Filomat 32:15 (2018), 5307–5318
https://doi.org/10.2298/FIL1815307H

Published by Faculty of Sciences and Mathematics,
University of Niš, Serbia
Available at: http://www.pmf.ni.ac.rs/filomat

Biconjugate Residual Algorithm for Solving General Sylvester-transpose Matrix Equations

Masoud Hajarian[a]

[a]*Department of Mathematics, Faculty of Mathematical Sciences, Shahid Beheshti University, General Campus, Evin, Tehran 19839, Iran*

Abstract. The present paper is concerned with the solution of the coupled generalized Sylvester-transpose matrix equations

$$\begin{cases} A_1XB_1 + C_1XD_1 + E_1X^TF_1 = M_1, \\ A_2XB_2 + C_2XD_2 + E_2X^TF_2 = M_2, \end{cases}$$

including the well-known Lyapunov and Sylvester matrix equations. Based on a variant of biconjugate residual (BCR) algorithm, we construct and analyze an efficient algorithm to find the (least Frobenius norm) solution of the general Sylvester-transpose matrix equations within a finite number of iterations in the absence of round-off errors. Two numerical examples are given to examine the performance of the constructed algorithm.

1. Introduction

Linear matrix equations have variety of applications in applied mathematics and engineering [3–5, 25, 26, 29]. For instance, the Sylvester-transpose matrix equation

$$AX + X^TB = C, \tag{1.1}$$

has close relations with many problems in system theory, such as eigenstructure assignment, observer design, control of system with input constraint, and fault detection [13, 18–20, 32]. The generalized Sylvester matrix equation

$$AXB + CXD = E, \tag{1.2}$$

appears in the model reduction and stability, reachability, observability and controllability analysis of discrete-time and continuous-time linear systems [36, 38–40]. During the last two decades, the linear matrix equations have drawn much attention due to their wide applications [3, 14, 27, 28, 35, 42]. Li and Wang introduced the weighted steepest descent algorithms to solve the general linear matrix equation including the Lyapunov and Sylvester matrix equations [31]. In [37], the explicit expression of least squares Hermitian solution with the least norm of (1.2) was presented. By applying Galerkin projection with an extended Krylov subspace method, Shank et al. proposed an iterative method for the low-rank

2010 *Mathematics Subject Classification*. Primary 65F08; Secondary 15A24.
Keywords. BCR algorithm; General Sylvester-transpose matrix equations; Lyapunov matrix equation; Sylvester matrix equation.
Received: 29 January 2018; Accepted: 04 November 2018
Communicated by Dijana Mosić
Email address: m_hajarian@sbu.ac.ir, mhajarian@aut.ac.ir, masoudhajarian@gmail.com (Masoud Hajarian)

approximate solution of the generalized Lyapunov equations [1]. In [2], Dopico et al. introduced several projection algorithms based on different Krylov spaces to find the low-rank approximations to the solution of the Sylvester-transpose matrix equation. The development of iterative methods proposed for $Ax = b$ has gained interest to compute numerical solutions of linear matrix equations in recent years [12, 21, 33, 45–47]. By developing the conjugate gradient normal equation residual (CGNR), the conjugate gradient normal equation error (CGNE) and the least-squares QR-factorization (LSQR) algorithms, various efficient methods were constructed for solving diverse linear matrix equations [12, 23, 24, 30]. In [15–17], by applying the hierarchical identification principle, the gradient based iterative algorithms were proposed for solving several linear matrix equations.

Our main result in this paper is to generalize a variant of BCR algorithm to compute the solution $X \in \mathrm{R}^{m\times n}$ of the coupled generalized Sylvester-transpose matrix equations

$$\begin{cases} A_1XB_1 + C_1XD_1 + E_1X^TF_1 = M_1, \\ A_2XB_2 + C_2XD_2 + E_2X^TF_2 = M_2, \end{cases} \tag{1.3}$$

where $A_i, C_i \in \mathrm{R}^{p_i\times m}$, $B_i, D_i \in \mathrm{R}^{n\times q_i}$, $E_i \in \mathrm{R}^{p_i\times n}$, $F_i \in \mathrm{R}^{m\times q_i}$ and $M_i \in \mathrm{R}^{p_i\times q_i}$ for $i = 1, 2$.

The coupled generalized Sylvester-transpose matrix equations (1.3) contain several matrix equations as special cases such as Lyapunov, Sylvester and Sylvester-transpose matrix equations.

The rest of this article is organized as follows. In Section 2, after a brief presentation of BCR algorithm, we obtain the generalization of the algorithm to solve the coupled generalized Sylvester-transpose matrix equations (1.3). In addition, we discuss the convergence properties of the generalized BCR algorithm. We provide two numerical evaluations of the obtained algorithm in Section 3. In Section 4, conclusions are given.

Some notations used in the sequel are presented as follows:

The symbols $\mathrm{tr}(A)$, A^T, $\mathcal{R}(A)$ and $A^\dagger$ are used to denote the trace, the transpose, the column space and pseudo inverse of A, respectively. The inner product of $A \in \mathrm{R}^{m\times n}$ and $B \in \mathrm{R}^{m\times n}$ is defined by $\langle A, B\rangle = \mathrm{tr}(B^TA)$. The associated norm is the well-known Frobenius norm denoted by $\|.\|$. For a matrix $A \in \mathrm{R}^{m\times n}$, the so-called stretching function $\mathrm{vec}(A)$ is defined by $\mathrm{vec}(A) = (a_1^T, a_2^T, ..., a_n^T)^T$, where a_k is the k-th column of A. The notation $A \otimes B$ stands for the Kronecker product of matrices A and B.

