Universiteit Utrecht

Mathematics Institute

A Comparison of Subspace Methods for Sylvester Equations

by

Jan Brandts

Preprint

No. 1183

March, 2001

A Comparison of Subspace Methods for Sylvester Equations

Jan Brandts[*]

March, 2001

Abstract

Sylvester equations $AX - XB = C$ play an important role in numerical linear algebra. For example, they arise in the computation of invariant subspaces, in control problems, as linearizations of algebraic Riccati equations, and in the discretization of partial differential equations. For small systems, direct methods are feasible. For large systems, iterative solution methods are available, like Krylov subspace methods.

It can be observed that there are essentially two types of subspace methods for Sylvester equations: one in which block matrices are treated as rigid objects (functions on a grid), and one in which the blocks are seen as a basis of a subspace.

In this short note we compare the two different types, and aim to identify which applications should make use of which solution methods.

1 Different types of Sylvester equations

In this paper, we study solution methods for Sylvester equations $AX - XB = C$. Here, A and B are square matrices of size n and k, whereas C and the unknown X are matrices of dimensions $n \times k$. We distinguish between two different types of solutions X that frequently occur in practical applications.

(A) As a numerical approximation to the solution of a partial differential equation, X may represent a function on a rectangular grid.

(B) X may represent a k-dimensional subspace of $\mathbb{R}^n$ in algorithms for computation of invariant subspaces; merely the column span of X is of interest.

A natural context for equations of type (A) is to view the solution X as an element of the Hilbert space $\mathcal{H}(n,k)$ of $n \times k$ matrices endowed with the Frobenius inner product $\langle G, H \rangle = \text{trace}(G^*H)$ and its derived Frobenius norm $\|\cdot\|_F$. This setting enables Ritz-Galerkin projection onto subspaces in a canonical way. Another feasible solution method for equations of this type, in which X is also not seen as a number of column vectors, is MultiGrid.

[*]Mathematical Institute, Utrecht University, P.O.Box 80.010, 3508 TA, Utrecht, The Netherlands. E-mail: brandts@math.uu.nl

Equations of type (B) are different in the sense that it does not really matter whether X or XF is produced by the numerical algorithm, where F may be any basis transformation of $\mathbb{R}^k$; indeed, right-multiplication of X by F does not change the column span, showing that F does not even have to be known explicitly. This freedom should, whenever possible, be exploited by the solution algorithms.

We remind the reader [4] that the Sylvester equation $AX - XB = C$ is non-singular if and only if A and B do not have an eigenvalue in common. For perturbation theory (which is different than for general linear systems) we refer to [6].

1.1 Kronecker product formulation

Recall that any Sylvester equation can be written as an ordinary linear system of equations since $\mathbf{T} : X \mapsto AX - XB$ is a linear mapping on $\mathbb{R}^{n\times k}$. Defining a function **vec** from the space of $n \times k$ matrices to the space of nk vectors by

$$\mathbf{vec}(X) = \mathbf{vec}\left(\left[\begin{array}{c|c|c} x_1 & \cdots & x_k \end{array}\right]\right) = (x_1^*, \cdots, x_k^*)^*, \tag{1}$$

the action of $\mathbf{T}$ can be mimicked by an ordinary left-multiplication:

$$\mathbf{vec}(\mathbf{T}(X)) = \mathbf{vec}(AX - XB) = (I_k \otimes A - B^* \otimes I_{n-k})\,\mathbf{vec}(X). \tag{2}$$

Here, I_q is the $q \times q$ identity matrix and $\otimes$ the Kronecker product, which, for general matrices $Y = (y_{ij})$ and $Z = (z_{ij})$, is defined as,

$$Y \otimes Z = \begin{bmatrix} y_{11}Z & \cdots & y_{1n}Z \\ \vdots & & \vdots \\ y_{n1}Z & \cdots & y_{nn}Z \end{bmatrix}. \tag{3}$$

Observation 1.1 *The Kronecker product formulation in $\mathbb{R}^{n\times k}$ endowed with the standard ℓ_2-inner product is equivalent to the formulation in the space $\mathcal{H}(n,k)$ by the identity*

$$\mathbf{vec}(A)^*\mathbf{vec}(B) = \langle A, B\rangle. \tag{4}$$

This shows that the application of standard solution methods for linear systems to the Kronecker product formulation of a Sylvester equation, results in methods that are particularly fit for equations of type (A).

