Applied Mathematics, 2013, 4, 6-10
http://dx.doi.org/10.4236/am.2013.41002 Published Online January 2013 (http://www.scirp.org/journal/am)

A New Iterative Scheme for Solving the Semi Sylvester Equation

Saeed Karimi, Fatemeh Attarzadeh
Department of Mathematics, Persian Gulf University, Bushehr, Iran
Email: karimi@pgu.ac.ir, fatemehattarzadeh66@yahoo.com

Received October 20, 2012; revised November 20, 2012; accepted November 27, 2012

ABSTRACT

In this paper, the Galerkin projection method is used for solving the semi Sylvester equation. Firstly the semi Sylvester equation is reduced to the multiple linear systems. To apply the Galerkin projection method, some propositions are presented. The presented scheme is compared with the $\mathcal{L}$-GL-LSQR algorithm in point of view CPU-time and iteration number. Finally, some numerical experiments are presented to show that the efficiency of the new scheme.

Keywords: Iterative Method; Multiple Linear Systems; Galerkin Projection Method; $\mathcal{L}$-GL-LSQR Algorithm

1. Introduction

We want to solve, using Galerkin projection method, the following semi Sylvester equation

$$\boldsymbol{A}X - \boldsymbol{E}X\boldsymbol{B} = \boldsymbol{C}, \tag{1}$$

where $\boldsymbol{A}$, $\boldsymbol{E}$, $\boldsymbol{B}$ and $\boldsymbol{C}$ are $n\times n$, $n\times n$, $m\times m$ and $n\times m$ matrices, respectively, and the unknown matrix X is $n\times m$. Equation (1) has a unique solution if and only if (A,C) and (B^T,I) are regular matrix pairs with disjoint spectra, which will be assumed throughout this paper.

The semi Sylvester equations appear frequently in many areas of applied mathematics. We refer the reader to the elegant survey by Bhatia and Rosenthal [1] and references therein for a history of the equation and many interesting and important theoretical results. The semi Sylvester equations are important in a number of applications such as matrix eigen-decompositions [2,3], control theory [4,5], model reduction [6-8], numerical solution of matrix differential Riccati equations, and many more.

When the sizes of the coefficient matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ are small, the popular and widely used numerical method is the Hessenberg-Schur algorithm [9]. For large and sparse matrices $\boldsymbol{A}$ and $\boldsymbol{B}$, iterative schemes to solve the semi Sylvester equations such as those based on the matrix sign function or the Newton method are widely used [10-12]. During the last years, several projection methods based on Krylov subspace methods have also been proposed, see, e.g., [13-15] and the references therein. The main idea developed in these methods is to construct suitable bases of the Krylov subspace and projects the large problem into a smaller one. Naturally, a direct method such the one developed in [16] is used to solve the projected problem. The final step in the projection process consists in recovering the solution of the original problem from the solution of the smaller problem.

The semi Sylvesters Equation (1), in the special case, can be reduced to the following multiple linear systeam

$$A^{(i)}x^{(i)} = b^{(i)} \qquad i = 1,2,\cdots,n. \tag{2}$$

In [17], Ton F. Chan and Michael K. Ng presented the Galerkin projection method for solving linear systems (2). In this paper, we convert the semi Sylvester Equation (1) to the multiple linear system. We present some propositions for applying the Galerkin projection method to solve the reduced semi Sylvester equation. For showing the efficiency of the new scheme, in point of view CPU-time and iteration number, we compare the new scheme with the $\mathcal{L}$-GL-LSQR algorithm [18]. Note that the semi Sylvester Equation (1) is converted to standard Sylvester equation, when $\boldsymbol{E}$ be a identity matrix I.

The remainder of the paper is organized as follows. Section 2 is devoted to a short review of the Galerkin projection method. In Section 3, we show that how to apply the Galerkin projection method for solving the semi Sylvester Equation (1). Some numerical experiments and comparing the new scheme with the $\mathcal{L}$-GL-LSQR algorithm, is devoted in Section 4. Finally, we make some concluding remarks in Section 5.