2. Description of the BCR Algorithm and its Convergence

In this section, we first briefly describe the BCR algorithm to solve $Ax = b$. Then we develop the new modified BCR algorithm to calculate the solution of the coupled generalized Sylvester-transpose matrix equations (1.3). Various iterative methods have been introduced for the solution of the nonsymmetric linear system $Ax = b$; see, e.g., [6–10, 43, 44] and references therein. Vespucci and Broyden in [34] presented the different computational variations of BCR algorithm without any convergence analysis. One of those variations is summarized as follows:

BCR algorithm

```
Initial values: $x(1)$ and $s(1)$ arbitrary, $r(1) = Ax(1) - b$,
$u(1) = s(1)$, $v(1) = r(1)$, $w(1) = Au(1)$, and $z(1) = A^Tv(1)$.
Recursions:
$\alpha(k) = \frac{w(k)^Tr(k)}{w(k)^Tw(k)}$, $x(k+1) = x(k) - \alpha(k)u(k)$,
$r(k+1) = r(k) - \alpha(k)w(k)$,
$\beta(k) = \frac{z(k)^Ts(k)}{z(k)^Tz(k)}$, $s(k+1) = s(k) - \beta(k)z(k)$,
$\gamma(k) = \frac{w(k)^TAs(k+1)}{w(k)^Tw(k)}$, $u(k+1) = s(k+1) - \gamma(k)u(k)$,
$\eta(k) = \frac{z(k)^TA^Tr(k+1)}{z(k)^Tz(k)}$, $v(k+1) = r(k+1) - \eta(k)v(k)$,
$w(k+1) = Au(k+1)$,
$z(k+1) = A^Tv(k+1)$.
```

In order to develop the above algorithm for solving (1.3), first we transform (1.3) into a linear system. By utilizing the stretching function and the Kronecker product, the coupled generalized Sylvester-transpose matrix equations (1.3) can be equivalently transformed into the following linear system

$$\underbrace{\begin{pmatrix} B_1^T \otimes A_1 + D_1^T \otimes C_1 + (F_1^T \otimes E_1)P \\ B_2^T \otimes A_2 + D_2^T \otimes C_2 + (F_2^T \otimes E_2)P \end{pmatrix}}_{A} \underbrace{\text{vec}(X)}_{x} = \underbrace{\begin{pmatrix} \text{vec}(M_1) \\ \text{vec}(M_2) \end{pmatrix}}_{b}, \tag{2.1}$$

where $P \in \mathrm{R}^{mn\times mn}$ is an orthogonal matrix. For more details about the form of the matrix P see [41]. Substituting the above system in the BCR algorithm leads to

$$\begin{aligned} r(1) = Ax(1) - b &= \begin{pmatrix} B_1^T \otimes A_1 + D_1^T \otimes C_1 + (F_1^T \otimes E_1)P \\ B_2^T \otimes A_2 + D_2^T \otimes C_2 + (F_2^T \otimes E_2)P \end{pmatrix} x(1) - \begin{pmatrix} \text{vec}(M_1) \\ \text{vec}(M_2) \end{pmatrix}, \\ w(k+1) = Au(k+1) &= \begin{pmatrix} B_1^T \otimes A_1 + D_1^T \otimes C_1 + (F_1^T \otimes E_1)P \\ B_2^T \otimes A_2 + D_2^T \otimes C_2 + (F_2^T \otimes E_2)P \end{pmatrix} u(k+1), \\ z(k+1) = A^T v(k+1) &= \begin{pmatrix} B_1^T \otimes A_1 + D_1^T \otimes C_1 + (F_1^T \otimes E_1)P \\ B_2^T \otimes A_2 + D_2^T \otimes C_2 + (F_2^T \otimes E_2)P \end{pmatrix}^T v(k+1) \\ &= \begin{pmatrix} B_1 \otimes A_1^T + D_1 \otimes C_1^T + P^T(F_1 \otimes E_1^T) & B_2 \otimes A_2^T + D_2 \otimes C_2^T + P^T(F_2 \otimes E_2^T) \end{pmatrix} v(k+1). \end{aligned} \tag{2.2}$$

According to (2.2), let us define

$$x(k) = \text{vec}(X(k)), \quad s(k) = \text{vec}(S(k)), \quad z(k) = \text{vec}(Z(k)), \quad u(k) = \text{vec}(U(k)), \tag{2.3}$$

$$v(k) = \begin{pmatrix} \text{vec}(V_1(k)) \\ \text{vec}(V_2(k)) \end{pmatrix}, \quad r(k) = \begin{pmatrix} \text{vec}(R_1(k)) \\ \text{vec}(R_2(k)) \end{pmatrix}, \quad w(k) = \begin{pmatrix} \text{vec}(W_1(k)) \\ \text{vec}(W_2(k)) \end{pmatrix}, \tag{2.4}$$

where $X(k), S(k), Z(k), U(k) \in \mathrm{R}^{m\times n}$ and $R_i(k), V_i(k), W_i(k) \in \mathrm{R}^{p_i\times q_i}$ for $i = 1, 2$. Substituting (2.3) and (2.4) into (2.2) gives us

$$\begin{gathered} \text{vec}(R_i(1)) = \text{vec}(A_iX(1)B_i + C_iX(1)D_i + E_iX(1)^TF_i - M_i), \quad i = 1, 2, \\ \text{vec}(W_i(k+1)) = \text{vec}(A_iU(k+1)B_i + C_iU(k+1)D_i + E_iU(k+1)^TF_i), \quad i = 1, 2, \\ \text{vec}(Z(k+1)) = \text{vec}(A_1^TV_1(k+1)B_1^T + C_1^TV_1(k+1)D_1^T + F_1V_1(k+1)^TE_1 \\ +A_2^TV_2(k+1)B_2^T + C_2^TV_2(k+1)D_2^T + F_2V_2(k+1)^TE_2). \end{gathered}$$

Now by considering the above discussion, we establish the matrix form of BCR algorithm for solving (1.3) as follows.

Algorithm 1. (Matrix form of BCR algorithm to solve (1.3))

Step 1. *Given the initial matrix $X(1) \in R^{m\times n}$ and nonzero arbitrary matrix $S(1) \in R^{m\times n}$. Input the tolerance $\tau > 0$;*

Step 2. *Compute*

$$\begin{aligned} &R_i(1) = A_iX(1)B_i + C_iX(1)D_i + E_iX(1)^TF_i - M_i, \quad i = 1, 2, \\ &U(1) = S(1),\ V_1(1) = R_1(1),\ V_2(1) = R_2(1), \\ &W_i(1) = A_iU(1)B_i + C_iU(1)D_i + E_iU(1)^TF_i, \quad i = 1, 2, \\ &Z(1) = A_1^TV_1(1)B_1^T + C_1^TV_1(1)D_1^T + F_1V_1(1)^TE_1 + A_2^TV_2(1)B_2^T + C_2^TV_2(1)D_2^T + F_2V_2(1)^TE_2; \end{aligned}$$

Set $k = 1$;

Step 3. *Exit if the stopping criterion $\sqrt{\|R_1(k)\|^2 + \|R_2(k)\|^2} \le \tau$ has been met;*

