International Journal of Automation and Computing 11(1), February 2014, 25-29
DOI: 10.1007/s11633-014-0762-0

Developing Bi-CG and Bi-CR Methods to Solve Generalized Sylvester-transpose Matrix Equations

Masoud Hajarian
Department of Mathematics, Faculty of Mathematical Sciences, Shahid Beheshti University, General Campus, Evin, Tehran 19839, Iran

Abstract: The bi-conjugate gradients (Bi-CG) and bi-conjugate residual (Bi-CR) methods are powerful tools for solving nonsymmetric linear systems $Ax = b$. By using Kronecker product and vectorization operator, this paper develops the Bi-CG and Bi-CR methods for the solution of the generalized Sylvester-transpose matrix equation $\sum_{i=1}^{p}(A_iXB_i + C_iX^{\mathrm{T}}D_i) = E$ (including Lyapunov, Sylvester and Sylvester-transpose matrix equations as special cases). Numerical results validate that the proposed algorithms are much more efficient than some existing algorithms.

Keywords: Linear systems, iterative method, bi-conjugate gradients (Bi-CG) method, bi-conjugate residual (Bi-CR) method, Sylvester matrix equation.

1 Introduction

This paper considers the generalized Sylvester-transpose matrix equation

$$\sum_{i=1}^{p}(A_iXB_i + C_iX^{\mathrm{T}}D_i) = E \tag{1}$$

where $A_i, B_i, C_i, D_i, E \in \mathbf{R}^{m\times m}$ $(i = 1, 2, \cdots, p)$ are given matrices and matrix $X \in \mathbf{R}^{m\times m}$ needs to be determined. The generalized Sylvester-transpose matrix equation (1) is quite general and includes as particular cases several classical and important linear problems in the space of matrices: block linear systems, commuting matrices, the Lyapunov matrix equation:

$$D^{\mathrm{T}}X + XD = E. \tag{2}$$

The Sylvester matrix equation:

$$AX + XD = E. \tag{3}$$

And the Sylvester-transpose matrix equation is

$$AX + X^{\mathrm{T}}B = C. \tag{4}$$

It is well-known that the Lyapunov, Sylvester and Sylvester-transpose matrix equations are important equations which play a fundamental role in the various fields of engineering theory, particularly in theories and applications of stability and control[1−4]. The Sylvester and Lyapunov matrix equations arise in stability analysis of linear systems[5], model reduction[6] and in the solution of the algebraic Riccati matrix equation[7]. The solution of linear matrix equations such as Sylvester and Lyapunov matrix equations has been addressed in a large body of literature[8−13].

In [14], a Hessenberg method was proposed for solving the Sylvester matrix equation $XA+BX = C$. A new Smith accelerative iteration (containing the well-known Smith accelerative iteration as a special case) was established in [2] for solving the Stein matrix equation $X = AXB + C$. In [15], by using the so-called Kronecker matrix polynomials, closed form solutions to a family of generalized Sylvester matrix equation were given. Li and Huang[16] proposed an iterative method to solve generalized coupled Sylvester matrix equations, based on a matrix form of the least-squares QR-factorization (LSQR) algorithm. In [17, 18], Ding and Chen presented a gradient based method and a least-squares based iterative method for generalized Sylvester matrix equations and general coupled matrix equations by introducing the star ($\star$) product of matrices. The gradient iterative (GI) algorithms of solving general matrix equations were studied in [19] by using the hierarchical identification principle[20]. Recently by extending the conjugate gradient (CG) approach, Dehghan and Hajarian proposed efficient iterative algorithms to find the generalized bisymmetric, skew-symmetric, (generalized) reflexive and anti-reflexive solutions of linear matrix equations[21−23]. Solvability, existence of unique solution, closed-form solution and numerical solution of matrix equation $X = Af(X)B+C$ were studied in [24], where $f(X) = X^{\mathrm{T}}$, $f(X) = \overline{X}^{\mathrm{T}}$ and $f(X) = \overline{X}$. By Moore-Penrose generalized inverse, Piao et al.[25] obtained some necessary and sufficient condition for the existence of the solution and the expressions of the Sylvester-transpose matrix equation (4). Zhou et al.[26, 27] established the solution of the several generalized Sylvester matrix equations. Zhou et al.[28] proposed an iterative method for finding weighted least squares solutions to coupled Sylvester matrix equations. The Bi-CG and Bi-CR methods are two of the most important and useful algorithms for the numerical solutions of linear equations. The purpose of this paper is to develop the Bi-CG and Bi-CR methods for solving the generalized Sylvester-transpose matrix equation (1) via Kronecker product and vectorization operator.

