COMPUTATIONAL & APPLIED MATHEMATICS
Volume 27, N. 1, pp. 47–59, 2008

ISSN 0101-8205
www.scielo.br/cam

Block linear method for large scale Sylvester equations

MARLLINY MONSALVE
Scientific Computing Center, Departamento de Computación, Facultad de Ciencias
Universidad Central de Venezuela, Ap. 47002, Caracas 1041-A, Venezuela
E-mail: mmonsalv@kuaimare.ciens.ucv.ve

Abstract. We present and analyze a new iterative scheme for large-scale solution of the well-known Sylvester equation. The proposed scheme is based on fixed point iteration approach and can make good use of the recently developed methods for solving block linear systems. It is shown mathematically that the iterative process converges under some assumptions on the coefficient matrices. Results on our numerical experiments with large-scale matrices are quite encouraging. In particular, the method compares favorably with the other block methods and a recently proposed method for Sylvester equation based on low-rank approximation of the right hand side matrix C.

Mathematical subject classification: 65J10, 65F10, 15A24.

Key words: Sylvester equation, block linear systems, iterative methods.

1 Introduction

In this paper, we present a new block algorithm for solving the Sylvester matrix equation (SE):

$$AX - XB = C, \tag{1}$$

where A, B, and C are given matrices of dimensions n, p, and $n \times p$, respectively and the matrix X of dimension $n \times p$ needs to be found.

It is well-known (see [6] and the original paper of Sylvester [16]) that the equation (1) has a unique solution if and only if A and B do not have an eigenvalue in

#658/06. Received: 15/III/06. Accepted: 16/IV/07.

common. The necessity of solving this equation arises in a wide variety of practical applications, including control systems design and analysis [4, 6], numerical solutions of differential equations, including boundary value problems [2, 8].

Because of its importance, the problem has been well studied in the literature and there now exist many methods for its solutions. An account of these methods can be found in the recent book by Datta [6]. In theory, the problem can be reduced to a linear system problem whose system matrix is a Kronecker product matrix of large dimension. The Kronecker-product approach is not numerically viable even for a small and dense system. For details see again the book by Datta [6]. The best-known and very widely used numerical method for small and dense problems is the Hessenberg-Schur method by Golub, Nash, and Van Loan [9]. The method is based on reduction of the largest of two matrices to Hessenberg form and the other to real Schur form. The Hessenberg-Schur method is an efficient implementation of the Bartels-Stewart method [1] proposed earlier, based on the reductions of both matrices to Schur forms. Unfortunately, these methods are not practical for large and sparse problems.

For large-scale Sylvester equations, several Krylov subspace methods have been recently proposed [5, 6, 7, 10, 11, 12, 14]. The Krylov methods are basically projection methods. The idea behind these methods is to first project the large problem into a smaller one by constructing an orthonormal basis of the Krylov subspace, then solve the smaller projected problem using a standard technique such as the Hessenberg-Schur method, and finally recover the solution of the original large problem from the solution of the smaller problem. In most cases, the smaller projected problem itself is a Sylvester equation. However, the method proposed in [11] by Hu and Reichel is different in the sense that their idea is to project the associated Kronecker system rather than the Sylvester equation itself. In the recent Ph.D Thesis [14], Peng has proposed two new Krylov subspace methods, one is divide-and-conquer type and the other is based on low-rank approximation of the right-hand matrix C. Yet another type of iterative methods which are neither Krylov methods nor based on solution of the Kronecker product systems have been proposed by several authors. For details see Miller [13] and the references therein. These iterative methods seem to be more of theoretical interests and are not practical for large problems.

In this paper, we propose a new iterative scheme based on fixed point iteration. The scheme requires solution of a linear systems with multiple right-hand sides at each iteration, which can be solved by using block Krylov subspace methods for linear systems (see Brezinski [3], Saad [15]). Our method method does not require solution of a low-dimensional Sylvester equation at every iteration. The results of our numerical experiments show that the proposed method works quite well for large-scale problems and is quite competitive with the other existing competing methods. Mathematical results on the convergence of our iterative scheme is also provided in the paper.