Step 4. *Compute*

$$\alpha(k) = \frac{\mathrm{tr}(W_1(k)^T R_1(k)) + \mathrm{tr}(W_2(k)^T R_2(k))}{\mathrm{tr}(W_1(k)^T W_1(k)) + \mathrm{tr}(W_2(k)^T W_2(k))},$$
$$X(k+1) = X(k) - \alpha(k)U(k),$$
$$R_i(k+1) = R_i(k) - \alpha(k)W_i(k), \ \ i = 1,2,$$
$$\beta(k) = \frac{\mathrm{tr}(Z(k)^T S(k))}{\mathrm{tr}(Z(k)^T Z(k))},$$
$$S(k+1) = S(k) - \beta(k)Z(k),$$
$$\gamma(k) = \frac{\sum_{i=1}^{2} \mathrm{tr}(W_i(k)^T (A_i S(k+1) B_i + C_i S(k+1) D_i + E_i S(k+1)^T F_i))}{\mathrm{tr}(W_1(k)^T W_1(k)) + \mathrm{tr}(W_2(k)^T W_2(k))},$$
$$U(k+1) = S(k+1) - \gamma(k)U(k),$$
$$\eta(k) = \frac{1}{\mathrm{tr}(Z(k)^T Z(k))}[\mathrm{tr}(Z(k)^T (A_1^T R_1(k+1) B_1^T + C_1^T R_1(k+1) D_1^T + F_1 R_1(k+1)^T E_1$$
$$+ A_2^T R_2(k+1) B_2^T + C_2^T R_2(k+1) D_2^T + F_2 R_2(k+1)^T E_2)],$$
$$V_i(k+1) = R_i(k+1) - \eta(k)V_i(k), \ \ i = 1,2,$$
$$W_i(k+1) = A_i U(k+1) B_i + C_i U(k+1) D_i + E_i U(k+1)^T F_i$$
$$= A_i S(k+1) B_i + C_i S(k+1) D_i + E_i S(k+1)^T F_i - \gamma(k) W_i(k), \ \ i = 1,2,$$
$$Z(k+1) = A_1^T V_1(k+1) B_1^T + C_1^T V_1(k+1) D_1^T + F_1 V_1(k+1)^T E_1 + A_2^T V_2(k+1) B_2^T + C_2^T V_2(k+1) D_2^T + F_2 V_2(k+1)^T E_2$$
$$= A_1^T R_1(k+1) B_1^T + C_1^T R_1(k+1) D_1^T + F_1 R_1(k+1)^T E_1 + A_2^T R_2(k+1) B_2^T + C_2^T R_2(k+1) D_2^T + F_2 R_2(k+1)^T E_2 - \eta(k) Z(k);$$

Step 5. *Set $k = k+1$ and go to Step 3.*

In order to provide a convergence theorem of Algorithm 1, we first present the following lemma which gives some key properties of the algorithm.

Lemma 1. *Suppose that there exists a positive integer number r such that $\alpha(k) \neq 0$, $\alpha(k) \neq \infty$ and $[\|R_1(k)\| + \|R_2(k)\|] \neq 0$ for all $k = 1,2,...,r$. Then*

$$\mathrm{tr}(R_1(v)^T W_1(u)) + \mathrm{tr}(R_2(v)^T W_2(u)) = 0, \quad for \ \ u,v = 1,2,...,r, \quad v > u, \tag{2.5}$$
$$\mathrm{tr}(S(v)^T Z(u)) = 0, \quad for \ \ u,v = 1,2,...,r, \quad v > u, \tag{2.6}$$
$$\mathrm{tr}(Z(v)^T Z(u)) = 0, \quad for \ \ u,v = 1,2,...,r, \quad u \neq v, \tag{2.7}$$
$$\mathrm{tr}(W_1(v)^T W_1(u)) + \mathrm{tr}(W_2(v)^T W_2(u)) = 0, \quad for \ \ u,v = 1,2,...,r, \quad u \neq v. \tag{2.8}$$

Proof. It is described in the Appendix. □

Using the above lemma, we will be able to obtain the convergence theorem of Algorithm 1 as follows.

Theorem 1. *Let the coupled generalized Sylvester-transpose matrix equations (1.3) be consistent. Algorithm 1 can compute the solution of (1.3) within a finite number of iterations in the absence of round-off errors.*

Proof. Let us first consider the space $\mathrm{R}^{p_1 \times q_1} \times \mathrm{R}^{p_2 \times q_2}$ with a inner product defined as follows

$$\langle (M_1, M_2), (N_1, N_2) \rangle = \mathrm{tr}(M_1^T N_1) + \mathrm{tr}(M_2^T N_2) \ \ M_1, N_1 \in \mathrm{R}^{p_1 \times q_1}, \quad M_2, N_2 \in \mathrm{R}^{p_2 \times q_2}.$$

If $W_1(k) \neq 0$ or $W_2(k) \neq 0$ for $k = 1,2,...,p_1q_1 + p_2q_2$ then from Lemma 1 we deduce that $(W_1(k), W_2(k))$, $k = 1,2,...,p_1q_1 + p_2q_2$ is an orthogonal basis of the inner product space $\mathrm{R}^{p_1 \times q_1} \times \mathrm{R}^{p_2 \times q_2}$. From this result and (2.5), we infer that Algorithm 1 can obtain the solution of (1.3) in a finite number of iterations in the absence of round-off errors. □

In what follows, we show that Algorithm 1 with the special initial matrix can obtain the least Frobenius norm solution of (1.3).

Theorem 2. *Let the coupled generalized Sylvester-transpose matrix equations (1.3) be consistent. If we choose the initial matrix*

$$X(1) = A_1^T M_1(1) B_1^T + C_1^T M_1(1) D_1^T + F_1 M_1(1)^T E_1 + A_2^T M_2(1) B_2^T + C_2^T M_2(1) D_2^T + F_2 M_2(1)^T E_2, \tag{2.9}$$

and matrix

$$S(1) = A_1^T N_1(1) B_1^T + C_1^T N_1(1) D_1^T + F_1 N_1(1)^T E_1 + A_2^T N_2(1) B_2^T + C_2^T N_2(1) D_2^T + F_2 N_2(1)^T E_2, \tag{2.10}$$

where $M_1(1), N_1(1) \in R^{p_1 \times q_1}$ and $M_2(1), N_2(1) \in R^{p_2 \times q_2}$ are arbitrary, then the solution X^* generated by Algorithm 1 is the least Frobenius norm solution of (1.3).

