \\
&= -\frac{\|R_{t+1}\|^2}{\alpha_{t+2}} \frac{\|R_{t-1}\|^2}{\|R_{t-2}\|^2} \left[\operatorname{tr}\left(R_{t+1}^T Q_t\right) + \frac{\|R_{t+1}\|^2}{\|R_t\|^2} \operatorname{tr}\left(P_{t+1}^T Q_{t-1}\right)\right] \\
&= -\frac{\|R_{t+1}\|^2}{\alpha_{t+2}} \frac{\|R_{t-1}\|^2}{\|R_{t-2}\|^2} \frac{\alpha_{t-1}}{\|R_{t-2}\|^2} \operatorname{tr}\left(R_{t+1}^T R_{t-2}\right),
\end{aligned}$$

This material is reserved for educational use only, not allowed for commercial use.
Forbidden to modify the content, and cite the document when use.

$$\begin{aligned}
\text{tr}\left(P_{t+2}^T Q_{t-1}\right) &= \text{tr}(R_{t+1}^T Q_{t-1}) + \frac{\|R_{t+1}\|^2}{\|R_t\|^2}\,\text{tr}(P_{t+1}^T Q_{t-1}) \\
&= \frac{\alpha_{t-1}}{\|R_{t-2}\|^2}\left[\text{tr}\left(R_{t+1}^T R_{t-2}\right) - \text{tr}\left(R_{t+1}^T R_{t-1}\right)\right] \\
&= \frac{\alpha_{t-1}}{\|R_{t-2}\|^2}\left[\text{tr}\left(R_t^T R_{t-2}\right) - \frac{\|R_t\|^2}{\alpha_{t+1}}\,\text{tr}\left(Q_{t+1}^T R_{t-2}\right)\right] \\
&= -\frac{\alpha_{t-1}}{\|R_{t-2}\|^2}\frac{\|R_t\|^2}{\alpha_{t+1}}\left[\text{tr}\left(Q_{t+1}^T P_{t-1}\right) - \frac{\|R_{t-2}\|^2}{\|R_{t-3}\|^2}\,\text{tr}\left(Q_{t+1}^T P_{t-2}\right)\right] \\
&= \frac{\alpha_{t-1}}{\|R_{t-2}\|^2}\frac{\|R_t\|^2}{\alpha_{t+1}}\frac{\|R_{t-2}\|^2}{\|R_{t-3}\|^2}\,\text{tr}\left(P_{t+1}^T Q_{t-2}\right).
\end{aligned}$$

Hence, we can write $\text{tr}\left(R_{t+3}^T R_t\right)$ and $\text{tr}\left(P_{t+3}^T Q_t\right)$ in terms of $\text{tr}\left(R_{t+2}^T R_{t-1}\right)$ and $\text{tr}\left(P_{t+2}^T Q_{t-1}\right)$. We use this idea to prove the conclusion (4.6).

Suppose that $\text{tr}\left(R_{s-1}^T R_{t-1}^T\right) = 0$ and $\text{tr}\left(P_s^T Q_t^T\right) = 0$ for $s = t+1, \ldots, m$. Then for $s = m+1$, we get

$$\begin{aligned}
\text{tr}\left(R_m^T R_{t-1}\right) &= \text{tr}\left(R_{m-1}^T R_{t-1}\right) - \frac{\|R_{m-1}\|^2}{\alpha_{m-1}}\,\text{tr}\left(Q_{m-1}^T R_{t-1}\right) \\
&= -\frac{\|R_{m-1}\|^2}{\alpha_{m-1}}[\text{tr}\left(Q_{m-1}^T P_t\right) - \frac{\|R_{t-1}\|^2}{\alpha_{t-2}}\left(Q_{m-1}^T P_{t-1}\right)] \\
&= 0,
\end{aligned}$$

and thus

$$\begin{aligned}
\text{tr}\left(P_{m+1}^T Q_t\right) &= \text{tr}\left(R_m^T Q_t\right) + \frac{\|R_m\|^2}{\alpha_m - 1}\,\text{tr}\left(P_m^T Q_t\right) \\
&= \frac{\alpha_t}{\|R_{t-1}\|^2}\left[\text{tr}\left(R_m^T R_{t-1}\right) - \text{tr}\left(R_m^T R_t\right)\right] \\
&= 0.
\end{aligned}$$

Hence, by induction principle, the conclusion (4.6) holds for any s, t such that $s \neq t$. □

Theorem 4.6. *Suppose that the sequence $\{X_k\}$ is generated by Algorithm 2. Then for initial zero matrix $X_0 = 0$, the exact solution X can be obtained in at most n^2 iteration steps.*