2 Methods based on SE and block linear systems

The main idea to solve the SE is to write this equation as a block linear system and then use some suitable iterative scheme. This can be accomplished by the following change of variable: $AX = Z$ where $Z = C + XB$. This possibility generates the iterative method:

Algorithm 1 Using: $AX = Z$

1: Given $X_0 \in \mathbb{R}^{n\times p}$
2: Compute $Z_0 = C + X_0 B$
3: **for** $k = 0, 1, \cdots$ until convergence **do**
4: **Solve** $AX_{k+1} = Z_k$
5: **Set** $Z_{k+1} = C + X_{k+1}B$
6: **end for**

Notice that, the method requires the solution of one block linear system and one matrix-matrix product per iteration. The matrix of coefficients of the internal system is of dimension n, therefore from a computational-cost point of view if $n < p$ it is convenient to use Algorithm 1 as described, but if $n > p$ it is convenient to transpose the Sylvester equation, and then apply Algorithm 1. To be precise, when $n > p$ solving $-B^T X^T + X^T A^T = C^T$, instead of (1), should be preferred. This guarantees that the matrix of coefficients of the internal system is of dimension p. On the other hand, the norm of the matrices A and B could also help to decide whether it is convenient to solve (1) or its transpose, as discussed below after Theorem 2.1.

Our next result establishes convergence under an additional hypothesis on the matrices A and B. This additional hypothesis is sufficient to guarantee also the existence of a unique solution, and as far as we know, it is the classical hypothesis assumed when dealing with iterative schemes that are not related to Krylov subspace methods, see [13].

Theorem 2.1. *Let $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{p\times p}$, $C \in \mathbb{R}^{n\times p}$ and let $\|.\|$ be an induced norm. If $\|A^{-1}\|\|B\| < 1$ then equation* (1) *has a unique solution, and the sequence $\{X_k\}_{k\geq 0}$ generated by Algorithm* 1 *converges q-linearly to the solution.*

Proof. Let $\lambda_i(A)$ be the eigenvalues of A for $1 \leq i \leq n$ and let $\lambda_j(B)$ be the eigenvalues of B for $1 \leq j \leq p$. Since $\|A\| \geq |\lambda_i(A)|$ for any i and for any induced norm we have $\|A^{-1}\| \geq \max_{1\leq i\leq n} |\lambda_i(A^{-1})|$ and thus,

$$\frac{1}{\|A^{-1}\|} \leq \frac{1}{\max_{1\leq i\leq n} |\lambda_i(A^{-1})|} = \min_{1\leq i\leq n} |\lambda_i(A)|. \tag{2}$$

On the other hand, $\|A^{-1}\|\|B\| < 1 \Rightarrow \|B\| < \frac{1}{\|A^{-1}\|}$, now using (2) we obtain,

$$\max_{1\leq j\leq p} |\lambda_j(B)| < \min_{1\leq i\leq n} |\lambda_i(A)| \Rightarrow \sigma(A) \cap \sigma(B) = \emptyset. \tag{3}$$

From (3) we conclude, that equation (1) has a unique solution since the spectra of A and B are disjoint.

For the convergence we proceed as follows. From Algorithm 1 we obtain that $X_k = A^{-1}(C + X_{k-1}B)$ and from SE we know that $X = A^{-1}(C + XB)$. Combining these two equations it follows that

$$X - X_k = A^{-1}(X - X_{k-1})B, \tag{4}$$

and so,

$$X - X_k = (A^{-1})^k(X - X_0)(B)^k.$$

Therefore

$$\|X - X_k\| \leq (\|A^{-1}\|\|B\|)^k\|(X - X_0)\|.$$

Since $\|A^{-1}\|\|B\| < 1$ then the sequence X_k converges to X when k tends to infinity. On the other hand, let $E_k = X - X_k$ the absolute error, and taking norm

on equation (4) we have that

$$\|E_k\| \leq \|A^{-1}\|\|B\|\|E_{k-1}\|,$$

since $\|A^{-1}\|\|B\| < 1$ then the sequence X_k converges q-linearly to the the solution. □