Proof. From (2.9) and (2.10) together with Algorithm 1 it follows that there exist the matrices $M_1(k), N_1(k) \in R^{p_1 \times q_1}$ and $M_2(k), N_2(k) \in R^{p_2 \times q_2}$ such that

$$X(k) = A_1^T M_1(k) B_1^T + C_1^T M_1(k) D_1^T + F_1 M_1(k)^T E_1 + A_2^T M_2(k) B_2^T + C_2^T M_2(k) D_2^T + F_2 M_2(k)^T E_2, \tag{2.11}$$

and

$$S(k) = A_1^T N_1(k) B_1^T + C_1^T N_1(k) D_1^T + F_1 N_1(k)^T E_1 + A_2^T N_2(k) B_2^T + C_2^T N_2(k) D_2^T + F_2 N_2(k)^T E_2. \tag{2.12}$$

By using (2.11) and (2.12), it is clear that

$$\begin{aligned}
\mathrm{vec}(X(k)) =& \mathrm{vec}(A_1^T M_1(k) B_1^T + C_1^T M_1(k) D_1^T + F_1 M_1(k)^T E_1 + A_2^T M_2(k) B_2^T + C_2^T M_2(k) D_2^T + F_2 M_2(k)^T E_2) \\
=& \begin{pmatrix} B_1 \otimes A_1^T + D_1 \otimes C_1^T + P^T(F_1 \otimes E_1^T) & B_2 \otimes A_2^T + D_2 \otimes C_2^T + P^T(F_2 \otimes E_2^T) \end{pmatrix} \begin{pmatrix} \mathrm{vec}(M_1(k)) \\ \mathrm{vec}(M_2(k)) \end{pmatrix} \\
=& \begin{pmatrix} B_1^T \otimes A_1 + D_1^T \otimes C_1 + (F_1^T \otimes E_1)P \\ B_2^T \otimes A_2 + D_2^T \otimes C_2 + (F_2^T \otimes E_2)P \end{pmatrix}^T \begin{pmatrix} \mathrm{vec}(M_1(k)) \\ \mathrm{vec}(M_2(k)) \end{pmatrix} \\
\in& \mathcal{R}\left(\begin{pmatrix} B_1^T \otimes A_1 + D_1^T \otimes C_1 + (F_1^T \otimes E_1)P \\ B_2^T \otimes A_2 + D_2^T \otimes C_2 + (F_2^T \otimes E_2)P \end{pmatrix}^T \right).
\end{aligned}$$

Therefore, it comes that the solution X^* generated by Algorithm 1 is the least Frobenius norm solution of (1.3). □

Remark 1. *It holds*

$$\begin{aligned}
\|R_1(k+1)\|^2 + \|R_2(k+1)\|^2 =& \mathrm{tr}((R_1(k) - \alpha(k)W_1(k))^T(R_1(k) - \alpha(k)W_1(k))) \\
&+ \mathrm{tr}((R_2(k) - \alpha(k)W_2(k))^T(R_2(k) - \alpha(k)W_2(k))) \\
=& \|R_1(k)\|^2 + \|R_2(k)\|^2 + \alpha(k)^2(\|W_1(k)\|^2 + \|W_2(k)\|^2) \\
&- 2\alpha(k)[\mathrm{tr}(W_1(k)^T R_1(k)) + \mathrm{tr}(W_2(k)^T R_2(k))] \\
=& \|R_1(k)\|^2 + \|R_2(k)\|^2 - \alpha(k)[\mathrm{tr}(W_1(k)^T R_1(k)) + \mathrm{tr}(W_2(k)^T R_2(k))] \\
=& \|R_1(k)\|^2 + \|R_2(k)\|^2 - \frac{[\mathrm{tr}(W_1(k)^T R_1(k)) + \mathrm{tr}(W_2(k)^T R_2(k))]^2}{\|W_1(k)\|^2 + \|W_2(k)\|^2} \\
\leq& \|R_1(k)\|^2 + \|R_2(k)\|^2.
\end{aligned}$$

This shows that Algorithm 1 guarantees the descent of the residual norms.

3. Numerical Experiments

This section presents two numerical examples to further illustrate the numerical effectiveness of BCR algorithm compared with the CGNR and CGNE methods. All computations are carried out in MATLAB with double machine precision.

Example 1. *As the first example, we consider the Sylvester-transpose matrix equation* $AXB + CX^TD = M$ *with the following parameters*

$$A = triu(rand(n,n),1) + diag(2 + diag(rand(n))), \quad B = triu(rand(n,n),1) + diag(2 + diag(rand(n))),$$

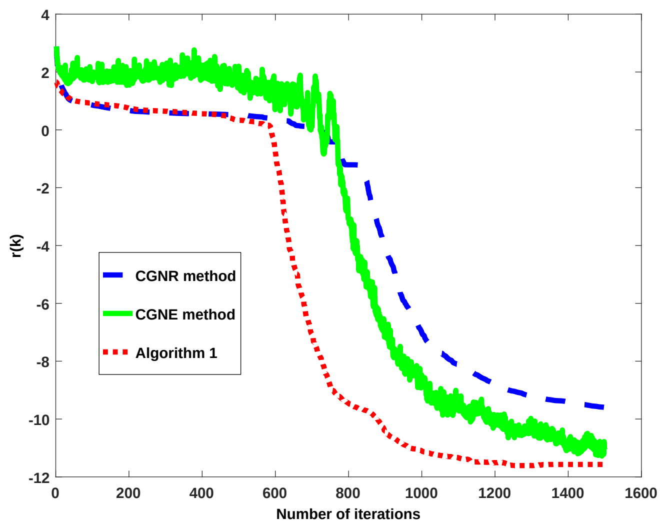
$$C = tril(rand(n,n),1) + diag(1.5 + diag(rand(n))), \quad D = triu(rand(n,n),1) + diag(1.5 + diag(rand(n))),$$

$$M = 10 * rand(n).$$

For $n = 15$, *we apply CGNR and CGNE methods and Algorithm 1 with the initial matrix* $X(0)$ *to solve this matrix equation. The obtained results are presented in Figure 1 where*

$$r(k) = \log_{10} \|M - AX(k)B - CX(k)^TD\|.$$

Figure 1: Numerical comparison of the testing methods for Example 1.



