On the ADI Method for Sylvester Equations

Peter Benner

Mathematics in Industry and Technology, Fakultät für Mathematik, TU Chemnitz, 09107 Chemnitz, Germany

Ren-Cang Li [1]

Department of Mathematics, University of Texas at Arlington, P.O. Box 19408, Arlington, TX 76019-0408, USA

Ninoslav Truhar [2]

Department of Mathematics, J.J. Strossmayer University of Osijek, Trg Ljudevita Gaja 6, 31000 Osijek, Croatia

Abstract

This paper is concerned with the numerical solution of large scale Sylvester equations $AX - XB = C$, Lyapunov equations as a special case in particular included, with C having very small rank. For stable Lyapunov equations, Penzl (2000) and Li and White (2002) demonstrated that the so called Cholesky factor ADI method with decent shift parameters can be very effective. In this paper we present a generalization of the Cholesky factor ADI method for Sylvester equations. ***An easily implementable extension of Penz's shift strategy for the Lapunov equation is presented for the current case. It is demonstrated that Galerkin projection via ADI subspaces often produces much more accurate solutions than ADI solutions.***

Key words: Sylvester equation, factored ADI method, Galerkin projection
15A24, 15A90, 40C05, 37L99

1 Introduction

An $m \times n$ Sylvester equation takes the form

$$AX - XB = C, \tag{1.1}$$

where A, B, and C are $m \times m$, $n \times n$, and $m \times n$, respectively, and the unknown matrix X is $m \times n$. A Lyapunov equation is a special case with $m = n$, $B = -A^*$, and $C = C^*$, where the star superscript denotes complex conjugation and transposition. Equation (1.1) has a unique solution if and only if A and B have no common eigenvalues, see, e.g., [29], which will be assumed throughout this paper.

Sylvester equations appear frequently in many areas of applied mathematics, both theoretically and practically. We refer the reader to the elegant survey by Bhatia and Rosenthal [10] and references therein for a history of the equation and many interesting and important theoretical results. Sylvester equations play vital roles in a number of applications such as matrix eigendecompositions [25], control theory [13], model reduction [1, 4, 39], numerical solution of matrix differential Riccati equations [22], image processing [11], and many more.

This paper is concerned with the numerical solution of Sylvester equations. Lyapunov equations as a special case are also discussed. There are several numerical algorithms for that purpose. The standard ones are the Bartels-Stewart algorithm [3] and the Hessenberg-Schur method first described by Enright [22], but more often attributed to Golub, Nash, and Van Loan [24]. All these methods are efficient for dense matrices A and B. However, recent interest is directed more towards large and sparse matrices A and B, and $C = GF^*$ with very low rank, where G and F have only a few columns. Applications in which the constant term C naturally appears in factorized form range from optimal control [5] and image restoration [11] to model reduction based on the cross-Gramian approach [1, 4, 39]. In these cases, the standard methods are often too expensive to be practical, and iterative methods become more viable choices. Common methods for sparse A, B are Krylov subspace based algorithms [2, 20, 27, 28, 38] and Alternating-Directional-Implicit (ADI)

Email addresses: benner@mathematik.tu-chemnitz.de (Peter Benner), rcli@uta.edu (Ren-Cang Li), ntruhar@mathos.hr (Ninoslav Truhar).

[1] Supported in part by the National Science Foundation under Grant No. DMS-0510664 and DMS-0702335

[2] Supported in part by the National Science Foundation under Grant No. 235-2352818-1042. Part of this work was done while this author was a visiting professor at the Department of Mathematics, University of Texas at Arlington, Arlington, TX, USA.

iterations [7, 26, 30, 32, 33, 35, 41]. Advantages of Krylov subspace based algorithms over ADI iterations are that no knowledge about the spectra of A and B is needed and (except for [38]) no linear systems of equations with (shifted) A and B have to be solved. But ADI iterations often enable faster convergence if (sub)optimal shifts to A and B can be effectively estimated. So for a problem for which linear systems with shifted A and B can be solved at modest cost, ADI iterations may turn out to be better alternatives. This is often true for stable Lyapunov equations from control theory [7, 26, 30, 33].