1.2 Basis transformations and assumptions

In theory, but practically only feasible if k is small, any basis transformation $BF = FT$ of B can be used to change the equation $AX - XB = C$ into

$$AY - YT = CF \quad \text{with} \quad Y = XF \quad \text{and} \quad T = F^{-1}BF. \tag{5}$$

This shows for example that if B is diagonalizable, the Sylvester equation reduces to k decoupled linear systems.

We will assume that $k \leq n$ and that k and n are such, that direct solution methods are not feasible. Hence we concentrate on iterative methods. Moreover we assume that if k is small, B is not diagonalizable, since the resulting decoupling would remove the typical Sylvester features and lead to ordinary linear systems.

2 Two model problems

In order to illustrate the two different types of Sylvester equations mentioned in the previous section, we will now describe two sets of model problems. The first set of problems depends on a parameter that changes a partial differential equation from diffusive to convective, whereas in the second set, the matrix A can be taken from the Harwell-Boeing collection.

2.1 A model problem of type (A): Convection-Diffusion equation

Consider the following simple convection-diffusion problem defined on a rectangular domain Ω, with constant convection vector $b = (b_1, b_2)^*$ and right-hand side f,

$$-\Delta u + b^*\nabla u = f \quad \text{in } \Omega, \quad u = 0 \quad \text{on } \partial\Omega. \tag{6}$$

We will use a grid of rectangles on Ω, where the x_1-direction is subdivided into $n+1$ intervals of size h, and the x_2-direction into $k+1$ intervals of size s. This yields $n \times k$ unknowns $u(ih, js)$ that can be collected in an $n \times k$ matrix $X = (x_{ij})$ with $x_{ij} = u(ih, js)$. Note that due to numbering and notational conventions, the *vertical* columns of X represent the *horizontal* x_1-direction. The following discrete problem results,

$$\left(\frac{1}{h^2}D_n + \frac{b_1}{2h}K_n\right)X + X\left(\frac{1}{s^2}D_k + \frac{b_2}{2s}K_k\right) = F. \tag{7}$$

Here, D_j, for j either n or k, is the $j \times j$ tridiagonal matrix corresponding to the [-1 2 -1] approximation to the second derivative, and K_j the $j \times j$ tridiagonal matrix corresponding to the [-1 0 1] approximation to the first derivative. Left multiplication by these matrices represents differentiation in the x_1 direction, and right-multiplication differentiation in the x_2 direction. Finally, $F = (f_{ij}) = (f(ih, js))$.

2.2 A model problem of type (B): Invariant Subspace problem

A typical invariant subspace problem for a given matrix A would be to find a full-rank long tall matrix Y and a small matrix M such that $AY = YM$. If such Y and M are found, it also holds that $A\hat{X} = \hat{X}(\hat{X}^*A\hat{X})$, where $\hat{X}R = Y$ symbolizes a QR-decomposition of Y. This is because $\hat{\Pi} := I - \hat{X}\hat{X}^*$ represents orthogonal projection on the orthogonal complement of the columnspan of $\hat{X}$, so $\hat{\Pi}A\hat{X} = 0$. Now suppose we have an orthogonal matrix X_j that approximates the invariant subspace $\hat{X}$, then a new and hopefully better approximation X_{j+1} can be found by solving

$$AX_{j+1} - X_{j+1}(X_j^*AX_j) = AX_j - X_j(X_j^*AX_j). \tag{8}$$

This is one iteration of the block Rayleigh quotient method. Clearly, it is only the column span of X_{j+1} that is of interest here.