Example 2. *In this example, we study the generalized Sylvester-transpose matrix equation* $AXB+CXD+EX^TF = M$ *where*

$$A = \begin{pmatrix} 4 & 40 & 4 & 7 & 9 & 1 & 0 & 10 \\ 4 & 400 & -99 & -2 & -2 & 2 & 3 & 4 \\ -2 & -2 & 100 & 5 & 600 & -1 & -5 & 5 \\ 100 & 2 & -2 & -2 & 5 & 1 & 200 & 2 \\ -90 & -9 & 10 & 5 & 200 & 3 & 1 & 3 \\ 10 & -20 & -1 & 50 & 4 & 5 & 3 & 10 \\ 20 & 3 & 900 & 6 & 3 & 5 & 9 & 4 \\ 20 & 3 & 233 & 6 & 3 & 5 & 9 & 4 \end{pmatrix}, \quad B = \begin{pmatrix} 10 & -22 & 3 & 7 & 110 & -1 & 6 & 10 \\ 40 & -5 & 1 & -12 & 5 & 5 & 6 & 4 \\ -2 & 5 & 1 & 10 & 6 & -2 & 12 & 2 \\ 10 & 5 & -5 & -2 & 5 & -3 & 25 & 12 \\ 1 & -800 & 2 & 2 & 3 & 5 & 7 & 44 \\ 20 & -10 & -100 & 5 & 3 & 2 & 11 & 77 \\ 30 & 6 & 200 & 4 & 2 & 200 & 8 & 77 \\ 300 & 6 & 2 & 4 & 2 & 200 & 8 & 7700 \end{pmatrix},$$

$$C = \begin{pmatrix} -32 & 168 & -4 & -14 & -422 & 6 & -24 & -20 \\ -152 & 820 & -202 & 44 & -24 & -16 & -18 & -8 \\ 4 & -24 & 196 & -30 & 1176 & 6 & -58 & 2 \\ 160 & -16 & 16 & 4 & -10 & 14 & 300 & -44 \\ -184 & 3182 & 12 & 2 & 388 & -14 & -26 & -170 \\ -60 & 0 & 398 & 80 & -4 & 2 & -38 & -288 \\ -80 & -18 & 1000 & -4 & -2 & -790 & -14 & -300 \\ -1160 & -18 & 458 & -4 & -2 & -790 & -14 & -30792 \end{pmatrix},$$

$$D = \begin{pmatrix} -40 & -156 & -26 & -49 & -265 & -3 & -12 & -70 \\ -100 & -1990 & 493 & 34 & 0 & -20 & -27 & -28 \\ 14 & 0 & -502 & -45 & -3012 & 9 & 1 & -29 \\ -520 & -20 & 20 & 14 & -35 & 1 & -1050 & -34 \\ 448 & 1645 & -54 & -29 & -1006 & -25 & -19 & -103 \\ -90 & 120 & 205 & -260 & -26 & -29 & -37 & -204 \\ -160 & -27 & -4900 & -38 & -19 & -425 & -61 & -174 \\ -700 & -27 & -1169 & -38 & -19 & -425 & -61 & -15420 \end{pmatrix},$$

$$E = \begin{pmatrix} -28 & 208 & 0 & -7 & -413 & 7 & -24 & -10 \\ -148 & 1220 & -301 & 42 & -26 & -14 & -15 & -4 \\ 2 & -26 & 296 & -25 & 1776 & 5 & -63 & 7 \\ 260 & -14 & 14 & 2 & -5 & 15 & 500 & -42 \\ -274 & 3173 & 22 & 7 & 588 & -11 & -25 & -167 \\ -50 & -20 & 397 & 130 & 0 & 7 & -35 & -278 \\ -60 & -15 & 1900 & 2 & 1 & -785 & -5 & -296 \\ -1140 & -15 & 691 & 2 & 1 & -785 & -5 & -30788 \end{pmatrix},$$

$$F = \begin{pmatrix} -30 & -178 & -23 & -42 & -155 & -4 & -6 & -60 \\ -60 & -1995 & 494 & 22 & 5 & -15 & -21 & -24 \\ 12 & 5 & -501 & -35 & -3006 & 7 & 13 & -27 \\ -510 & -15 & 15 & 12 & -30 & -2 & -1025 & -22 \\ 449 & 845 & -52 & -27 & -1003 & -20 & -12 & -59 \\ -70 & 110 & 105 & -255 & -23 & -27 & -26 & -127 \\ -130 & -21 & -4700 & -34 & -17 & -225 & -53 & -97 \\ -400 & -21 & -1167 & -34 & -17 & -225 & -53 & -7720 \end{pmatrix}, \quad M = rand(8).$$

This generalized Sylvester-transpose matrix equation can be transformed into the following linear system

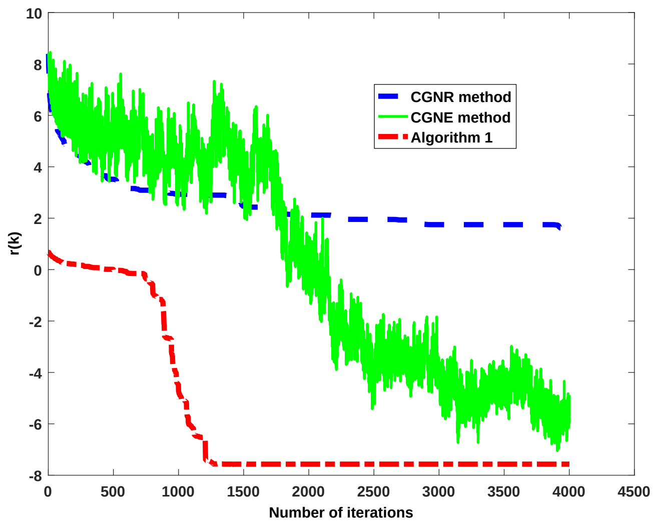
$$\underbrace{\left(B^T \otimes A + D^T \otimes C + (F^T \otimes E)P\right)}_{T} \underbrace{\mathrm{vec}(X)}_{x} = \underbrace{\mathrm{vec}(M)}_{d}.$$

Since the condition number $\kappa(T) = \|T\|\|T^{\dagger}\| = 1.8131 \times 10^6$, we see that the above system is ill-conditioned. By using the mentioned methods with $X(0)$, we obtain the sequence $\{X(k)\}$. In Figure 2, we depict the obtained results where

$$r(k) = \log_{10} \|M - AX(k)B - CX(k)D - EX(k)^T F\|.$$

The above both examples show that Algorithm 1 can solve the matrix equations efficiently.

Figure 2: Numerical comparison of the testing methods for Example 2.



4. Conclusions

In this paper, we have introduced the matrix form of BCR algorithm for solving the coupled generalized Sylvester-transpose matrix equations (1.3). We have shown that the introduced algorithm can find the least Frobenius norm solution of (1.3) within a finite number of iterations in the absence of round-off errors. It has been demonstrated by numerical examples that the introduced algorithm is better than some existing methods to solve linear matrix equation.

Acknowledgments

I would like to thank Professor Dijana Mosić (Editor) and four anonymous referees for carefully reading the manuscript and offering thoughtful comments.