Recently, Ding and Chen proposed a few simple iterative schemes for matrix equations in [17, 18, 19] (and others therein). The schemes, resembling the classical Jacobi and Gaussian iterations for linear systems, are easy to implement and cost little per step but converge linearly at the best. *It has been considered in [18, 19, 23] that the gradient-based iterative (GI) algorithms and least squares based iterative algorithms [20, 23] for solving (coupled) matrix equations are novel and highly efficient algorithms based on the hierarchical identification principle [15, 16] which regards the unknown matrix as the system parameter matrix to be identified. These (coupled) matrix equations include the Lyapunov matrix equations and Sylvester matrx equations as special cases.* ADI iterations have a different objective: achieving fast convergence rate through exploiting matrices' spectrum information.

In this paper, we shall first extend the Cholesky factor ADI for Lyapunov equations to solve Sylvester equations based on previous work in [6, 40, 43]. Then, we argue that often much more accurate solutions than the ADI solutions can be obtained by performing a Galerkin-type projection via the row and column subspaces of the computed solutions. The improvement is often more drastic with poor shifts. Indeed, in the absence of knowledge of the spectra, currently there is no provable way to select good shifts, and existing practices like [8, 33] are more heuristic than rigorously justifiable, except for stable Lyapunov equations with Hermitian A [21, 42, 31] and for Sylvester equations with Hermitian A and B [36][3].

The rest of this paper is organized as follows. Section 2 reviews the ADI method and derives factored ADI iterations for Sylvester equations. An extension of Penzl's shift strategy to Sylvester equations is explained in Section 3. Projection ADI subspace methods via Galerkin projection or the minimal residual condition are presented in Section 4. Section 5 explains the connection between the new algorithm and Cholesky factor ADI for Lyapunov equations. We report several numerical tests in Section 6 and finally present our conclusions in Section 7.

Notation. Throughout this paper, $\mathbb{C}^{n\times m}$ is the set of all $n \times m$ complex

[3] The parameters for the case were also made available by E. Wachspress in 2000

matrices, $\mathbb{C}^n = \mathbb{C}^{n\times 1}$, and $\mathbb{C} = \mathbb{C}^1$. Similarly define $\mathbb{R}^{n\times m}$, $\mathbb{R}^n$, and $\mathbb{R}$ except replacing the word *complex* by *real.* I_n (or simply I if its dimension is clear from the context) is the $n \times n$ identity matrix, and e_j is its jth column. The superscript "$\cdot^*$" denotes conjugate transposition while "$\cdot^{\mathrm{T}}$" stands for transposition only. For scalars, $\bar{\alpha}$ is the complex conjugate of α, and $\Re(\alpha)$ takes the real part of α. We shall also adopt MATLAB-like convention to access the entries of vectors and matrices. $i : j$ is the set of integers from i to j inclusive and $i : i = \{i\}$. For a vector u and a matrix X, $u_{(j)}$ is u's jth entry, $X_{(i,j)}$ is X's (i, j)th entry; X's submatrices $X_{(k:\ell,i:j)}$, $X_{(k:\ell,:)}$, and $X_{(:,i:j)}$ consist of intersections of row k to row ℓ and column i to column j, row k to row ℓ, and column i to column j, respectively.

2 ADI for Sylvester Equations

As it has been shown in [43] (more detailed in [40]) for given two sets of parameters $\{\alpha_i\}$ and $\{\beta_i\}$, the factored Alternating-Directional-Implicit (fADI) iteration for iteratively solving (1.1) proceeds as follows:

For $k = 0, 1, \ldots,$

$$Z_k = \left(Z^{(1)}\ Z^{(2)}\ \cdots\ Z^{(k)}\right), \\ \text{with} \begin{cases} Z^{(1)} = (A - \beta_1 I)^{-1}G, \\ Z^{(i+1)} = (A - \alpha_i I)(A - \beta_{i+1} I)^{-1}Z^{(i)} \\ \qquad\quad = Z^{(i)} + (\beta_{i+1} - \alpha_i)(A - \beta_{i+1} I)^{-1}Z^{(i)}, \end{cases} \tag{2.1}$$

and

$$Y_k = \left(Y^{(1)}\ Y^{(2)}\ \cdots\ Y^{(k)}\right), \\ \text{with} \begin{cases} Y^{(1)^*} = F^*(B - \alpha_1 I)^{-1}, \\ Y^{(i+1)^*} = Y^{(i)^*}(B - \alpha_{i+1} I)^{-1}(B - \beta_i I) \\ \qquad\quad = Y^{(i)^*} + (\alpha_{i+1} - \beta_i)Y^{(i)^*}(B - \alpha_{i+1} I)^{-1}, \end{cases} \tag{2.2}$$

and

$$X_k = Z_k D_k Y_k^*, \quad D_k = \operatorname{diag}\left((\beta_1 - \alpha_1)I_r, \ldots, (\beta_k - \alpha_k)I_r\right). \tag{2.3}$$

Formulas (2.1) – (2.3) yields a new fADI which is a natural extension of CF-ADI [30] and LR-ADI [33, 35] for stable Lyapunov equations.

Algorithm 1 (fADI for Sylvester equation $AX - XB = GF^*$)

```
Input:  (a) $A_{(m\times m)}$, $B_{(n\times n)}$, $G_{(m\times r)}$, and $F_{(n\times r)}$;
        (b) ADI shifts $\{\beta_1, \beta_2, \ldots\}$, $\{\alpha_1, \alpha_2, \ldots\}$;
        (c) $k$, the number of ADI steps;
Output: $Z_{(m\times kr)}$, $D_{(kr\times kr)}$, and $Y_{(n\times kr)}$ such that $ZDY^*$ approximately
        solves Sylvester equation $AX - XB = GF^*$;
    $Z_{(:,1:r)} = (A - \beta_1 I)^{-1}G$; $(Y^*)_{(1:r,:)} = F^*(B - \alpha_1 I)^{-1}$;
    for $i = 1, 2, \ldots, k$ do
        $Z_{(:,ir+1:(i+1)r)} = Z_{(:,(i-1)r+1:ir)}$
                $+(\beta_{i+1} - \alpha_i)(A - \beta_{i+1}I)^{-1}Z_{(:,(i-1)r+1:ir)}$;
        $(Y^*)_{(ir+1:(i+1)r,:)} = (Y^*)_{((i-1)r+1:ir),:)}$
                $+(\alpha_{i+1} - \beta_i)\,(Y^*)_{((i-1)r+1:ir),:)}\,(B - \alpha_{i+1}I)^{-1}$;
    end for;
    $D = \text{diag}\,((\beta_1 - \alpha_1)I_r, \ldots, (\beta_k - \alpha_k)I_r)$.
```

Remark 1 For general dense A and B, Algorithm 1 as is is not appealing computationally because a linear system with a shifted A costs $O(m^3)$ while a linear system with a shifted B costs $O(n^3)$. That makes it no better than, e.g., the standard Bartels-Stewart algorithm [3] or the Hessenberg-Schur method [22, 24]. For large m and n which is the focus of this article, often A and B either are very sparse, meaning only a small percentage of their entries are nonzero, or have certain structures, e.g., narrow banded, so that the costs for solving the linear systems may likely be $O(m^2)$ and $O(n^2)$, or sometimes even less $O(m)$ and $O(n)$. For modest m and n, some pre-processing can be done to bring down the cost of each fADI step to $O(m^2 + n^2)$ by first performing Hessenberg reductions on A and B, a standard practice for computing the Schur form of a nonsymmetric matrix [14]. This is done via unitary transformations and thus stable. As Wachspress pointed out to the authors, it is also possible to reduce A and B to very narrow banded matrices by general similarity transformations but care must be taken to monitor the conditioning of the transformation matrices.

We also note that the Z- and Y-factors in Algorithm 1 can be computed in parallel.

3 A shift strategy

ADI shifts determine the speed of the convergence of the method. There are a number of strategies out there, and most of them are based on heuristic arguments, except in the Hermitian cases. In his thesis, Sabino [36] presented a quite complete review of the existing strategies. Since this paper, however, is not about looking for yet another shift strategy, for testing purpose we shall simply discuss an easily implementable extension of Penzl's [33, 35] who did it for Lyapunov equations.