2. The Short Summary of Galerkin Projection Method

In this section, we consider using conjugate gradient (CG) methods for solving multiple linear systems

$A^{(i)}x^{(i)}=b^{(i)}$, for $1\le i\le s$, where the coefficient matrices $A^{(i)}$ and the right-hand side $b^{(i)}$ are different in general. In particular, we focus on the seed projection method which generates a Krylov subspace from a set of direction vectors obtained by solving one of the systems, called the seed system, by the CG method and then projects the residuals of other systems onto the generated Krylov subspace to get the approximate solutions. The whole process is repeated until all the systems are solved. CG methods can be seen as iterative solution methods to solve a linear system of equations by minimizing an associated quadratic functional. For simplicity, we let

$$f_i(x)=\frac{1}{2}x^{\mathrm{T}}A^{(i)}x-\left(b^{(i)}\right)^{\mathrm{T}}x, \tag{3}$$

be the associated quadratic functional of the linear system $A^{(i)}x^{(i)}=b^{(i)}$. The minimizer of f_j is the solution of the linear system $A^{(i)}x^{(i)}=b^{(i)}$. The idea of the projection method is that for each restart, a seed system $A^{(k)}x^{(k)}=b^{(k)}$ is selected from the unsolved ones which are then solved by the CG method. An approximate solution $\hat{x}^{(j)}$ of the nonseed system $A^{(j)}x^{(j)}=b^{(j)}$ can be obtained by using search direction p_i^k generated from the ith iteration of the seed system. More precisely, given the ith iterate x_i^j of the nonseed system and the direction vector p_i^k, the approximate solution $\hat{x}^{(j)}$ is found by solving the following minimization problem

$$\min_{\eta} f_j\left(x_i^j+\eta p_i^k\right). \tag{4}$$

It is easy to check that the minimizer of (4) is attained at

$$\hat{x}^{(j)}=x_i^j+\eta p_i^k, \tag{5}$$

where

$$\eta=\frac{\left(p_i^k\right)^{\mathrm{T}}r_i^j}{\left(p_i^k\right)^{\mathrm{T}}A^{(j)}p_i^k}\quad\text{and}\quad r_i^j=b^{(j)}-A^{(j)}x_i^j. \tag{6}$$

After the seed system $A^{(k)}x^{(k)}=b^{(k)}$ is solved to the desired accuracy, a new seed system is selected and the whole procedure is repeated. We note from (6) that the matrix-vector multiplication $A^{(j)}p_i^k$ is required for each projection of the nonseed iteration. In general, the matrix-vector multiplication $A^{(j)}p_i^k$ cannot be computed cheaply. The cost of the method will be expensive in the general case where the matrices $A^{(j)}$ and $A^{(k)}$ are different.

In order to reduce the extra cost in projection method in the general case, we propose using the modified quadratic function $\overline{f}_j$,

$$\overline{f}_i(x)\equiv\frac{1}{2}x^{T}A^{(k)}x-\left(b^{(i)}\right)^{T}x,$$

to compute the approximate solution of the nonseed system. Note that we have used $A^{(k)}$ instead of $A^{(j)}$ in the above definition. In this case, we determine the next iterate of the nonseed system by solving the following minimization problem:

$$\min_{\alpha}\overline{f}_j\left(x_i^j+\alpha p_i^k\right). \tag{7}$$

The approximate solution $\hat{x}^{(j)}$ of the nonseed system $A^{(j)}x^{(j)}=b^{(j)}$ is given by

$$\hat{x}^{j}=x_i^j+\alpha p_i^k, \tag{8}$$

where

$$\alpha=\frac{\left(p_i^k\right)^{\mathrm{T}}\overline{r}_i^{\,j}}{\left(p_i^k\right)^{\mathrm{T}}A^{(k)}p_i^k}\quad\text{and}\quad\overline{r}_i^{\,j}=b^{(j)}-A^{(k)}x_i^j. \tag{9}$$

Now the projection process does not require the matrix-vector product involving the coefficient matrix $A^{(j)}$ of the nonseed system. Therefore, the method does not increase the dominant cost (matrix-vector multiplies) of each CG iteration. Of course, unless $A^{(j)}$ is close to $A^{(k)}$ in some sense, we do not expect this method to work well because $\overline{f}_j$ is then far from the current f_j. So we have the following algorithm.