Notice that if no information is available about $\|A^{-1}\|$ or $\|B^{-1}\|$ then $\|A\|$ and $\|B\|$ could be of help to decide whether it is convenient to solve (1) or its transpose to increase the possibility of convergence. For example, if $\|A\|$ is much larger than $\|B\|$, then the condition $\|A^{-1}\|\|B\| < 1$ is likely to be satisfied for convergence when solving (1). Similarly, the transpose approach should be preferred when $\|B\|$ is much larger than $\|A\|$. Notice also that, if we want to solve SE with $B = A$ (or $B = A^T$), then Theorem 2.1 does not guarantee convergence, since $\|A\|\|A^{-1}\| = cond(A) \geq 1$. Unfortunately, this includes the well-know Lyapunov equation.

For the proposed algorithm, the residual matrix at iteration k is defined as $R_k = C - (AX_k - X_kB)$. This expression involves a high computational cost, since it require two matrix-matrix products per iteration. The following result provides an equivalent and less expensive way to calculate the residual.

Theorem 2.2. *Let $\{X_k\}_{k\geq 0}$ be the sequence generated by Algorithm* 1. *The following expressions for the residual matrix are equivalent:*

a) $R_k = C - (AX_k - X_kB) = C + X_kB - AX_k$.

b) $R_k = Z_k - Z_{k-1}$.

c) $R_k = (X_k - X_{k-1})B$.

Proof. The residual is given by $R_k = C + X_kB - AX_k$ and from Algorithm 1 we have that $Z_k = C + X_kB$ and $AX_k = Z_{k-1}$. Therefore

$$R_k = Z_k - Z_{k-1}. \tag{5}$$

On the other hand, substituting Z_k from Algorithm 1 in (5) we obtain $R_k = C + X_kB - (C + X_{k-1}B)$. Therefore

$$R_k = (X_k - X_{k-1})B. \tag{6 □}$$

Equation (5) provides a less expensive form to calculate the residual since it does not involve matrix-matrix products. Finally from (6) we obtain the following corollary which yields a more convenient stopping criterion.

Corollary 2.1. *Let* $\{X_k\}_{k\geq 0}$ *be the sequence generated by Algorithm* 1. *If* $\|X_k - X_{k-1}\| \to 0$, *then* $\|R_k\| \to 0$.

Finally, the following result establishes a bound that relates the norm of the residual and the norm of the error.

Theorem 2.3. *Let* $\{X_k\}_{k\geq 0}$ *be the sequence generated by Algorithm* 1. *The norm of the residual matrix satisfies that:*

$$\|R_k\| \leq (\|A\| + \|B\|)\|E_k\| \tag{7}$$

Proof. $R_k = C - (AX_k - X_kB) = AX - XB - AX_k + X_kB = A(X - X_k) - (X - X_k)B = AE_k - E_kB$. Therefore

$$\|R_k\| = \|AE_k - E_kB\| \leq (\|A\| + \|B\|)\|E_k\| \qquad \square$$

3 Numerical results

In this section we present preliminary numerical experiments to illustrate the performance of the new proposed algorithm. For solving the internal block linear systems for this algorithm we use the block SOR scheme (BSOR), the non-Hermitian block steepest descent method (BSD) and the block minimal residual iteration (BMR) fully described in [3]. For the proposed algorithm, we will only present the combination that better results generated in CPU time. For each experiment, we will indicate which method was used to solve the internal block linear system.