As it has been emphasize in the introduction, our approach here is based on the following idea: try to solve Sylvester equation using any two sets of ADI parameters (we propose extension of Penzl's shifts) and then improve the obtained solution by performing a Galerkin-type projection via the row and column subspaces of the computed solutions.

When A and B are Hermitian (this is in fact true for normal A and B), the optimal ADI parameters are solutions of the following ADI minimax problem for k ADI steps with $\mathbb{E} = \mathsf{eig}(A)$ and $\mathbb{F} = \mathsf{eig}(B)$, where $\mathsf{eig}(\cdot)$ denotes the spectrum of a matrix.

Find α_j and β_j, $j = 1, \ldots, k$, such that

$$\min_{\substack{\alpha_i \in \mathbb{C} \\ \beta_j \in \mathbb{C}}} \max_{\substack{x \in \mathbb{E} \\ y \in \mathbb{F}}} \prod_{j=1}^{k} \left| \frac{(x-\alpha_j)(y-\beta_j)}{(x-\beta_j)(y-\alpha_j)} \right|. \tag{3.1}$$

In practice since $\mathsf{eig}(A)$ and $\mathsf{eig}(B)$ are not known a *priori*, $\mathbb{E}$ and $\mathbb{F}$ are often replaced by intervals that contain the eigenvalues of A and B, respectively. In the case for Lyapunov equations, $B = -A^*$, $\beta_j = -\bar{\alpha}_j$ (the complex conjugate of α_j), and $\mathbb{F} = -\mathbb{E}$, Problem (3.1) reduces to

Find α_j, $j = 1, \ldots, k$, such that

$$\min_{\alpha_i \in \mathbb{C}} \max_{x \in \mathbb{E}} \prod_{j=1}^{k} \left| \frac{x-\alpha_j}{x+\bar{\alpha}_j} \right|. \tag{3.2}$$

Regardless of whether A is Hermitian or not, for stable Lyapunov equations Penzl [35] proposed a heuristic shift-selection strategy by solving a much simplified (3.2): Find α_j, $j = 1, \ldots, k$, such that

$$\min_{\alpha_i \in \mathbb{E}} \max_{x \in \mathbb{E}} \prod_{j=1}^{k} \left| \frac{x-\alpha_j}{x+\bar{\alpha}_j} \right| \tag{3.3}$$

with $\mathbb{E}$ set to be a collection of certain estimates of the extreme eigenvalues of A. The strategy usually works very well. In obtaining $\mathbb{E}$, Penzl proposed to run a pair of Arnoldi processes. The first process delivers k_+ Ritz values that tend to approximate well "outer" eigenvalues, which are generally not close to the origin. The second process is used to get k_- Ritz values to approximate those eigenvalues near the origin. His algorithm then chooses a set of shift parameters out of $\mathbb{E}$ by solving (3.3). The shifts delivered by the heuristic are ordered in such a way that shifts, which should reduce the ADI error most, are applied first.

Penzl's strategy can be naturally extended to the case for Sylvester equations. Now we need to compute two sets $\{\alpha_1, \dots, \alpha_k\}$ and $\{\beta_1, \dots, \beta_k\}$ of presumed good shift parameters. We start by generating two discrete sets $\mathbb{E}$ and $\mathbb{F}$ which "well" approximates parts of the spectra of A and B, respectively, and then solve a much simplified (3.1): Find α_j and β_j, $j = 1, \dots, k$, such that

$$\min_{\substack{\alpha_i \in \mathbb{E} \\ \beta_j \in \mathbb{F}}} \max_{\substack{x \in \mathbb{E} \\ y \in \mathbb{F}}} \prod_{j=1}^{k} \left| \frac{(x-\alpha_j)(y-\beta_j)}{(x-\beta_j)(y-\alpha_j)} \right|. \tag{3.4}$$

Again the selected shifts are ordered in such a way that shifts, which should reduce the ADI error most, are applied first. This is summarized in Algorithm 2.