The paper is organized as follows. In Section 2, the Bi-CG and Bi-CR methods are briefly described. By applying Kronecker product and vectorization operator, we derive new algorithms for solving (1) based on the Bi-CG and Bi-CR methods in Section 3. In Section 4, numerical results demonstrate that new algorithms are more efficient than some existing algorithms. Conclusions will be drawn

Manuscript received Feburary 4, 2013; revised April 16, 2013

in Section 5.

Throughout the paper, we denote the n-vector space by $\mathbf{R}^n$, and the set of $m \times n$ matrices by $\mathbf{R}^{m\times n}$. A^{T} stands for the transpose of matrix A. We denote by I the identity matrix. For matrix $A \in \mathbf{R}^{m\times n}$, the so-called stretching function vec(A) is defined by the following

$$\mathrm{vec}(A) = (\ a_1^{\mathrm{T}} \ \ a_2^{\mathrm{T}} \ \cdots \ a_n^{\mathrm{T}}\ )^{\mathrm{T}}$$

where a_k is the k-th column of A. Symbol $A \otimes B$ stands for Kronecker product of matrices A and B. Moreover, we define the inner product: $\langle A, B\rangle = \mathrm{tr}(B^{\mathrm{T}}A)$ for all $A, B \in \mathbf{R}^{m\times n}$. Then $\mathbf{R}^{m\times n}$ is a Hilbert inner product space, and the norm of a matrix generated by this inner product is the matrix Frobenius norm $||\cdot||$.

2 A brief description of Bi-CG and Bi-CR methods

To solve nonsymmetric linear systems as

$$Ax = b \tag{5}$$

where A is an $m \times m$ real nonsymmetric matrix and b is an m-vector; the Bi-CG and Bi-CR methods have been proposed as an extension of CG and CR, respectively.

First we present the Bi-CG algorithm and then based on the Bi-CG derivation, we present the Bi-CR algorithm. There are several ways to derive the algorithm of Bi-CG. Here we give the details of one of the simplest derivations[29]. By using (5) and a dual linear system $A^{\mathrm{T}}x^* = b^*$, we obtain the following $2m \times 2m$ symmetric linear system:

$$\begin{pmatrix} O & A \\ A^{\mathrm{T}} & O \end{pmatrix}\begin{pmatrix} x^* \\ x \end{pmatrix} = \begin{pmatrix} b \\ b^* \end{pmatrix}, \quad \text{or} \quad \hat{A}\hat{x} = \hat{b}. \tag{6}$$

Now for solving (6), we apply the CG algorithm with the following preconditioner:

$$P = \begin{pmatrix} O & I \\ I & O \end{pmatrix}. \tag{7}$$

Hence the resulting algorithm at the n-th iteration step can be written as

$$\begin{aligned}
&\hat{p}_n = P^{-1}\hat{r}_n + \beta_{n-1}\hat{p}_{n-1}\\
&\alpha_n = \frac{(P^{-1}\hat{r}_n, \hat{r}_n)}{(\hat{p}_n, \hat{A}\hat{p}_n)}\\
&\hat{x}_{n+1} = \hat{x}_n + \alpha_n\hat{p}_n\\
&\hat{r}_{n+1} = \hat{r}_n - \alpha_n\hat{A}\hat{p}_n\\
&\beta_n = \frac{(P^{-1}\hat{r}_{n+1}, \hat{r}_{n+1})}{(P^{-1}\hat{r}_n, \hat{r}_n)}.
\end{aligned}$$

Substituting P^{-1} of (7) and the vectors

$$\hat{x}_n = \begin{pmatrix} x_n^* \\ x_n \end{pmatrix} \qquad \hat{r}_n = \begin{pmatrix} r_n^* \\ r_n \end{pmatrix} \qquad \hat{p}_n = \begin{pmatrix} p_n^* \\ p_n \end{pmatrix} \tag{8}$$

into the previous recurrences, we obtain the following Bi-CG algorithm[30].