Algorithm 1 [17]: Preconditioned version of Projection Method

1. Set for $k=1,\cdots,s$ until all the systems are solved
2. Select the *k*th system as seed.
3. For $i=0,1,2,\cdots,m_{k+1}$ CG iteration,
4. For $j=k,k+1,\cdots,s$ unsolved systems
5. If $j=k$ then perform usual CG steps
6. $\delta_i^{k,k}=\left(z_i^{k,k}\right)^{T}r_i^{k,k}\Big/\left(z_{i-1}^{k,k}\right)^{T}\left(z_{i-1}^{k,k}\right)$
7. $p_i^{k,k}=z_i^{k,k}+\delta_i^{k,k}p_{i-1}^{k,k}$
8. $\sigma_i^{k,k}=\left(z_i^{k,k}\right)^{\mathrm{T}}r_i^{k,k}\Big/\left(p_i^{k,k}\right)^{\mathrm{T}}A^{(k)}p_i^{k,k}$
9. $x_{i+1}^{k,k}=x_i^{k,k}+\sigma_i^{k,k}p_i^{k,k}$
10. $r_{i+1}^{k,k}=r_i^{k,k}-\sigma_i^{k,k}A^{(k)}p_i^{k,k}$
11. $z_i=\left(C^{(k)}\right)^{-1}r_{i+1}^{k,k}$ preconditioning
12. Else perform Galerkin projection
13. $\eta_i^{k,j}=\left(z_i^{k,k}\right)^{\mathrm{T}}r_i^{k,j}\Big/\left(p_i^{k,k}\right)^{\mathrm{T}}A^{(k)}p_i^{k,k}$
14. $x_{i+1}^{k,j}=x_i^{k,j}+\eta_i^{k,j}p_i^{k,k}$
15. $r_{i+1}^{k,j}=r_i^{k,j}-\eta_i^{k,j}A^{(k)}p_i^{k,k}$
16. end if
17. end for
18. end for
19. end for

3. Solving the Semi Sylvester Equation

In this section, we focus on numerical solution of the

semi Sylvester equations

$$\boldsymbol{A}X - \boldsymbol{E}X\boldsymbol{B} = \boldsymbol{C}, \tag{10}$$

where $\boldsymbol{A} \in \mathbb{R}^{n\times n}$, $\boldsymbol{B} \in \mathbb{R}^{m\times m}$, $\boldsymbol{E} \in \mathbb{R}^{n\times n}$, $\boldsymbol{C} \in \mathbb{R}^{n\times m}$ and $X \in \mathbb{R}^{n\times m}$ is the solution matrix. The semi Sylvester equation (10) has a unique solution if and only if (A,C) and (B^{T},I) are regular matrix pairs with disjoint spectra.

Now we want to apply Galerkin projection methods for solving the semi Sylvester Equation (10). Let B be symmetric matrix then semi Sylvester Equation (10) can be reduced to the following multiple linear systeams

$$A^{(i)}x^{(i)} = b^{(i)},\ i = 1,\cdots,s, \tag{11}$$

where $A^{(i)} \in \mathbb{R}^{n\times n}$, $b^{(i)} \in \mathbb{R}^{n\times 1}$ and $x^{(i)} \in \mathbb{R}^{n\times 1}$.

By using the Schur decomposition, there is a unitary matrix Q_B such that

$$B = Q_B \Lambda_B Q_B^{\mathrm{T}}, \tag{12}$$

where Λ_B is diagonal matrix and it's digonal components are the eigenvalues of B. By substitution of (12) in (10), we have

$$AX - EXQ_B\Lambda_B Q_B^{\mathrm{T}} = C,$$

$$AXQ_B - EXQ_B\Lambda_B = CQ_B.$$

By taking $\hat{X} = XQ_B$ and $\hat{C} = CQ_B$, we obtain the following multiple linear systems

$$\hat{A}^{(i)}\hat{x}^{(i)} = \hat{c}^{(i)},\ i = 1,\cdots,m, \tag{13}$$

where $\hat{A}^{(i)}$, $\hat{x}^{(i)}$ and $\hat{c}^{(i)}$ are $(A-\lambda_i E)$, i-th column of the matrix $\hat{X}$ and i-th column of the matrix $\hat{C}$, respectively.