Algorithm 2 (ADI parameters by Ritz values (ADIpR))

```
Input: $A$, $F$, $B$, $G$, $k$;
Output: ADI parameters $\{\alpha_1, \dots, \alpha_k\}$ and $\{\beta_1, \dots, \beta_k\}$;
    Run Arnoldi process with $A$ on $G$ to give the set $\mathbb{E}_A^+$ of Ritz values;
    Run Arnoldi process with $A^{-1}$ on $G$ to give the set $\mathbb{E}_A^-$ of Ritz values;
    $\mathbb{E} = \mathbb{E}_A^+ \cup (1/\mathbb{E}_A^-)$;
    Run Arnoldi process with $B^*$ on $F$ to give the set $\mathbb{F}_B^+$ of Ritz values;
    Run Arnoldi process with $B^{-*}$ on $F$ to give the set $\mathbb{F}_B^-$ of Ritz values;
    $\mathbb{F} = \mathrm{conj}(\mathbb{F}_B^+) \cup \mathrm{conj}(1/\mathbb{F}_B^-)$;
    Set $\{\alpha_1, \beta_1\} = \arg\min_{\substack{\alpha \in \mathbb{E} \\ \beta \in \mathbb{F}}} \max_{\substack{x \in \mathbb{E} \\ y \in \mathbb{F}}} \left| \frac{(x-\alpha)(y-\beta)}{(x-\beta)(y-\alpha)} \right|$;
    For $i = 2, \dots, k$ do
        Set $\{\alpha_i, \beta_i\} = \arg\min_{\substack{\alpha \in \mathbb{E}' \\ \beta \in \mathbb{F}'}} \max_{\substack{x \in \mathbb{E} \\ y \in \mathbb{F}}} \left| \frac{(x-\alpha)(y-\beta)}{(x-\beta)(y-\alpha)} \right| \prod_{j=1}^{i-1} \left| \frac{(x-\alpha_j)(y-\beta_j)}{(x-\beta_j)(y-\alpha_j)} \right|$,
        where $\mathbb{E}'$ is $\mathbb{E}$ with $\alpha_1, \dots, \alpha_{i-1}$ deleted, and similarly for $\mathbb{F}'$;
    EndDo.
```

For more details about its efficient implementation, the reader is referred to [40].

Remark 2 Recently, Wachspress in [44] improved spectral alignment. He showed that the crucial values for determination of a proper set of ADI parameters are the minimum real part, the maximum real part, and the maximum angle for each spectrum. This can be done with the precise knowledge of the spectra. On the other hand as we have already mentioned in the previous remark it is possible, for modest m and n, to reduce A and B to very narrow banded matrices by general similarity transformations but care must be taken to monitor the conditioning of the transformation matrices. Since this reduction enhances accurate eigenvalue estimation with the aid of double-implicit

LR reduction with bounded pivots to maintain low bandwidth. The LR approach could replace Arnoldi in algorithm 2 with shifts chosen to reveal the crucial eigenvalues. This could be interesting problem for further studies.

4 Projection ADI Subspace Methods for Sylvester Equation

Given parameters $\{\alpha_i\}$ and $\{\beta_i\}$, we define the kth *ADI column subspace* to be the column space of the kth ADI solution $X_k = Z_k D_k Y_k^*$ and the kth *ADI row subspace* to be the row space of X_k^*. Equivalently the kth ADI column subspace is the column space of Z_k, and the kth ADI row subspace is the row space of Y_k^*.

Our numerical experiments strongly suggest often these ADI subspaces are quite good in the sense that the ADI column subspaces come very close to the column space of X, the exact solution, and ADI row subspaces come very close to the row space of X. This is true even for not so good parameters $\{\alpha_i\}$ and $\{\beta_i\}$. Our numerical experiments also suggest that *one* single poor shift can effectively offset all previous good shifts and thus degrade ADI approximations enormously for the next many iterations.