Algorithm 1. (Bi-CG algorithm)
x_0 is an initial guess, $r_0 = b - Ax_0$;
choose r_0^* (e.g., $r_0^* = r_0$);
set $p_{-1}^* = p_{-1} = 0$, $\beta_{-1} = 0$;
for $n = 0, 1, \cdots$, until convergence, do:

$$\begin{aligned}
&p_n = r_n + \beta_{n-1}p_{n-1};\\
&p_n^* = r_n^* + \beta_{n-1}p_{n-1}^*;\\
&s_n = Ap_n;\\
&s_n^* = A^{\mathrm{T}}p_n^*;\\
&\alpha_n = \frac{\langle r_n^*, r_n\rangle}{\langle p_n^*, s_n\rangle};\\
&x_{n+1} = x_n + \alpha_n p_n;\\
&r_{n+1} = r_n - \alpha_n s_n;\\
&r_{n+1}^* = r_n^* - \alpha_n s_n^*;\\
&\beta_n = \frac{\langle r_{n+1}^*, r_{n+1}\rangle}{\langle r_n^*, r_n\rangle}.
\end{aligned}$$

By using the preconditioned CR method with the preconditioner (7) to symmetric linear system (6), we have

$$\begin{aligned}
&\hat{p}_n = P^{-1}\hat{r}_n + \beta_{n-1}\hat{p}_{n-1}\\
&\alpha_n = \frac{(P^{-1}\hat{r}_n, \hat{A}P^{-1}\hat{r}_n)}{(P^{-1}\hat{A}\hat{p}_n, \hat{A}\hat{p}_n)}\\
&\hat{x}_{n+1} = \hat{x}_n + \alpha_n\hat{p}_n\\
&\hat{r}_{n+1} = \hat{r}_n - \alpha_n\hat{A}\hat{p}_n\\
&\beta_n = \frac{P^{-1}\hat{r}_{n+1}, \hat{A}P^{-1}\hat{r}_{n+1}}{(P^{-1}\hat{r}_n, \hat{A}P^{-1}\hat{r}_n)}.
\end{aligned}$$

Substituting P^{-1} of (7) and vectors (8) into the previous recurrences, we obtain the following Bi-CR algorithm[30].

Algorithm 2. (Bi-CR algorithm)
x_0 is an initial guess, $r_0 = b - Ax_0$;
choose r_0^* (e.g., $r_0^* = r_0$);
set $p_{-1}^* = p_{-1} = 0$, $\beta_{-1} = 0$;
for $n = 0, 1, \cdots$, until convergence, do:

$$\begin{aligned}
&p_n = r_n + \beta_{n-1}p_{n-1};\\
&p_n^* = r_n^* + \beta_{n-1}p_{n-1}^*;\\
&s_n = Ap_n;\\
&s_n^* = A^{\mathrm{T}}p_n^*;\\
&t_n = Ar_n;\\
&\alpha_n = \frac{\langle r_n^*, t_n\rangle}{\langle s_n^*, s_n\rangle};\\
&x_{n+1} = x_n + \alpha_n p_n;\\
&r_{n+1} = r_n - \alpha_n s_n;\\
&r_{n+1}^* = r_n^* - \alpha_n s_n^*;\\
&t_{n+1} = Ar_{n+1};\\
&\beta_n = \frac{\langle r_{n+1}^*, t_{n+1}\rangle}{\langle r_n^*, t_n\rangle}.
\end{aligned}$$

For more details about Bi-CG and Bi-CR methods see [29 – 31].

In the next section, based on Algorithms 1 and 2, we propose new algorithms for solving the generalized Sylvester-transpose matrix equation (1).

3 New algorithms

In this section, we develop Algorithms 1 and 2 to solve the generalized Sylvester-transpose matrix equation (1).

By using Kronecker product and vectorization operator, the generalized Sylvester-transpose matrix equation (1) can be transformed into the following nonsymmetric linear systems

$$\underbrace{\Big(\sum_{i=1}^{p}(B_i^{\mathrm{T}} \otimes A_i + (D_i^{\mathrm{T}} \otimes C_i)P) \Big)}_{A} \underbrace{\mathrm{vec}(X)}_{x} = \underbrace{\mathrm{vec}(E)}_{b} \quad (9)$$