Appendix

The proof of Lemma 1

We apply induction to prove (2.5)-(2.8). Since the inner product is commutative, we only need to demonstrate (2.5)-(2.8) for $1 \leq u < v \leq r$. When $v = 2$ and $u = 1$, we have

$$\mathrm{tr}(R_1(2)^T W_1(1)) + \mathrm{tr}(R_2(2)^T W_2(1))$$

$$= \mathrm{tr}(R_1(1)^T W_1(1)) + \mathrm{tr}(R_2(1)^T W_2(1)) - \mathrm{tr}(W_1(1)^T R_1(1)) - \mathrm{tr}(W_2(1)^T R_2(1)) = 0,$$

$$\mathrm{tr}(S(2)^T Z(1)) = \mathrm{tr}(S(1)^T Z(1)) - \mathrm{tr}(Z(1)^T S(1)) = 0,$$

$$\mathrm{tr}(Z(2)^T Z(1))$$

$$= \mathrm{tr}((A_1^T R_1(1) B_1^T + C_1^T R_1(1) D_1^T + F_1 R_1(1)^T E_1 + A_2^T R_2(1) B_2^T + C_2^T R_2(1) D_2^T + F_2 R_2(1)^T E_2 - \eta(1) Z(1))^T Z(1))$$

$$= \mathrm{tr}((A_1^T R_1(2) B_1^T + C_1^T R_1(2) D_1^T + F_1 R_1(2)^T E_1 + A_2^T R_2(2) B_2^T + C_2^T R_2(2) D_2^T + F_2 R_2(2)^T E_2)^T Z(1))$$

$$- \mathrm{tr}(Z(1)^T (A_1^T R_1(2) B_1^T + C_1^T R_1(2) D_1^T + F_1 R_1(2)^T E_1 + A_2^T R_2(2) B_2^T + C_2^T R_2(2) D_2^T + F_2 R_2(2)^T E_2)) = 0,$$

$$\mathrm{tr}(W_1(2)^T W_1(1)) + \mathrm{tr}(W_2(2)^T W_2(1))$$

$$= \mathrm{tr}((A_1S(2)B_1 + C_1S(2)D_1 + E_1S(2)^TF_1 - \gamma(1)W_1(1))^TW_1(1))$$
$$+\mathrm{tr}((A_2S(2)B_2 + C_2S(2)D_2 + E_2S(2)^TF_2 - \gamma(1)W_2(1))^TW_2(1))$$
$$= \mathrm{tr}((A_1S(2)B_1 + C_1S(2)D_1 + E_1S(2)^TF_1)^TW_1(1)) + \mathrm{tr}((A_2S(2)B_2 + C_2S(2)D_2 + E_2S(2)^TF_2)^TW_2(1))$$
$$-\sum_{i=1}^{2} \mathrm{tr}(W_i(1)^T(A_iS(2)B_i + C_iS(2)D_i + E_iS(2)^TF_i)) = 0.$$

These imply that (2.5)-(2.8) hold for $v = 2$ and $u = 1$. Now for $u < w < r$, we assume that

$$\mathrm{tr}(R_1(w)^TW_1(u)) + \mathrm{tr}(R_2(w)^TW_2(u)) = 0, \quad \mathrm{tr}(S(w)^TZ(u)) = 0, \tag{4.1}$$

$$\mathrm{tr}(Z(w)^TZ(u)) = 0, \ \ \mathrm{tr}(W_1(w)^TW_1(u)) + \mathrm{tr}(W_2(w)^TW_2(u)) = 0. \tag{4.2}$$

By making use of (4.1) and (4.2), we can obtain

$$\mathrm{tr}(R_1(w+1)^TW_1(u)) + \mathrm{tr}(R_2(w+1)^TW_2(u))$$
$$= \mathrm{tr}((R_1(w) - \alpha(w)W_1(w))^TW_1(u)) + \mathrm{tr}((R_2(w) - \alpha(w)W_2(w))^TW_2(u)) = 0,$$
$$\mathrm{tr}(S(w+1)^TZ(u)) = \mathrm{tr}((S(w) - \beta(w)Z(w))^TZ(u)) = 0,$$
$$\mathrm{tr}(Z(w+1)^TZ(u))$$
$$= \mathrm{tr}((A_1^TR_1(w+1)B_1^T + C_1^TR_1(w+1)D_1^T + F_1R_1(w+1)^TE_1 + A_2^TR_2(w+1)B_2^T$$
$$+C_2^TR_2(w+1)D_2^T + F_2R_2(w+1)^TE_2 - \eta(w)Z(w))^TZ(u))$$
$$= \frac{1}{\beta(u)}[\mathrm{tr}(R_1(w+1)^T(A_1(S(u) - S(u+1))B_1 + C_1(S(u) - S(u+1))D_1 + E_1(S(u) - S(u+1))^TF_1))$$
$$+\mathrm{tr}(R_2(w+1)^T(A_2(S(u) - S(u+1))B_2 + C_2(S(u) - S(u+1))D_2 + E_2(S(u) - S(u+1))^TF_2))]$$
$$= \frac{1}{\beta(u)}[\mathrm{tr}(R_1(w+1)^T(W_1(u) + \gamma(u-1)W_1(u-1))) - \mathrm{tr}(R_1(w+1)^T(W_1(u+1) + \gamma(u)W_1(u)))$$
$$+\mathrm{tr}(R_2(w+1)^T(W_2(u) + \gamma(u-1)W_2(u-1))) - \mathrm{tr}(R_2(w+1)^T(W_2(u+1) + \gamma(u)W_2(u)))]$$
$$= -\frac{1}{\beta(u)}[\mathrm{tr}(R_1(w+1)^TW_1(u+1)) + \mathrm{tr}(R_2(w+1)^TW_2(u+1))], \tag{4.3}$$

$$\mathrm{tr}(W_1(w+1)^TW_1(u)) + \mathrm{tr}(W_2(w+1)^TW_2(u))$$
$$= \mathrm{tr}((A_1S(w+1)B_1 + C_1S(w+1)D_1 + E_1S(w+1)^TF_1 - \gamma(w)W_1(w))^TW_1(u))$$
$$+\mathrm{tr}((A_2S(w+1)B_2 + C_2S(w+1)D_2 + E_2S(w+1)^TF_2 - \gamma(w)W_2(w))^TW_2(u))$$
$$= \mathrm{tr}(S(w+1)^T(A_1^TW_1(u)B_1^T + C_1^TW_1(u)D_1^T + F_1W_1(u)^TE_1 + A_2^TW_2(u)B_2^T + C_2^TW_2(u)D_2^T + F_2W_2(u)^TE_2))$$
$$= \frac{1}{\alpha(u)}[\mathrm{tr}(S(w+1)^T(A_1^T(R_1(u) - R_1(u+1))B_1^T + C_1^T(R_1(u) - R_1(u+1))D_1^T + F_1(R_1(u) - R_1(u+1))^TE_1$$
$$+A_2^T(R_2(u) - R_2(u+1))B_2^T + C_2^T(R_2(u) - R_2(u+1))D_2^T + F_2(R_2(u) - R_2(u+1))^TE_2))]$$
$$= \frac{1}{\alpha(u)}[\mathrm{tr}(S(w+1)^T(Z(u) + \eta(u-1)Z(u-1)) - Z(u+1) - \eta(u)Z(u)))]$$
$$= -\frac{1}{\alpha(u)}[\mathrm{tr}(S(w+1)^TZ(u+1))]. \tag{4.4}$$