Proof. Suppose that $R_m \neq 0$ for $m = 0, 1, \ldots, n^2 - 1$. Then we compute X_{n^2}. Assume that $R_{n^2} \neq 0$. By Theorem 4.5, the set $\{R_0, R_1, ..., R_{n^2}\}$ is orthogonal in $\mathbb{R}^{n\times n}$. So, $\{R_0, R_1, ..., R_{n^2}\}$ is linearly independent. Since the dimension of $\mathbb{R}^{n\times n}$ is n^2, any linearly independent subset of $\mathbb{R}^{n\times n}$ must have at most n^2 elements. So this is false because the set $\{R_0, R_1, ..., R_{n^2}\}$ has $n^2 + 1$ elements. Thus, $R_{n^2} = 0$, and hence X_{n^2} is a solution of the equation. □

This material is reserved for educational use only, not allowed for commercial use.
Forbidden to modify the content, and cite the document when use.

5 Conclusion

We propose an iterative procedure (Algorithm 2) to construct a sequence of approximate solutions for the generalized Sylvester-transpose matrix equation (1.8). The algorithm is applicable whenever the matrix M, defined by Eq. (3.2), is invertible and symmetric, not necessarily positive definite. We get the exact solution within n^2 iterations. Moreover, Algorithm 2 can be applied to any special case of Eq. (1.8), e.g. Eqs. (1.1)-(1.7).

References

[1] M. Dehghan, M Hajarian, *An iterative algorithm for solving a pair of matrix equations $AYB = E, CYD = F$ over generalized centro-symmetric matrices*, Comput. Math. Appl., **12** (2008), 3246–3260.

[2] F. Ding, T. Chen, *Iterative least squares solutions of coupled Sylvester matrix equations*, Systems Control Lett., **5** (2005), 95–107.

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

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

[5] Y.J. Xie ,C.F. Ma , *The accelerated gradient based iterative algorithm for solving a class of generalized Sylvester-transpose matrix equation*, Appl. Math. Comp., **273** (2016), 1257–1269.

[6] Y.F. Ke, C.F. Ma, *A preconditioned nested splitting conjugate gradient iterative method for the large sparse generalized Sylvester equation*, Appl. Math. Comput., **10** (2014), 1409–1420.

[7] Y. Lei, A. Liao, *A minimal residual algorithm for the inconsistent matrix equation $AXB = C$ over symmetric matrices*, Appl. Math. Comput, **188** (2007), 499–513.

[8] S. Li, T. Huang, *A shift-splitting Jacobi-gradient algorithm for Lyapunov matrix equation arising form control theory*, J.Comput. Anal. Appl., **13** (2011), 1246–1257.

[9] K. Liang, J. Liu, *Iterative algorithms for the minimum-norm solution and the least-squares solution of the linear matrix equations*,Appl. Math. Comput., **218** (2011), 3166–3175.

[10] A. Liao, Z.Z. Bai, Y. Lei, *best approximate solution of matrix equation $AXB + CXD = E$*, SIAM Journal on Matrix Analysis and Applications,**27** (2005), 675–688.

[11] Y. Lin, *Minimal residual methods augmented with eigenvectors for solving Sylvester equations and generalized Sylvester equatios*, Appl. Math. Comput., **181** (2006), 487–499.

[12] Q. Niu, X. Wang, L. Lu, *A relaxed gradient based algorithm for solving Sylvester equation*, Asian J. Control, **13** (2011), 461–464.

[13] P.J. Olver, C. Shakiban, *Applied Linear Algebra*, Pearson Education, 2006.

[14] M. Wang, X. Cheng, M. Wei, *Iterative algorithms for solving the matrix equation $AXB + CX^TD = E$*, Appl. Math. Comput., **187** (2007), 622–629.

This material is reserved for educational use only, not allowed for commercial use.
Forbidden to modify the content, and cite the document when use.

[15] Y.J. Xie, *C.F. Ma, Gradient based and least square based iterative algorithms for matrix equation* $AXB + CX^TB = F$, Appl. Math. Comp, **217** (2010), 2191–2199.

[16] Y.J. Xie, C.F. Ma, *The accelerated gradient based iterative algorithm for solving a class of generalized Sylvester-transpose matrix equation*,Appl. Math. Comp., **218** (2012), 5620–5628.

[17] B. Zhou, G.R. Duan, Z.Y. Li, *Gradient based iterative algorithm for solving coupled matrix equations*, Systems Control Lett, **58** (2009), 327–333.

This material is reserved for educational use only, not allowed for commercial use.
Forbidden to modify the content, and cite the document when use.

NO THIS PAGE IN PROFILE

This material is reserved for educational use only, not allowed for commercial use.
Forbidden to modify the content, and cite the document when use.