where $A \in \mathbf{R}^{m^2 \times m^2}$, $x,\ b \in \mathbf{R}^{m^2}$, and $P \in \mathbf{R}^{m^2 \times m^2}$ is a unitary matrix[24]. It is obvious that the size of the above system is large. However, iterative methods will consume more computer time and memory space once the size of the system is large. To overcome this complication, we extend Algorithms 1 and 2. Now by considering the linear systems (9) and using the vectorization operator, we rewrite vectors r_n, r_n^*, p_n, p_n^*, s_n, s_n^*, t_n and x_n of Algorithms 1 and 2 into the matrix forms. We can write

$$\begin{aligned} r_0 = \ & b - Ax_0 \to r_0 = \mathrm{vec}(E) = \\ & (\sum_{i=1}^{p}(B_i^{\mathrm{T}} \otimes A_i + (D_i^{\mathrm{T}} \otimes C_i)P))x_0 \quad (10) \\ s_n = & \ Ap_n \to s_n = \\ & (\sum_{i=1}^{p}(B_i^{\mathrm{T}} \otimes A_i + (D_i^{\mathrm{T}} \otimes C_i)P))p_n \quad (11) \\ s_n^* = & \ A^{\mathrm{T}} p_n^* \to s_n^* = \\ & \sum_{i=1}^{p}(B_i^{\mathrm{T}} \otimes A_i + (D_i^{\mathrm{T}} \otimes C_i)P)^{\mathrm{T}} p_n^* = \\ & \sum_{i=1}^{p}(B_i \otimes A_i^{\mathrm{T}} + P(D_i \otimes C_i^{\mathrm{T}}))p_n^* \quad (12) \\ t_n = & \ Ar_n \to t_n = \\ & \sum_{i=1}^{p}(B_i^{\mathrm{T}} \otimes A_i + (D_i^{\mathrm{T}} \otimes C_i)P)r_n. \quad (13) \end{aligned}$$

By considering (10) – (13), we define

$$x_n = \mathrm{vec}(X_n), \quad s_n = \mathrm{vec}(S_n) \quad (14)$$
$$p_n = \mathrm{vec}(P_n), \quad r_n = \mathrm{vec}(R_n) \quad (15)$$
$$s_n^* = \mathrm{vec}(S_n^*), \quad p_n^* = \mathrm{vec}(P_n^*) \quad (16)$$
$$r_n^* = \mathrm{vec}(R_n^*), \quad t_n = \mathrm{vec}(T_n) \quad (17)$$

where $X_n, S_n, S_n^*, R_n, R_n^*, P_n, P_n^*, T_n \in \mathbf{R}^{m \times m}$. From (14)–(17), we can get

$$R_0 = E - \sum_{i=1}^{p}(A_i X_0 B_i + C_i X_0^{\mathrm{T}} D_i) \quad (18)$$

$$S_n = \sum_{i=1}^{p}(A_i P_n B_i + C_i P_n^{\mathrm{T}} D_i) \quad (19)$$

$$S_n^* = \sum_{i=1}^{p}(A_i^{\mathrm{T}} P_n^* B_i^{\mathrm{T}} + D_i P_n^{*\mathrm{T}} C_i) \quad (20)$$

$$T_n = \sum_{i=1}^{p}(A_i R_n B_i + C_i R_n^{\mathrm{T}} D_i) \quad (21)$$

$$P_n = R_n + \beta_{n-1} P_{n-1}, \qquad P_n^* = R_n^* + \beta_{n-1} P_{n-1}^* \quad (22)$$
$$X_{n+1} = X_n + \alpha_n P_n \quad (23)$$
$$R_{n+1} = R_n - \alpha_n S_n, \qquad R_{n+1}^* = R_n^* - \alpha_n S_n^*. \quad (24)$$

For Algorithms 1 and 2, parameters α_n and β_n can be determined as

$$\alpha_n = \frac{\langle r_n^*, r_n \rangle}{\langle p_n^*, s_n \rangle} = \frac{\langle \mathrm{vec}(R_n^*), \mathrm{vec}(R_n) \rangle}{\langle \mathrm{vec}(P_n^*), \mathrm{vec}(S_n) \rangle} = \frac{\langle R_n^*, R_n \rangle}{\langle P_n^*, S_n \rangle} \quad (25)$$

$$\beta_n = \frac{\langle r_{n+1}^*, r_{n+1} \rangle}{\langle r_n^*, r_n \rangle} = \frac{\langle \mathrm{vec}(R_{n+1}^*), \mathrm{vec}(R_{n+1}) \rangle}{\langle \mathrm{vec}(R_n^*), \mathrm{vec}(R_n) \rangle} = \frac{\langle R_{n+1}^*, R_{n+1} \rangle}{\langle R_n^*, R_n \rangle} \quad (26)$$