Also for $u = w$, we can get

$$\mathrm{tr}(R_1(w+1)^TW_1(w)) + \mathrm{tr}(R_2(w+1)^TW_2(w)) = 0,$$

$$\begin{aligned}
&\mathrm{tr}(S(w+1)^T Z(w)) = 0,\\
&\mathrm{tr}(Z(w+1)^T Z(w)) = \mathrm{tr}((A_1^T R_1(w+1)B_1^T + C_1^T R_1(w+1)D_1^T + F_1 R_1(w+1)^T E_1\\
&+A_2^T R_2(w+1)B_2^T + C_2^T R_2(w+1)D_2^T + F_2 R_2(w+1)^T E_2 - \eta(w)Z(w))^T Z_2(w))\\
&= \mathrm{tr}((A_1^T R_1(w+1)B_1^T + C_1^T R_1(w+1)D_1^T + F_1 R_1(w+1)^T E_1\\
&+A_2^T R_2(w+1)B_2^T + C_2^T R_2(w+1)D_2^T + F_2 R_2(w+1)^T E_2)^T Z(w))\\
&-\mathrm{tr}(Z(w)^T(A_1^T R_1(w+1)B_1^T + C_1^T R_1(w+1)D_1^T + F_1 R_1(w+1)^T E_1\\
&+A_2^T R_2(w+1)B_2^T + C_2^T R_2(w+1)D_2^T + F_2 R_2(w+1)^T E_2) = 0,\\
&\mathrm{tr}(W_1(w+1)^T W_1(1)) + \mathrm{tr}(W_2(w+1)^T W_2(1))\\
&= \mathrm{tr}((A_1 S(w+1)B_1 + C_1 S(w+1)D_1 + E_1 S(w+1)^T F_1 - \gamma(w)W_1(w))^T W_1(w))\\
&+\mathrm{tr}((A_2 S(w+1)B_2 + C_2 S(w+1)D_2 + E_2 S(w+1)^T F_2 - \gamma(w)W_2(w))^T W_2(w))\\
&= \mathrm{tr}((A_1 S(w+1)B_1 + C_1 S(w+1)D_1 + E_1 S(w+1)^T F_1)^T W_1(w))\\
&+\mathrm{tr}((A_2 S(w+1)B_2 + C_2 S(w+1)D_2 + E_2 S(w+1)^T F_2)^T W_2(w))\\
&-\sum_{i=1}^{2} \mathrm{tr}(W_i(w)^T(A_i S(w+1)B_i + C_i S(w+1)D_i + E_i S(w+1)^T F_i)) = 0.
\end{aligned}$$

From

$$\mathrm{tr}(Z(w)^T Z(u)) = 0, \quad \mathrm{tr}(R_1(w+1)^T W_1(w)) + \mathrm{tr}(R_2(w+1)^T W_2(w)) = 0,$$

and (4.3), it could be concluded that

$$\mathrm{tr}(Z(w+1)^T Z(u)) = 0.$$

Also from

$$\mathrm{tr}(W_1(w)^T W_1(u)) + \mathrm{tr}(W_2(w)^T W_2(u)) = 0, \quad \mathrm{tr}(S(w+1)^T Z(w)) = 0,$$

and (4.4), one can easily verify

$$\mathrm{tr}(W_1(w+1)^T W_1(u)) + \mathrm{tr}(W_2(w+1)^T W_2(u)) = 0.$$

Hence Lemma 1 holds by the principle of induction.

References

[1] F.M. Dopico, J. González, D. Kressner, V. Simoncini, Projection methods for large-scale T-Sylvester equations, Mathematics of Computation, 85 (2016) 2427-2455.

[2] S.D. Shank, V. Simoncini, D.B. Szyld, Efficient low-rank solution of generalized Lyapunov equations, Numerische Mathematik, 134 (2016) 327-342.

[3] V. Simoncini, Computational methods for linear matrix equations, SIAM Review, 58 (2016) 377-441.

[4] A.C. Antoulas, Approximation of Large-Scale Dynamical Systems, Adv. Des. Control 6,. SIAM, Philadelphia, 2005.

[5] H.C. Elman, M. Wu, Lyapunov inverse iteration for computing a few rightmost eigenvalues of large generalized eigenvalue problems, SIAM Journal on Matrix Analysis and Applications, 34 (2013) 1685-1707.

[6] A.T. Chronopoulos, S-step iterative methods for (non)symmetric (in)definite linear systems, SIAM Journal on Numerical Analysis, 28 (1991) 1776-1789.

[7] A.T. Chronopoulos, On the squared unsymmetric lanczos method, Journal of Computational and Applied Mathematics 54 (1994) 65-78.

[8] A.T. Chronopoulos, D. Kincaid, On the odir iterative method for non-symmetric indefinite linear systems, Numerical Linear Algebra with Applications, 8 (2001) 71-82.

[9] A.T. Chronopoulos, A.B. Kucherov, Block s-step krylov iterative methods, Numerical Linear Algebra with Applications, 17 (2010) 3-15.

[10] A.T. Chronopoulos, C.D. Swanson, Parallel iterative s-step methods for unsymmetric linear systems, Parallel Computing, 22 (1996) 623-641.

[11] M. Hajarian, Developing CGNE algorithm for the periodic discrete-time generalized coupled Sylvester matrix equations, Computational & Applied Mathematics, 34 (2015) 755-771.