Remark 2.1 Another approach leads to a Sylvester equation that is neither of type (A) nor (B). Let $\Pi := I - X_jX_j^*$. Then $X_j + Q$ with $Q^*X_j = 0$ spans an invariant subspace if Q satisfies

$$X_j^*Q = 0 \quad \text{and} \quad \Pi AQ - Q(X_j^*AX_j) = Q(X_j^*A)Q - AX_j. \tag{9}$$

This is a generalized algebraic Riccati equation [2] for Q. Approximations to solutions Q can be found by iteration: set $Q_0 = 0$ and solve the Sylvester equations

$$X_j^* Q_{i+1} = 0 \quad \text{and} \quad \Pi A Q_{i+1} - Q_{i+1}(X_j^* A X_j) = Q_i (X_j^* A) Q_i - A X_j. \tag{10}$$

Since Q_i denotes a *correction* to an invariant subspace approximation, the precise columns of Q_i are indeed of interest. But since the columns of X_j are to a certain extend arbitrary, no particular structure can be expected to be present in Q_i. For theory on convergence of the above and related iterations, we refer to [13, 3, 9].

3 Iterative methods for the Sylvester equation

An iterative algorithm for the Sylvester equation will basically have the following structure. Given an initial guess X_0 for the solution X, we compute the residual $R_0 := C - AX_0 + X_0 B$, put $k = 0$, solve U_k approximately and cheaply from the residual correction equation $AU_k - U_k B = R_k$, and update

$$C_k := AU_k - U_k B, \quad R_{k+1} := R_k - C_k, \quad X_{k+1} := X_k + U_k, \quad k := k + 1, \tag{11}$$

after which the process is repeated if necessary. If U_k is solved exactly, then clearly, $X_{k+1} = X$. Otherwise, the hope is that the algorithm will produce a sequence X_k that eventually converges to X. The equivalent of classical ideas in linear system theory leading to Richardson, Jacobi and Gauss-Seidel can be applied by replacing A and B by their diagonals or upper triangular parts in order to get approximations for U_k. For a study of SOR applied to the Kronecker formulation, see [12].

3.1 Preconditioning by direct methods on approximate systems

Right multiplication of $AU_k - U_k B = R_k$ by the j-th canonical basis vector e_j of $\mathbb{R}^k$ leads, after a simple rearrangement, to

$$(A - b_{jj} I) u_j = C e_j + U_k (B - b_{jj} I) e_j. \tag{12}$$

In case B is upper triangular, the columns u_j ($j = 1, \ldots, k$) of U_k can be solved from (12) recursively since in the right-hand side of (12), only the columns $u_1, \ldots, u_{j-1}$ appear. Assuming that A is lower triangular, left-multiplication with e_j^* leads to a similar construction. Bringing both A and B on triangular form leads to a system that can be solved directly. This is the Bartels-Stewart algorithm [1]. As observed by Golub, Nash and Van Loan [5], it may be more efficient to bring the largest of the two matrices merely on Hessenberg form. Clearly, both methods can play an important role as preconditioners in iterative methods.

3.2 Residual correction in a Krylov subspace

The main idea of Krylov subspace methods like GCR, GMRES and FOM [4] is that the residual correction takes place by projection onto a Krylov subspace of some dimension m. If more than one cycle of (11) is necessary for sufficient accuracy, one speaks of a restarted method, like GMRES(m). Here we will study one cycle only, so, residual correction in an m-dimensional Krylov subspace. In the literature, two essentially different types of Krylov subspace methods for Sylvester equations are frequently found. In the first, one Krylov subspace belonging to the operator $\mathbf{T}$ is used to project upon. In the second, a Krylov subspace for A is tensored with a (left-)Krylov subspace for B and the result is used to project upon.