and

$$\alpha_n = \frac{\langle r_n^*, t_n \rangle}{\langle s_n^*, s_n \rangle} = \frac{\langle \mathrm{vec}(R_n^*), \mathrm{vec}(T_n) \rangle}{\langle \mathrm{vec}(S_n^*), \mathrm{vec}(S_n) \rangle} = \frac{\langle R_n^*, T_n \rangle}{\langle S_n^*, S_n \rangle} \quad (27)$$

$$\beta_n = \frac{\langle r_{n+1}^*, t_{n+1} \rangle}{\langle r_n^*, t_n \rangle} = \frac{\langle \mathrm{vec}(R_{n+1}^*), \mathrm{vec}(T_{n+1}) \rangle}{\langle \mathrm{vec}(R_n^*), \mathrm{vec}(T_n) \rangle} = \frac{\langle R_{n+1}^*, T_{n+1} \rangle}{\langle R_n^*, T_n \rangle}. \quad (28)$$

Here by applying (14) – (28), we present the matrix form of Algorithms 1 and 2 for finding the solution of (1).

Algorithm 3. (Matrix form of Bi-CG algorithm to solve (1))

$X_0 \in \mathbf{R}^{m \times m}$ is an initial guess and
$R_0 = E - \sum_{i=1}^{p}(A_i X_0 B_i + C_i X_0^{\mathrm{T}} D_i)$;
choose R_0^* (e.g., $R_0^* = R_0$);
set $P_{-1}^* = P_{-1} = 0$, $\beta_{-1} = 0$;
for $n = 0, 1, \cdots$, until convergence, do:
$P_n = R_n + \beta_{n-1} P_{n-1}$;
$P_n^* = R_n^* + \beta_{n-1} P_{n-1}^*$;
$S_n = \sum_{i=1}^{p}(A_i P_n B_i + C_i P_n^{\mathrm{T}} D_i)$;
$S_n^* = \sum_{i=1}^{p}(A_i^{\mathrm{T}} P_n^* B_i^{\mathrm{T}} + D_i P_n^{*\mathrm{T}} C_i)$;
$\alpha_n = \dfrac{\langle R_n^*, R_n \rangle}{\langle P_n^*, S_n \rangle}$;
$X_{n+1} = X_n + \alpha_n P_n$;
$R_{n+1} = R_n - \alpha_n S_n$;
$R_{n+1}^* = R_n^* - \alpha_n S_n^*$;
$\beta_n = \dfrac{\langle R_{n+1}^*, R_{n+1} \rangle}{\langle R_n^*, R_n \rangle}$.

Algorithm 4. (Matrix form of Bi-CR algorithm to solve (1))

$X_0 \in \mathbf{R}^{m \times m}$ is an initial guess and
$R_0 = E - \sum_{i=1}^{p}(A_i X_0 B_i + C_i X_0^{\mathrm{T}} D_i)$;
choose R_0^* (e.g., $R_0^* = R_0$);
set $P_{-1}^* = P_{-1} = 0$, $\beta_{-1} = 0$;
for $n = 0, 1, \cdots$, until convergence, do:
$P_n = R_n + \beta_{n-1} P_{n-1}$;
$P_n^* = R_n^* + \beta_{n-1} P_{n-1}^*$;
$S_n = \sum_{i=1}^{p}(A_i P_n B_i + C_i P_n^{\mathrm{T}} D_i)$;
$S_n^* = \sum_{i=1}^{p}(A_i^{\mathrm{T}} P_n^* B_i^{\mathrm{T}} + D_i P_n^{*\mathrm{T}} C_i)$;
$T_n = \sum_{i=1}^{p}(A_i R_n B_i + C_i R_n^{\mathrm{T}} D_i)$;
$\alpha_n = \dfrac{\langle R_n^*, T_n \rangle}{\langle S_n^*, S_n \rangle}$;
$X_{n+1} = X_n + \alpha_n P_n$;
$R_{n+1} = R_n - \alpha_n S_n$;
$R_{n+1}^* = R_n^* - \alpha_n S_n^*$;
$T_{n+1} = \sum_{i=1}^{p}(A_i R_{n+1} B_i + C_i R_{n+1}^{\mathrm{T}} D_i)$;
$\beta_n = \dfrac{\langle R_{n+1}^*, T_{n+1} \rangle}{\langle R_n^*, T_n \rangle}$.