To use the Galerkin projection method for solving the multiple linear systeams (13), it is need the matrix $\hat{A}^{(i)}$, $i = 1,2,\cdots,m$ to be symmetric positive definite. So the following propositons are presented and their proof are clear.

Proposition 1: Let $\boldsymbol{A}$ and $\boldsymbol{B}$ are symmetric matrices and $\boldsymbol{E}$ is symmetric positive definite matrix and

$$\lambda_j < \frac{(\boldsymbol{A}x,x)}{(\boldsymbol{E}x,x)}, \qquad j = 1,\cdots,s,$$

where λ_j be the eigenvalues of matrix $\boldsymbol{B}$. Then $\hat{A}^{(i)}$ is symmetric positive difinite.

Proposition 2: Let $\boldsymbol{A}$, $\boldsymbol{B}$ and $\boldsymbol{E}$ be symmetric positive difinite matrices and symmetric positive semi-difinite matrix, respectively. Then $(\boldsymbol{A}-\lambda_j\boldsymbol{E}), j = 1,\cdots,s$ are symmetric positive definite, where λ_j be the eigenvalues of matrix $\boldsymbol{B}$.

Note that the semi Sylvester Equation (1) is converted to standard Sylvester equation, when $\boldsymbol{E}$ be a identity matrix I. Now by using the assumptions of the above propositions, we can apply the previous algorithm to solve the semi Sylvester Equation (10).

4. Numerical Experiments

In this section, all the numerical experiments were computed in double precision with some MATLAB codes. For all the examples the initial guess X_0 was taken to be zero matrix. We apply the new scheme for solving the standard Sylvester equation.

For obtaining the numerical solution of the Sylvester equation by using new scheme, we consider two examples. At the first example the dense coefficient matrix is as the form

$$\boldsymbol{A} = \mathrm{randint}(500),\ E = I$$

where $\mathrm{randint}(m)$ is an $m\times m$ matrix of uniformly distributed random integers and I is an identity matrix. At the second example the coeffient matrix is

$$\boldsymbol{A} = \mathrm{tridiag}\left(-1+\frac{1}{n+1}, 3, -1+\frac{1}{n+1}\right),$$

$$\boldsymbol{E} = \mathrm{tridiag}\left(-1+\frac{1}{n+1}, 4, -1+\frac{1}{n+1}\right)$$

where $tridiag(a,b,c)$ is 4000×4000 tridiagonal matrix. In both examples the matrix $\boldsymbol{B}$ is symmetric tridiagonal matrix with s size, as follows

$$\boldsymbol{B} = -\mathrm{tridiag}\left(-1+\frac{1}{s+1}, 5, -1+\frac{1}{s+1}\right).$$

Also the right-hand side matrix $\boldsymbol{C}$ in both examples is normalized random matrix. The $\mathcal{L}$-GL-LSQR algorithm and Galerkin projection method are stopped when the residual norms of the first and second examples are less than 10^{-7} and 10^{-6}, respectively.

In **Table 1**, we compare two $\mathcal{L}$-GL-LSQR algorithm and Galerkin projection method for the first test example. We show that Galerkin projection method is better than $\mathcal{L}$-GL-LSQR algorithm in point of view CPU-time and iteration number. In this table, (s)m is the sum of iterations when we apply the Galerkin projection method for solving s linear systems (2). In **Table 2**, we compare the $\mathcal{L}$-GL-LSQR algorithm and Galerkin projection method for the second test example. We find that the $\mathcal{L}$-GL-LSQR algorithm is better in point of veiw the iteration number and the new scheme is better in point of veiw the CPU-time. Finally, **Figures 1** and **2** show that the efficientcy of the Galerkin projection method for solving the semi Sylvester equation. In both tables, the symbols **cond** and **iter** are the condition number of the coefficient matrix $\boldsymbol{A}$ and the iteration number of the methods, respectively.