[12] M. Hajarian, Solving the general coupled and the periodic coupled matrix equations via the extended QMRCGSTAB algorithms, Computational & Applied Mathematics, 33 (2014) 349-362.
[13] L. Dai Singular Control Systems, Berlin: Springer-Vertag, 1989.
[14] M. Dehghan, M. Hajarian, Construction of an iterative method for solving generalized coupled Sylvester matrix equations, Transactions of the Institute of Measurement and Control, 35 (2013) 961-970.
[15] F. Ding, T. Chen, Iterative least squares solutions of coupled Sylvester matrix equations, Systems & Control Letters, 54, (2005) 95-107.
[16] F. Ding, T. Chen, Gradient based iterative algorithms for solving a class of matrix equations, IEEE Transactions on Automatic Control, 50 (2005) 1216-1221.
[17] F. Ding, T. Chen, On iterative solutions of general coupled matrix equations, SIAM Journal on Control and Optimization, 44 (2006) 2269-2284.
[18] G.R. Duan, The solution to the matrix equation $AV + BW = EVJ + R$, Applied Mathematics Letters, 17 (2004) 1197-1202.
[19] L.R. Fletcher, J. Kuatsky, N.K. Nichols, Eigenstructure assignment in descriptor systems, IEEE Transactions on Automatic Control, 31 (1986) 1138-1141.
[20] P.M. Frank, Fault diagnosis in dynamic systems using analytical and knowledge-based redundancy: A survey and some new results, Automatica, 26 (1990) 459-474.
[21] M. Hajarian, Developing Bi-CG and Bi-CR methods to solve generalized Sylvester-transpose matrix equations, International Journal of Automation and Computing, 11 (2014) 25-29.
[22] M. Hajarian, Developing the CGLS algorithm for the least squares solutions of the general coupled matrix equations, Mathematical Methods in the Applied Sciences, 37 (2014) 2782-2798.
[23] G.X. Huang, N. Wu, F. Yin, Z.L. Zhou, K. Guo, Finite iterative algorithms for solving generalized coupled Sylvester systems-Part I: One-sided and generalized coupled Sylvester matrix equations over generalized reflexive solutions, Applied Mathematical Modelling, 36 (2012) 1589-1603.
[24] F. Yin, G.X. Huang, D.Q. Chen, Finite iterative algorithms for solving generalized coupled Sylvester systems-Part II: Two-sided and generalized coupled Sylvester matrix equations over reflexive solutions Applied Mathematical Modelling, 36 (2012) 1604-1614.
[25] E.I. Jury, Inners and Stability of Dynamic Systems. John Wiley & Sons, New York, 1974.
[26] D.L. Kleinman, On an iterative technique for Riccati equation computations, IEEE Transactions on Automatic Control, 13 (1968) 114-115.
[27] I. Kyrchei, Explicit representation formulas for the minimum norm least squares solutions of some quaternion matrix equations, Linear Algebra and its Applications, 438 (2013) 136-152.
[28] I. Kyrchei, Explicit formulas for determinantal representations of the Drazin inverse solutions of some matrix and differential matrix equations, Applied Mathematics and Computation, 219 (2013) 7632-7644.
[29] W.S. Levine, M. Athans, On the determination of the optimal constant output feedback gains for linear multivariable systems, IEEE Transactions on Automatic Control, 15 (1970) 44-48.
[30] S.K. Li, T.Z. Huang, LSQR iterative method for generalized coupled Sylvester matrix equations, Applied Mathematical Modelling, 36 (2012) 3545-3554.
[31] Z.Y. Li, Y. Wang, Weighted steepest descent method for solving matrix equations, International Journal of Computer Mathematics, 89 (2012) 1017-1038.
[32] F. Piao, Q. Zhang, Z. Wang, The solution to matrix equation $AX + X^TC = B$, Journal of the Franklin Institute, 344 (2007) 1056-1062.
[33] C. Song, G. Chen, L. Zhao, Iterative solutions to coupled Sylvester-transpose matrix equations, Applied Mathematical Modelling, 35 (2011) 4675-4683.
[34] M.T. Vespucci, C.G. Broyden, Implementation of different computational variations of biconjugate residual methods, Computers & Mathematics with Applications, 42 (2001) 1239-1253.
[35] Q.W. Wang, Z.H. He, Solvability conditions and general solution for mixed Sylvester equations, Automatica, 49 (2013) 2713-2719.
[36] B. Yan, S.X.D. Tan P. Liu, B. McGaughy, Passive interconnect macromodeling via balanced truncation of linear systems in descriptor form, Design Automation Conference, Asia and South Pacific, 23-26 January, pp. 355-360, 2007.
[37] S. Yuan, A. Liao, Least squares Hermitian solution of the complex matrix equation $AXB + CXD = E$ with the least norm, Journal of the Franklin Institute, 351 (2014) 4978-4997.
[38] L.Q. Zhang, J. Lam, Q.L. Zhang, Lyapunov and Riccati equations of discrete-time descriptor systems, IEEE Transactions on Automatic Control, 44 (1999) 2134-2139.
[39] B. Zhou, Z. Lin, Parametric Lyapunov equation approach to stabilization of discrete-time systems with input delay and saturation, IEEE Transactions on Circuits and Systems I: Regular Papers, 58 (2011) 2741-2754.
[40] B. Zhou, Z. Lin, G. Duan, A parametric Lyapunov equation approach to low gain feedback design for discrete-time systems, Automatica, 45 (2009) 238-244.
[41] B. Zhou, J. Lam, G.R. Duan, Toward solution of matrix equation $X = Af(X)B + C$, Linear Algebra and its Applications, 435 (2011) 1370-1398.
[42] S. Şimşek, M. Sarduvan, H. Özdemir, Centrohermitian and skew-centrohermitian solutions to the minimum residual and matrix nearness problems of the quaternion matrix equation $(AXB, DXE) = (C, F)$, Advances in Applied Clifford Algebras, 27 (2017) 2201-2214.
[43] C.G. Broyden, M.T. Vespucci, Krylov Solvers for Linear Algebraic Systems, ELSEVIER Inc., San Diego, 2004.
[44] Y. Saad, Iterative Methods for Sparse Linear Systems, SIAM, Philadelphia,. 2nd. ed., 2003.
[45] M. Hajarian, Developing BiCOR and CORS methods for coupled Sylvester-transpose and periodic Sylvester matrix equations, Applied Mathematical Modelling, 39 (2015) 6073-6084.
[46] M. Hajarian, Matrix GPBiCG algorithms for solving the general coupled matrix equations, IET Control Theory & Applications,

9 (2015) 74-81.
[47] M. Hajarian, Matrix iterative methods for solving the Sylvester-transpose and periodic Sylvester matrix equations, Journal of the Franklin Institute, 350 (2013) 3328-3341.