We compare the new proposed algorithm with the following methods for solving SE:

- The block Arnoldi-Sylvester algorithm (BAS($\widetilde{m}$)) and the block GMRES-Sylvester algorithm (BGS($\widetilde{m}$)) proposed and fully described in [10]. These methods are based on Krylov subspace methods and solve a small SE per

iteration of order $\widetilde{m}\,p^2$, where $\widetilde{m}$ is the restart value. These internal SE are solved using the algorithm proposed by Golub, Nash and Van Loan in [9]. The BAS($\widetilde{m}$) and BGS($\widetilde{m}$) can only be used when $n >> p$. When $n < p$ we can always transpose SE and then apply the same techniques.

- Galerkin method for Sylvester equation (GMSE($\widetilde{m}$)) and the restarted minimal projected residual algorithm (RMPR($\widetilde{m}$)) proposed and fully described in [11]. These methods, as well as BAS($\widetilde{m}$) and BGM($\widetilde{m}$), are based on Krylov subspace methods and solve a small SE per iteration of order $\widetilde{m}^2$. The internal SE is written as a linear system of equation of order $\widetilde{m}^2$ that is solved using direct methods, but the operation count for these algorithms is $\widetilde{m}np$ flops. GMSE($\widetilde{m}$) and RMPR($\widetilde{m}$) can be used for any values of n and p.

- Arnoldi method for low-rank approximate solution to the Sylvester equation (LRASE(k, $\widetilde{m}$)) proposed and fully described in [14]. The parameter k is a small value required by the algorithm, and $\widetilde{m}$ is the restart value. This method can be used for any values of n and p. In all our experiment, we prove for $1 \leq k \leq 5$ and we report the best result.

In our tests we use matrices from the Harwell-Boeing collection and also some sparse matrices from the Matlab gallery, and for these matrices the inequalities of Theorem 2.1 hold, therefore convergence is guaranteed for the new proposed algorithm. In all cases, the entries of the solution matrix X are $x_{ij} = f(x_i, y_j)$ where: $x_i = ih_n$, $y_j = jh_p$ for $0 \leq i \leq n$, $0 \leq j \leq p$, $h_n = 1/(n+1)$, $h_p = 1/(p+1)$ and $f(x, y) = xe^{xy}\sin(\pi x)\sin(\pi y)$. This is a typical test function that appears in several works, see e.g. [11, 10]. We stop the process when

$$\|X - X_k\| = \|E_k\| \leq 10^{-8},$$

or when 2500 external iterations are reached. The initial iterate, X_0, is the null matrix. To stop the iterative methods for the internal block linear systems we use the norm of the residual. To be precise, we stop the internal iterations of Algorithm 1 when $\|AX_{k+1} - Z_k\| \leq 10^{-2}$. The experiments were run in a Pentium IV computer at 3.4 GHz with 2 GB of RAM memory, using Matlab

7.0. In examples 3.1–3.4 we compare the performance of Algorithm 1 with other methods. In these examples we report number of iterations (Iter), CPU time, norm of the residual (Residual) and norm of the absolute error (Error). In example 3.5 we compare the different equivalent expressions for computing the residual norm.

We used the following notation for characterizing the different finalization states of the experiments, "–" means that the algorithm accomplished the maximum number of external iterations, "s" implies that the used method produced stagnation problems and "**" means that the memory requirements of the algorithm could not be supported by Matlab.

Example 3.1. In this example $n < p$ and n is a small value. The matrix B is the matrix *Hor__131* of the Harwell-Boeing collection with $p = 434$, and A is a sparse matrix of order n generated with the Matlab function `gallery('wathen',nx,ny)` where $n = 3*nx*ny+2*nx+2*ny+1$ which returns a sparse random finite element matrix.