5. Conclusion

We proposed a new scheme for solving the semi Syl-

Table 1. Numerical results of the new scheme and $\mathcal{L}$-GL-LSQR algorithm for the semi Sylvester equation with the first example.

cond	Error	iter	CPU-time	Method (n, s)
1.2599e+006	1.7581e−008	2160	96.8445	GL-LSQR (500, 5)
1.2599e+006	4.4181e−008	(5)537	8.6389	Galerkin (500, 5)

Table 2. Numerical results of the new scheme and $\mathcal{L}$-GL-LSQR algorithm for the semi Sylvester equation with the second example.

cond	Error	iter	CPU-time	Method (n, s)
4.9970	1.3837e−07	23	0.09578	GL-LSQR (4000, 4)
4.9970	5.0897e−007	(4)40	0.04339	Galerkin (4000, 4)

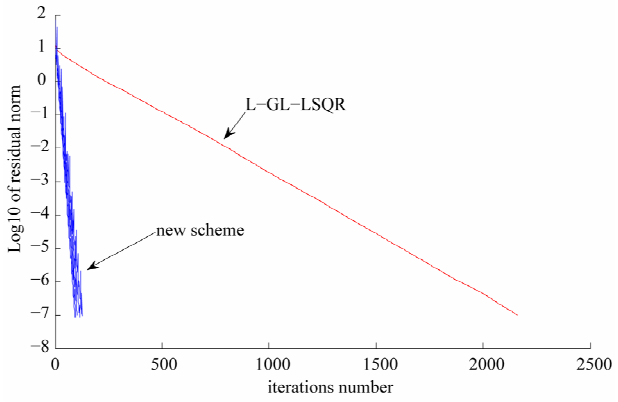

Figure 1. Comparing the new scheme with the $\mathcal{L}$-GL-LSQR algorithm with n = 500, s = 5.

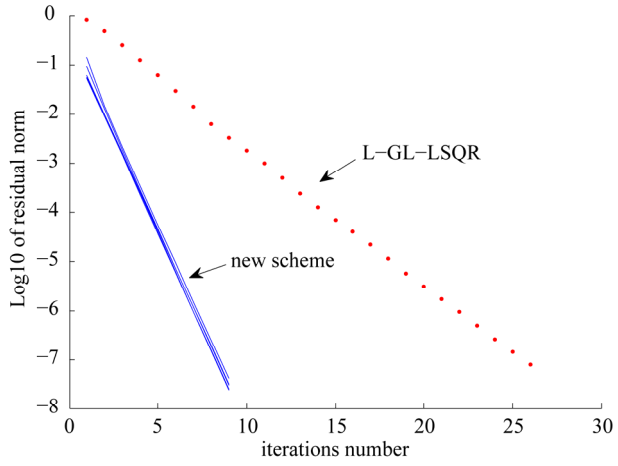

Figure 2. Comparing the new scheme with the $\mathcal{L}$-GL-LSQR algorithm with n = 4000 s = 4.

vester equation. By forcing some conditions on coefficent matrices $\boldsymbol{A}$, $\boldsymbol{B}$ and $\boldsymbol{E}$, We converted the semi Sylvester equation to s linear systems with different coefficent matrices and right-hands. Then we applied the Galerkin projection method for obtaining the numerrical solution, and we showed the efficiency of the new scheme. In **Table 1**, was shown that when the condition number of the coefficient matrix $\boldsymbol{A}$ is to be large then the new scheme is better then the $\mathcal{L}$-GL-LSQR algorithm for solving the semi Sylvester equation. But in **Table 2**, the $\mathcal{L}$-GL-LSQR algorithm is better than new scheme in point of view iteration number for sparse and well-conditioned matrix $\boldsymbol{A}$.

6. Acknowledgements

The authors would like to thank the anonymouws referees for their helpful suggestions, which would greatly improve the article.