In the next section, we report some numerical results to test the proposed iterative algorithms.

4 Numerical results

The aim of this section is to compare Algorithms 3 and 4 with the CG method (CG_M)[23], GI method (GI_M)[18] and the matrix LSQR method (LSQR_M)[16]. Our experiments were done in Matlab.

Example 1. In the first example, we consider the matrix equation $AXB = C$, where

$$A = \text{tril}(\text{rand}(m, m), 1) + \text{diag}(0.5 + \text{diag}(\text{rand}(m)))$$
$$B = \text{triu}(\text{rand}(m, m), 1) + \text{diag}(1 + \text{diag}(\text{rand}(m)))$$
$$C = \text{rand}(m, m).$$

For $m = 20$, Fig. 1 shows the convergence histories of the mentioned methods, where

$$r_n = \log_{10} ||C - AX_nB||. \tag{29}$$

From Fig. 1, it may be seen that Algorithms 3 and 4 have faster convergence rates than other methods.

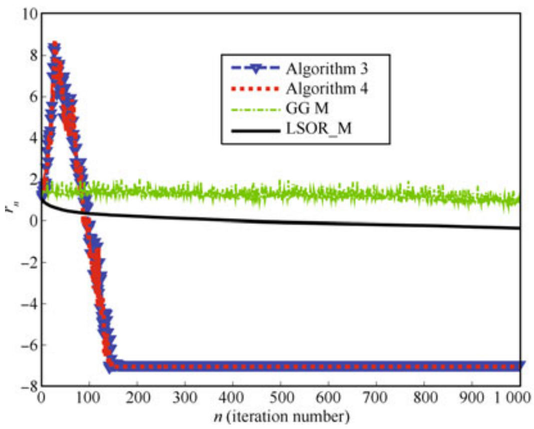

Fig. 1 The residual for Example 1

Example 2[32]. As the second example, the Sylvester matrix equation (3) is considered with the following parameters:

$$A = \text{tril}(\text{rand}(m, m), 1) + \text{diag}(1.75 + \text{diag}(\text{rand}(m)))$$
$$D = \text{triu}(\text{rand}(m, m), 1) + \text{diag}(2 + \text{diag}(\text{rand}(m)))$$
$$E = \text{rand}(m, m).$$

When $m = 100$, Fig. 2 represents the performance of the mentioned algorithms, where

$$r_n = \log_{10} ||E - AX_n - X_nD||. \tag{30}$$

From Fig. 2, we observe that Algorithms 3 and 4 are the most efficient among the five tested methods.

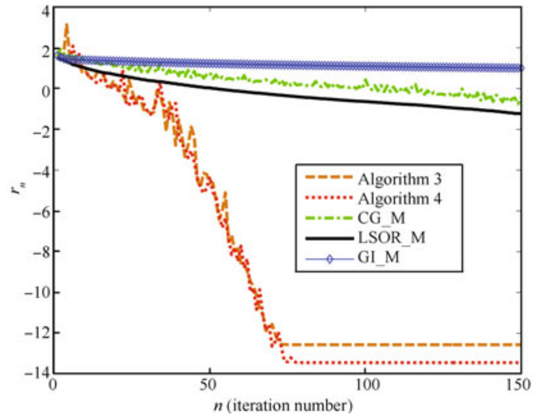

Fig. 2 The residual for Example 2

Example 3. Finally, we study the Sylvester-transpose matrix equation (4) with

$$A = \text{tril}(\text{rand}(m, m), 1) - \text{diag}(1.5 + \text{diag}(\text{rand}(m)))$$
$$B = \text{triu}(\text{rand}(m, m), 1) - \text{diag}(2 + \text{diag}(\text{rand}(m)))$$
$$C = \text{rand}(m, m).$$

For $m = 30$, the obtained results are depicted in Fig. 3 where

$$r_n = \log_{10}||C - AX_n - X_n^{\mathrm{T}}B||. \tag{31}$$

The results show that Algorithms 3 and 4 are quite efficient.