Methods	Iter	CPU time	Residual	Error
BAS(4)	15	2.20	1.21e-008	7.89e-009
BGS(4)	2	0.81	4.11e-012	4.06e-013
GMSE(4)	104	16.26	1.02e-007	9.19e-009
RMPR(4)	99	16.12	1.60e-007	7.28e-009
LRASE(1,4)	198	1.67	3.88e-008	9.86e-009
Algorithm 1 + BSOR	19	0.36	9.18e-008	5.49e-009

Table 1 – Performance of BAS($\widetilde{m}$), BGS($\widetilde{m}$), GMSE($\widetilde{m}$), RMPR($\widetilde{m}$), LRASE($k,\widetilde{m}$) and Algorithm 1 combined with BSOR for solving SE when B =*Hor__131* and A is a sparse random finite element matrix with $nx = 2$, $ny = 1$ and $n = 13$.

In example 3.1 reported in Table 1 we can see that all methods converge but GMSE and RMPR are not competitive in CPU time. In this case, Algorithm 1 combined with BSOR required less CPU time than other methods, but BAS($\widetilde{m}$), BGS($\widetilde{m}$) and LRASE could still be considered as competitive choices. Moreover BGS achieves better accuracy than the other options.

Example 3.2. In this example, A is the matrix *orsirr_2* with $n = 886$ and B is the matrix *sherman1* with $p = 1000$, both of the Harwell-Boeing collection.

Methods	Iter	CPU time	Residual	Error
BAS(4)	**	**	**	**
BGS(4)	**	**	**	**
GMSE(4)	-	4692.6	-	0.699
RMPR(4)	s	s	s	1.23
LRASE(1,4)	-	27299.79	-	14.62
Algorithm 1 + BSOR	1567	53494.81	1.03e-007	9.95e-009

Table 2 – Performance of BAS($\widetilde{m}$), BGS($\widetilde{m}$), GMSE($\widetilde{m}$), RMPR($\widetilde{m}$), LRASE($k, \widetilde{m}$) and Algorithm 1 combined with BSOR for solving SE when $A = orsirr\_2$ and $B = sherman1$.

As shown in Table 2, BAS and BGS produce storage problems. In this experiment $\|E_0\| = 616.24$, and we can observe that GMSE, RMPR and LRASE reduce the absolute error norm. RMPR produces stagnation problems while GMSE and LRASE stopped because the maximum number of external iterations was reached. Finally, Algorithm 1 with BSOR converges but the CPU time required is considerably higher.

Example 3.3. In this example $p < n$ and p is a small value. The matrix A is the matrix *gre__343* of the Harwell-Boeing collection with $n = 343$, and B is a tridiagonal matrix of order $p = 20$ given by

$$B = \alpha * tridiag(-1 - p_1 h,\ 2 - p_2 h^2,\ -1 + p_1 h) \tag{8}$$

with

$$h = \frac{1}{p+1}, \quad \alpha = -\frac{1}{h^2}, \quad \text{and} \quad p_1 = 100, \quad p_2 = 50.$$

This matrix has been used by several authors, see [10, 11, 14]. In this experiment we report Algorithm 1 combined with BMR.

In Table 3, we can observe that all methods converge but Algorithm 1 obtained the best performance in CPU time. BAS, GMSE and RMPR are similar in CPU time required to satisfy $tolE$ and the CPU time required by these methods is

Methods	Iter	CPU time	Residual	Error
BAS(4)	12	6.70	3.70e-006	4.15e-009
BGS(4)	1	8.21	9.48e-011	1.41e-013
GMSE(4)	368	6.96	8.30e-005	9.93e-009
RMPR(4)	344	6.39	5.32e-005	8.54e-009
LRASE(1,4)	248	3.67	7.80e-006	9.73e-009
Algorithm 1 + BMR	5	0.078125	5.56e-007	6.29e-010

Table 3 – Performance of BAS($\widetilde{m}$), BGS($\widetilde{m}$), GMSE($\widetilde{m}$), RMPR($\widetilde{m}$), LRASE(k, $\widetilde{m}$) and Algorithm 1 combined with BMR for solving SE when $A = gre\_\_343$ and $B =$ is a tridiagonal matrix.

almost twice the one required by LRASE. On the other hand, BGS converges in one iteration and it reduces the norm of the error more than the others, but the CPU time required by BGS is higher than the one required by the others. It is worth noticing that for this example, we use the values of p_1 and p_2 suggested in [11].