3.2.1 Krylov subspace methods of type (I)

Krylov subspace methods can be applied to the Kronecker product formulation (2) of a Sylvester equation. By Observation 1.1, it follows that in GCR, GMRES and FOM, a linear combination of the matrices $\mathbf{T}(R_0), \ldots, \mathbf{T}^m(R_0)$ is determined that approximates the initial residual R_0 in some sense. Explicitly, in GCR and GMRES, scalars $\gamma_1, \ldots, \gamma_m$ are determined such that

$$R_1^A := R_0 - \sum_{j=1}^{m} \mathbf{T}^j(R_0)\gamma_j \tag{13}$$

has minimal Frobenius norm, while in the Galerkin method FOM those scalars are determined such that R_1^A resulting from (13) is $\langle \cdot, \cdot \rangle$-orthogonal to $\mathbf{T}^j(R_0)$ for all $j = 1, \ldots, m$.

3.2.2 Krylov subspace methods of type (II)

The second approach, due to Hu and Reichel [7], is to associate Krylov subspaces to A and B separately, and to construct the tensor product space. Generally, assume that V_p is an orthogonal $n \times p$ matrix and W_q an orthogonal $k \times q$ matrix. Then, each $p \times q$ matrix Y_{pq} induces an approximation $V_p Y_{pq} W_q^*$ of the solution U_0 of $AU_0 - U_0 B = R_0$ by demanding that

$$V_p^*(AV_pY_{pq}W_q^* - V_pY_{pq}W_q^*B - R_0)W_q = 0. \tag{14}$$

By the identity

$$\mathbf{vec}(V_pY_{pq}W_q^*) = (W_q \otimes V_p)\mathbf{vec}(Y_{pq}) \tag{15}$$

it can be seen that (14) is a Galerkin projection onto the pq-dimensional subspace space $W_q \otimes V_p$ of $\mathbb{R}^{nk}$. By choosing for V_p and W_q block Krylov subspaces with starting blocks full rank matrices R_A and R_B such that $R_0 = R_A R_B^*$, (14) can be written as

$$H_A Y_{pq} - Y_{pq} H_B^* = (V_p^* R_A)(W_p^* R_B)^*, \tag{16}$$

where $H_A := V_p^* A V_p$ is $p \times p$ upper Hessenberg, $H_B = W_p^* B^* W_p$ is $q \times q$ upper Hessenberg, and both $V_p^* R_A$ and $W_p^* R_B$ tall upper triangular matrices. It was shown by Simoncini [8] that this Galerkin method results in a truncation of an exact series representation of the solution in terms of block Krylov matrices and minimal polynomials. Hu and Reichel [7] also present a minimal residual method based on the same idea.

Remark 3.1 In the case that k is small, W_q may be chosen as the $k \times k$ identity matrix. The action of B is then used exactly. The resulting projected equation is then

$$H_A Y_{pk} - Y_{pk} B = V_p^* R_0. \tag{17}$$

After computing a Schur decomposition for B, the Golub-Nash-Van Loan algorithm [5] can then be employed to solve the projected system.

3.2.3 Comparison of the costs

In the Galerkin method of type (I), the subspaces consist of m blocks of size $n \times k$ while the projected matrix is only of size $m \times m$. A sparse Sylvester action costs $\mathcal{O}(kn^2 + k^2 n)$ operations. The orthogonalization in step j costs j Frobenius inner products, each of costs kn^2, so up to step m the construction of the Hessenberg matrix and the projected right-hand side costs $\mathcal{O}(km^2n^2)$. Constructing the solution of the Hessenberg system costs only $\mathcal{O}(m^2)$ operations. Producing the solution of the large system costs $\mathcal{O}(mkn^2)$. So, assuming that $k << n$ is small, the overall costs are $\mathcal{O}(mnk)$ for storage and $\mathcal{O}(km^2n^2)$ for computation.