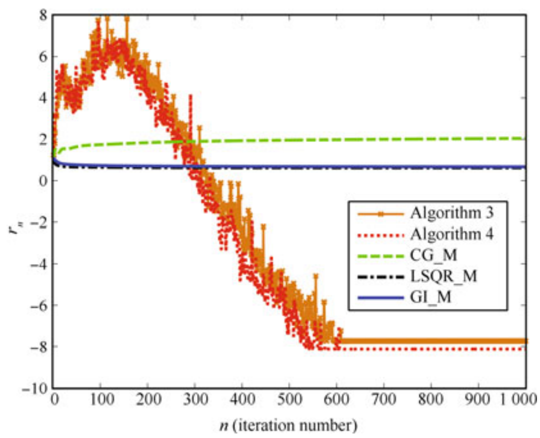

Fig. 3 The residual for Example 3

5 Conclusions

Linear matrix equations play a significant role in numerous problems in control, power systems and communication systems theory. By using Kronecker product and vectorization operator, we have proposed new neat algorithms based on Bi-CG and Bi-CR methods to find the solution of the generalized Sylvester-transpose matrix equations (1). The comparison results have shown that the proposed algorithms are more efficient than some existing algorithms.

Acknowledgments

The author would like to express his heartfelt thanks to the editor and anonymous referees for their valuable suggestions.

References

[1] G. R. Duan. Parametric approaches for eigenstructure assignment in high-order linear systems. *International Journal of Control, Automation, and Systems*, vol. 3, no. 3, pp. 419–429, 2005.

[2] B. Zhou, J. Lam, G. R. Duan. On Smith-type iterative algorithms for the Stein matrix equation. *Applied Mathematics Letters*, vol. 22, no. 7, pp. 1038–1044, 2009.

[3] G. R. Duan, R. J. Patton. Robust fault detection using Luenberger-type unknown input observers — A parametric approach. *International Journal of Systems Science*, vol. 32, no. 4, pp. 533–540, 2001.

[4] L. R. Fletcher, J. Kuatsky, N. K. Nichols. Eigenstructure assignment in descriptor systems. *IEEE Transactions on Automatic Control*, vol. 31, pp. 1138-1141, 1986.

[5] L. P. LaSalle, S. Lefschetz. *Stability by Liapunov's Direct Method: With Applications*, New York: Academic Press, 1961.

[6] D. S. Bernstein, D. C. Hyland. The optimal projection equations for reduced-order state estimation. *IEEE Transactions on Automatic Control*, vol. 30, no. 6, pp. 583–585, 1985.

[7] A. J. Laub. A Schur method for solving algebraic Riccati equations. *IEEE Transactions on Automatic Control*, vol. 24, no. 6, pp. 913–921, 1979.

[8] J. Ding, Y. J. Liu, F. Ding. Iterative solutions to matrix equations of the form $A_iXB_i = F_i$. *Computers and Mathematics with Applications*, vol. 59, no. 11, pp. 3500–3507, 2010.

[9] I. Kyrchei. Analogs of Cramer's rule for the minimum norm least squares solutions of some matrix equations. *Applied Mathematics and Computation*, vol. 218, no. 11, pp. 6375–6384, 2012.

[10] Q. Niu, X. Wang, L. Z. Lu. A relaxed gradient based algorithm for solving sylvester equations. *Asian Journal of Control*, vol. 13, no. 3, pp. 461–464, 2011.

[11] C. L. Yang, J. Z. Liu, Y. Liu. Solutions of the generalized Sylvester matrix equation and the application in eigenstructure assignment. *Asian Journal of Control*, vol. 14, no. 6, pp. 1669–1675, 2012.

[12] M. Dehghan, M. Hajarian. Two algorithms for finding the Hermitian reflexive and skew-Hermitian solutions of Sylvester matrix equations. *Applied Mathematics Letters*, vol. 24, no. 4, pp. 444–449, 2011.

[13] M. Dehghan, M. Hajarian. SSHI methods for solving general linear matrix equations. *Engineering Computations*, vol. 28, no. 8, pp. 1028–1043, 2012.

[14] M. A. Ramadan, N. M. El-Shazly, B. I. Selim. A Hessenberg method for the numerical solutions to types of block Sylvester matrix equations. *Mathematical and Computer Modelling*, vol. 52, no. 9–10, pp. 1716–1727, 2010.

[15] B. Zhou, Z. Y. Li, G. R. Duan, Y. Wang. Solutions to a family of matrix equations by using the Kronecker matrix polynomials. *Applied Mathematics and Computation*, vol. 212, no. 2, pp. 327–336, 2009.