Example 3.4. In this example, A is the matrix *gre_1107* with $n = 1107$ and B is the matrix *fs_760_1* with $p = 760$, both from the Harwell-Boeing collection.

Methods	Iter	CPU time	Residual	Error
BAS(4)	**	**	**	**
BGS(4)	**	**	**	**
GMSE(4)	-	4586.23	-	2.4682
RMPR(4)	s	s	s	2.6002
LRASE(1,3)	-	28879.03	-	755.96
Algorithm 1 + BSD	636	1596.26	0.147e-002	9.58e-009

Table 4 – Performance of BAS($\widetilde{m}$), BGS($\widetilde{m}$), GMSE($\widetilde{m}$), RMPR($\widetilde{m}$), LRASE(k, $\widetilde{m}$) and Algorithm 1 combined with BSD for solving SE when $A =$ *gre_343* and $B =$ *fs_760_1*.

As shown in Table 4, BAS and BGS produce storage problems. RMPR produces stagnation problems while GMSE and LRASE stopped because the maximum number of external iterations was reached. Algorithm 1 with BSD con-

verges. In this experiment $\|E_0\| = 458.5$ and we can see that GMSE and RMPR reduced the absolute error norm while LRASE increased it. Finally, we can observe that Algorithm 1 achieves convergence but $\|R_k\|$ is much higher than $\|E_k\|$.

Example 3.5. In this example we want to compare $\|E_k\|$, with the equivalent expressions for the norm of the residual described in Theorem 2.3. These expressions will be denoted by $\|R_k\| = \|C - (AX_k - X_kB)\|$, $\|R_k^1\| = \|Z_k - Z_{k-1}\|$ and $\|R_k^2\| = \|(X_k - X_{k-1})B\|$. The matrix A is the matrix *fs_680_1* of the Harwell-Boeing collection and B is a sparse matrix with random entries. The dimensions of these matrices are $n = 680$ and $p = 1000$ respectively. We use Algorithm 1 combined with BSOR to solve SE.

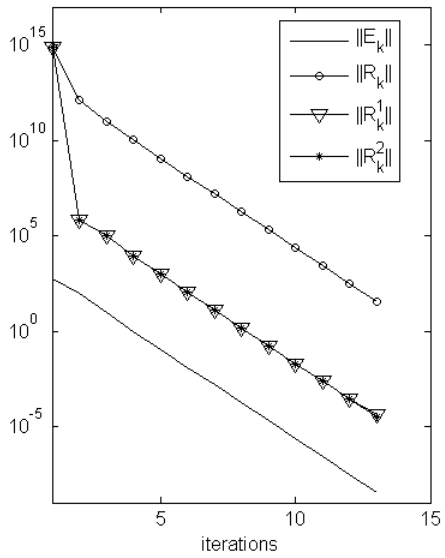

Figure 1 – Comparison of $\|E_k\|$, $\|R_k\|$, $\|R_k^1\|$ and $\|R_k^2\|$ when we use Algorithm 1 combined with BSOR, to solve a SE with A =*fs_680_1* and B has random entries.

In Figure 1, we can see that $\|R_k^1\|$ and $\|R_k^2\|$ are close to $\|E_k\|$ during the process, while $\|R_k\|$ is not. In this example $\|A\| + \|B\| = 7.19e + 013$ and the behavior observed in Figure (1) agrees and illustrates (7). Finally, we can conclude that if the stopping criterion is $\|R_k\| \leq tolE$, it is convenient to use $\|R_k^1\|$ instead of $\|R_k\|$, for two important reasons. The first one is computational

work: $\|R_k\|$ requires 2 matrix-matrix product, while $\|R_k^1\|$ requires a matrix substraction. The second reason is that $\|R_k^1\|$ is more accurate to measure the precision of the approximation generated by the proposed algorithms.