REFERENCES

[1] R. Bhatia and P. Rosenthal, "How and Why to Solve the Operator Equation AX − XB = Y," *Bulletin London Mathematical Society*, Vol. 29, No. 1, 1997, pp. 1-21. doi:10.1112/S0024609396001828

[2] G. H. Golub and C. F. Van Loan, "Matrix Computations," 3rd Edition, Johns Hopkins University Press, Baltimore, 1996.

[3] V. Sima, "Algorithms for Linear-Quadratic Optimization," Vol. 200, *Pure and Applied Mathematics*, Marcel Dekker, Inc., New York, 1996.

[4] B. Datta, "Numerical Methods for Linear Control Systems," Elsevier Academic Press, New York, 2004.

[5] A. Lu and E. Wachspress, "Solution of Lyapunov Equations by ADI Iteration," *Computers & Mathematics with Applications*, Vol. 21, 1991, pp. 43-58. doi:10.1016/0898-1221(91)90124-M

[6] A. C. Antoulas, "Approximation of Large-Scale Dynamical Systems," *Advances in Design and Control*, SIAM, Philadelphia, 2005.

[7] U. Baur and P. Benner, "Cross-Gramian Based Model Reduction for Data-Sparse Systems," *Technical Report*, Fakultat fur Mathematik, TU Chemnitz, Chemnitz, 2007.

[8] D. Sorensen and A. Antoulas, "The Sylvester Equation and Approximate Balanced Reduction," *Linear Algebra and Its Applications*, Vol. 351-352, 2002, pp. 671-700. doi:10.1016/S0024-3795(02)00283-5

[9] G. H. Golub, S. Nash and C. Van Loan, "A Hessenberg Schur Method for the Problem AX + XB = C," *IEEE Transactions on Automatic Control*, Vol. 24, No. 6, 1979, pp. 909-913. doi:10.1109/TAC.1979.1102170

[10] P. Benner, "Factorized Solution of Sylvester Equations with Applications in Control," In: B. De Moor, B. Motmans, J. Willems, P. Van Dooren and V. Blondel, Eds., *Proceedings of the 16th International Symposium on Mathematical Theory of Network and Systems*, Leuven, 5-9 July 2004.

[11] W. D. Hoskins, D. S. Meek and D. J. Walton, "The Numerical Solution of the Matrix Equation XA + AY = F," *BIT*, Vol. 17, 1977, pp. 184-190.

[12] D. Y. Hu and L. Reichel, "Application of ADI Iterative Methods to the Restoration of Noisy Images," *Linear Algebra and Its Applications*, Vol. 172, No. 15, 1992, pp. 283-313. doi:10.1016/0024-3795(92)90031-5

[13] A. El Guennouni, K. Jbilou and A. J. Riquet, "Block Krylov Subspace Methods for Solving Large Sylvester Equations," *Numerical Algorithms*, Vol. 29, No. 1, 2002, pp. 75-96. doi:10.1023/A:1014807923223

[14] K. Jbilou, "Low Rank Approximate Solutions to Large Sylvester Matrix Equations," *Applied Mathematics and Computation*, Vol. 177, No. 1, 2006, pp. 365-376. doi:10.1016/j.amc.2005.11.014

[15] M. Robb and M. Sadkane, "Use of Near-Breakdowns in the Block Arnoldi Method for Solving Large Sylvester Equations," *Applied Numerical Mathematics*, Vol. 58, No. 4, 2008, pp. 486-498. doi:10.1016/j.apnum.2007.01.025

[16] G. H. Golub, S. Nash and C. Van Loan, "A Hessenberg Schur Method for the Problem AX + XB = C," *IEEE Transactions on Automatic Control*, No. 24, No. 6, 1979, pp. 909-913.

[17] T. Chan and M. Ng, "Galerkin Projection Methods for Solving Multiple Linear Systems," Department of Mathematics, University of California, Los Angeles, 1996.

[18] S. Karimi, "A New Iterative Solution Method for Solving Multiple Linear Systeams," *Advances in Linear Algebra and Matrix Theory*, Vol. 1, 2012, pp. 25-30. doi:10.4236/alamt.2012.23004