[16] S. K. Li, T. Z. Huang. LSQR iterative method for generalized coupled Sylvester matrix equations. *Applied Mathematical Modelling*, vol. 36, no. 8, pp. 3545–3554, 2012.

[17] F. Ding, T. W. Chen. Iterative least-squares solutions of coupled Sylvester matrix equations. *Systems & Control Letters*, vol. 54, no. 2, pp. 95–107, 2005.

[18] F. Ding, T. W. Chen. On iterative solutions of general coupled matrix equations. *SIAM Journal on Control and Optimization*, vol. 44, no. 6, pp. 2269–2284, 2005.

[19] F. Ding, T. Chen. Gradient based iterative algorithms for solving a class of matrix equations. *IEEE Transactions on Automatic Control*, vol. 50, no. 8, pp. 1216–1221, 2005.

[20] F. Ding, T. W. Chen. Hierarchical least squares identification methods for multivariable systems. *IEEE Transactions on Automatic Control*, vol. 50, no. 3, pp. 397–402, 2005.

[21] M. Dehghan, M. Hajarian. On the generalized bisymmetric and skew-symmetric solutions of the system of generalized Sylvester matrix equations. *Linear and Multilinear Algebra*, vol. 59, no. 11, pp. 1281–1309, 2011.

[22] M. Dehghan, M. Hajarian. Solving the generalized Sylvester matrix equation $\sum_{i=1}^{p} A_iXB_i + \sum_{j=1}^{q} C_jYD_j = E$ over reflexive and anti-reflexive matrices. *International Journal of Control, Automation and Systems*, vol. 9, no. 1, pp. 118–124, 2011.

[23] M. Dehghan, M. Hajarian. On the generalized reflexive and anti-reflexive solutions to a system of matrix equations. *Linear Algebra and Its Applications*, vol. 437, no. 11, pp. 2793–2812, 2012.

[24] B. Zhou, J. Lamb, G. R. Duan. Toward solution of matrix equation $X = Af(X)B + C$. *Linear Algebra and Its Applications*, vol. 435, no. 6, pp. 1370–1398, 2011.

[25] F. Piao, Q. L. Zhang, Z. F. Wang. The solution to matrix equation $AX + X^{\mathrm{T}}C = B$. *Journal of the Franklin Institute*, vol. 344, no. 8, pp. 1056–1062, 2007.

[26] B. Zhou, Z. B. Yan. Solutions to right coprime factorizations and generalized Sylvester matrix equations. *Transactions of the Institute of Measurement and Control*, vol. 30, no. 5, pp. 397–426, 2008.

[27] B. Zhou, G. R. Duan. Solutions to generalized Sylvester matrix equation by Schur decomposition. *International Journal of Systems Science*, vol. 38, no. 5, pp. 369–375, 2007.

[28] B. Zhou, Z. Y. Li, G. R. Duan, Y. Wang. Weighted least squares solutions to general coupled Sylvester matrix equations. *Journal of Computational and Applied Mathematics*, vol. 224, no. 2, pp. 759–776, 2009.

[29] H. A. van der Vorst. *Iterative Krylov Methods for Large Linear Systems*, Cambridge: Cambridge University Press, 2003.

[30] T. Sogabe, M. Sugihara, S. L. Zhang. An extension of the conjugate residual method to nonsymmetric linear systems. *Journal of Computational and Applied Mathematics*, vol. 226, no. 1, pp. 103–113, 2009.

[31] Y. Saad. *Iterative Methods for Sparse Linear Systems*, 2nd ed., Philadelphia, PA: Society for Industrial and Applied Mathematics, 2003.

[32] C. Q. Gu, H. Y. Xue. A shift-splitting hierarchical identification method for solving Lyapunov matrix equations. *Linear Algebra and Its Applications*, vol. 430, no. 5–6, pp. 1517–1530, 2009.

Masoud Hajarian graduated from the Department of Mathematics, Shahid Chamran University of Ahvaz, Iran in 2006. He received his M. Sc. and Ph. D. degree in numerical linear algebra, both at Amirkabir University of Technology, Iran in 2008 and 2010, respectively. He is currently working as an assistant professor at the Department of Mathematics, Shahid Beheshti University, Iran. He is also acting as associate editor and reviewer for several journals.

His research interests include numerical linear algebra, matrix theory, numerical analysis and operational research.

E-mail: m_hajarian@sbu.ac.ir