4 Concluding remarks

We propose a new iterative scheme for solving Sylvester Equations (SE). At each iteration a block linear system of equations is solved and, for that, direct or iterative techniques can be used. This new scheme can be applied regardless of the dimensions of the involved matrices and, in most cases, it requires less computational work than competitors.

We also establish the conditions for which convergence is guaranteed. These conditions are sufficient but not necessary, and therefore strong. Nevertheless, they also guarantee the existence of a unique solution of the SE. Unfortunately, for some important applications these conditions are not satisfied, and our proposed scheme cannot be used. For example, the new scheme cannot be applied for solving the well-known Lyapunov equation.

Finally, we present an equivalent, stable, and inexpensive way of computing the residual matrix. This equivalent formula yields a very efficient stopping criterion, as shown in our numerical experiments.

Acknowledgement. I wish to thank Wujian Peng from Northern Illinois University for providing the Matlab code for LRASE and Prof. Biswa Datta for his helpful comments and recommendations that help the quality of the presentation. I am indebted to two anonymous referees whose comments helped me to improve the quality of this paper.

REFERENCES

[1] R.H. Bartels and G.W. Stewart, *Algorithm 432, solution of the matrix equation $AX - XB = C$*. Comm. ACM, **15** (1972), 820–826.

[2] W.G. Bickley and J. McNamee, *Matrix and other direct methods for the solution of systems of linear difference equations*. Philos. Trans Roy. Soc. London Ser. A., **252** (1960), 69–131.

[3] C. Brezinski, *Block descent methods and hybrid procedures for linear systems*. Numerical Algorithms, **29** (2002), 21–32.

[4] B. Datta, *Linear and numerical linear algebra in control theory: Some research problems*. Linear Algebra and its Appl., **198** (1994), 755–790.

[5] B. Datta, *Krylov subspace methods for large-scale matrix problems in control*. Future Generation Computer Systems, **19**(7) (2003), 1253–1263.

[6] B. Datta, *Numerical Methods for Linear Control Systems Design and Analysis*. Elsevier Academic Press, New York, 2003.

[7] B. Datta and Y. Saad, *Arnoldi methods for large Sylvester-like observer matrix equation, and an associated algorithm for partial spectrum assigment*. Linear Algebra and its Appl., **156** (1991), 225–244.

[8] A. Dou, *Method of undetermined coefficients in linear differential systems and the matrix equation* $YA - AY = F$. SIAM J. Appl. Math., **14** (1966), 691–696.

[9] G.H. Golub, S. Nash and C. Van Loan, *A Hessemberg-Schur method for the problem* $AX - XB = C$. IEE Trans. Automat. Control, **39** (1979), 167–188.

[10] A. El Guennouni, K. Jbilou and A.J. Riquet, *Block Krylov subspace methods for solving large Sylvester equations*. Numerical Algorithms, **29** (2001), 75–96.

[11] D.Y. Hu and L. Reichel, *Krylov-subspace methods for the Sylvester equation*. Linear Algebra Appl., **172** (1992), 283–313.

[12] I.M. Jaimoukha and E.M. Kasenally, *Krylov subspaces methods for solving large Lyapunov equations*. SIAM J. Numerical Anal., **31** (1994), 227–251.

[13] D.F. Miller, *The iterative solution of the matrix equation* $XA + BX + C = 0$. Linear Algebra Appl., **105** (1988), 131–137.

[14] Wujian Peng, *On the Krylov subspace solutions of matrix equations in control theory*. PhD thesis, Northern Illinois University, 2004.

[15] Y. Saad, *Iterative Methods for Sparse Linear Systems*. International Thompson Publishing Co., London, England, 1996.

[16] J.J. Sylvester, *Sur l'equation en matrices* $px = xq'$. C.R. Acad. Sci Paris, **99** (1884), 67–71 : 115–116.