In the method of type (II), the storage is $pn + qk$ for the two Krylov matrices. The construction of those matrices costs about $pn^2 + qk^2$ for the actions of sparse A and B. Orthogonalizations are $\mathcal{O}(p^2n^2)$ and $\mathcal{O}(q^2k^2)$. The Hessenberg matrices are of size $p \times p$ and $q \times q$ and solution is about $\mathcal{O}(k^3 + kp^2)$ for Schur decomposition and solving k Hessenberg systems. Again assuming that $k << n$, the storage costs are dominated by $\mathcal{O}(pn)$ and the computational costs by $\mathcal{O}(p^2n^2)$.

Observation 3.2 Assuming that $p \approx km$, which means that the number of n vectors involved in the projection process is for both methods the same, the second method is slightly more computationally expensive. Put differently, with the same computational costs, the first method is more efficient in the use of memory.

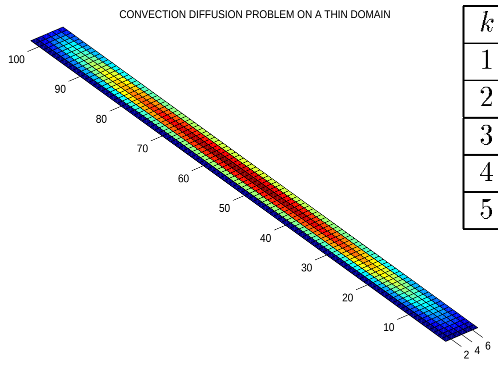
3.3 Implementation of the Galerkin methods

The implementation of the Galerkin methods FOM(I) and FOM(II) of type (I) and (II) respectively, is done through Arnoldi orthonormalization of the blocks from which the approximation is constructed. The orthogonalization takes place in different inner products, and for different operators. For FOM(I), the operator $\mathbf{T}$ is used, for FOM(II) we assume that C has full rank and put W_p equal to the identity of size k as in Remark 3.1. The Arnoldi parts are given as MatLab-like code below.

```
************ META-CODE USED IN FOM(I) * META-CODE USED IN FOM(II) *********
                                     *
function [V,H,E] = BARNOLDI(A,B,C,m); * function [V,H,E] = BARNOLDI(A,C,m);
                                     *
E = FrobNorm(C); V{1} = C/E;         * [V{1},E] = qr(C,0);
for k=2:m                            * for k=2:m
   W = A*V{k-1} - V{k-1}*B;          * W = A*V{k-1};
   for j = 1:k-1;                    *   for j = 1:k-1;
      H(j,k-1) = trace(V{j}'*W);     *      H{j,k-1} = V{j}'*W;
      W        = W - V{j}*H(j,k-1);  *      W        = W - V{j}*H{j,k-1};
   end                               *   end
   H(k,k-1) = FrobNorm(W);           *  [V{k},H{k,k-1}] = qr(W,0);
   V{k}     = W/H(k,k-1);            * end
end                                  *
                                     *
**************************************************************************
```

4 Numerical experiments

Both FOM(I) and FOM(II) will be applied to solve the Sylvester equations of type (A) and (B) described in Section 2. First problem is the convection-diffusion problem of Section 2.1 with $n = 200$ and different values for k and $h = s = 0.001$. This could correspond to a problem in a thin tube. Convection parameter was set to ten and in the long direction only. Listed in Table 1 is the amount of flops needed to get a relative residual reduction of 10^{-6}, and also the number of iterations.



k	flops(I)	iters(I)	flops(II)	iters(II)
1	1.1e6	49	1.1e6	49
2	2.1e6	47	1.9e6	44
3	3.1e6	47	2.7e6	43
4	4.4e6	48	3.3e6	41
5	5.9e6	50	3.7e6	39

Table 1. Number of flops and number of iterations for different values of k.

Left: the solution on a long thin strip.

As a second problem we took one iteration of the Block Raleigh Quotient iteration, as explained in Section 2.2, applied to the matrix SHERMAN2 from the Harwell-Boeing collection. This is a matrix of size 1080×1080. Again, for different values of k, we computed the next iterate with both FOM(I) and FOM(II) starting with the same approximation. In Table 2 below, the results are given in the same format as for Table 1.

k	flops(I)	iters(I)	flops(II)	iters(II)
1	3.5e5	5	3.3e5	5
2	2.0e6	12	8.5e5	6
3	2.2e7	46	2.8e6	11
4	1.2e7	26	2.2e6	7
5	4.2e7	49	1.9e6	5
10	∞	∞	5.3e6	6

Table 2.

4.1 Conclusions

In both cases, the method FOM(II) performed better than FOM(I). For the problem of type (A), the difference is small, and also it should be noted that in spite of the slightly larger number of flops needed for FOM(I), it was faster in time. For the problem of type (B), clearly FOM(II) outperformed FOM(I).
The main difference between the methods is, that FOM(I) produces the exact solution in general only after nk steps, while FOM(II), due to the exact representation of B, needs only n/k steps to bring A on upper Hessenberg form. Note that much depends on the rank of the right-hand side matrix. In all our experiments, we took it full rank. If it is not full rank, FOM(II) runs into problems because it produces a rank deficient Krylov basis.

Acknowledgments

The research leading to this note has been made possible through a Fellowship of the Royal Netherlands Academy of Arts and Sciences (KNAW). The support of KNAW is gratefully acknowledged.

References

[1] R.H. Bartels and G.W Stewart, *Solution of the equation* $AX + XB = C$, Comm. ACM 15, pp. 820–826, 1972.

[2] S. Bittanti, A.J. Laub, and J.C. Willems (Eds.)(1991). *The Riccati Equation, Communications and Control Engineering Series*, Springer-Verlag, Berlin.

[3] J.H. Brandts (2000). A Riccati Algorithm for Eigenvalues and Invariant Subspaces, *Preprint nr. 1150 of the Department of Mathematics, Utrecht University, Netherlands.*

[4] G.H. Golub and C.F. van Loan (1996). *Matrix Computations (third edition)*, The John Hopkins University Press, Baltimore and London.

[5] G.H. Golub, S. Nash and C. F. van Loan (1979). A Hessenberg-Schur method for the problem $AX - XB = C$. *IEEE Trans. Automat. Control.*, AC-24, pp. 909-913.

[6] N.J. Higham (1993). Perturbation theory and backward error for $AX - XB = C$, *BIT*, 33:124–136.

[7] D.Y. Hu and L. Reichel (1992). Krylov subspace methods for the Sylvester equations, *Linear Algebra Appl.*, 172:283–314.

[8] V. Simoncini (1996) On the numerical solution of $AX - XB = C$, *BIT*, 36(4):182–198.

[9] V. Simoncini and M. Sadkane (1996) Arnoldi-Riccati method for large eigenvalue problems, *BIT*, 36(3):579–594.

[10] G.L.G. Sleijpen and H.A. van der Vorst, *A Jacobi-Davidson iteration method for linear eigenvalue problems*, SIAM J. Matrix Anal. Applic. 17, pp. 401–425, 1996.

[11] E. de Souza and S.P. Bhattacharyya (1981). Controllability, observability and the solution of $AX - XB = C$, *Linear Algebra Appl.*, 39:167–188.

[12] G. Starke and W. Niethammer (1991). SOR for $AX - XB = C$, *Linear Algebra Appl.*, 154–156:355-375.

[13] G.W. Stewart (1973). Error and perturbation bounds for subspaces associated with certain eigenvalue problems, *SIAM Review*, Vol. 15(4).

[14] G.W Stewart and J.G. Sun (1990). *Matrix Perturbation Theory*, Academic Press, London.