Given that it is so hard to select optimal, sometime even decent, parameters in general for ADI solutions to be any good, perhaps we should seek instead solutions having form

$$\widetilde{X_k} = U_k W_k V_k^* \tag{4.1}$$

under the Galerkin condition or the minimal residual condition, where U_k has the same columns space as Z_k (X_k) and V_k^* has the same row space as Y_k^* (X_k). We call a method as such a *projection ADI subspace method.* Since Z_k and Y_k are computed one block at a time by Algorithm 1 from the very previous blocks, U_k and V_k can be computed along the way, by, e.g., the (modified) Gram-Schmidt orthogonalization process as soon as a new Z-block or Y-block becomes available. Doing so leads to U_k and V_k with orthoonormal columns.

The idea of using Galerkin projection or minimizing the residual is not new. What that *is* new here is our choices of projection subspaces. Previously it was used when columns of U_k and V_k span a Krylov subspace of A on G and a Krylov subspace of B^* on F^*, respectively [27, 28, 37] and more recently for Lyapunov equations for which $U_k = V_k$ spans a direct sum of Krylov subspaces of A on G and A^{-1} on G [38].

4.1 Galerkin projection

Suppose U_k and V_k have orthonormal columns. Let residual $R_k = A\widetilde{X}_k - \widetilde{X}_k B - C$ for an approximation solution $\widetilde{X}_k$. The Galerkin condition enforces $U_k^* R_k V_k = 0$. Thus

$$(U_k^* A U_k) W_k - W_k (V_k^* B V_k) = U_k^* C V_k \tag{4.2}$$

which is a Sylvester equation but of a much smaller size and can be solved by, e.g., Bartel-Stewart algorithm [3], or Golub-Nash-Van Loan algorithm [24].

Note that U_k and V_k do not necessarily have to have the same number of columns. When they don't, W_k will not be a square matrix.

4.2 A minimal residual method

The minimal residual condition requires to solve

$$\min_{W_k} \|R_k\|_{\rm F} \equiv \min_{W_k} \|A(U_k W_k V_k^*) - (U_k W_k V_k^*)B - C\|_{\rm F}\,. \tag{4.3}$$

It turns out going from the simple Galerkin projection to this minimal residual condition is utterly nontrivial computationally. The novel idea due to Hu and Reichel [27, p.293] can be modified to work, thanks to Theorem 4.1 below. But the amount of increased work makes it less attractive. Nevertheless, we still present Theorem 4.1 which may be of independent interest of its own right. Adopt the notation of Section 2 in its entirety. By (2.1) and (2.2), the kth ADI column and row spaces are

$$\mathscr{C}_k \stackrel{\rm def}{=} \mathsf{colspan}\{Z^{(1)}, Z^{(2)}, \ldots, Z^{(k)}\}, \quad \mathscr{R}_k \stackrel{\rm def}{=} \mathsf{rowspan}\{{Y^{(1)}}^*, {Y^{(2)}}^*, \ldots, {Y^{(k)}}^*\},$$

respectively.

Theorem 4.1 *We have for $i \geq 1$*

$$AZ^{(i)} = G + \prod_{j=1}^{i-1}(\beta_j - \alpha_j)Z^{(j)} + \beta_i Z^{(i)}, \tag{4.4}$$

$${Y^{(i)}}^* B = F^* + \prod_{j=1}^{i-1}(\alpha_j - \beta_j){Y^{(j)}}^* + \alpha_i {Y^{(i)}}^*, \tag{4.5}$$

where $\prod_{j=1}^{0}(\cdots)$ is taken to be 0. Therefore

$$A\mathscr{C}_k \subseteq \mathsf{colspan}\{G, Z^{(1)}, Z^{(2)}, \ldots, Z^{(k)}\} = \mathsf{colspan}\{G\} + \mathscr{C}_k, \tag{4.6}$$

$$\mathscr{R}_k B \subseteq \mathsf{rowspan}\{F^*, {Y^{(1)}}^*, {Y^{(2)}}^*, \ldots, {Y^{(k)}}^*\} = \mathsf{rowspan}\{F^*\} + \mathscr{R}_k. \tag{4.7}$$

Proof By (2.1), we have

$$\begin{aligned}
AZ^{(1)} &= A(A-\beta_1 I)^{-1}G \\
&= G + \beta_1(A-\beta_1 I)^{-1}G \\
&= G + \beta_1 Z^{(1)}, \\
AZ^{(i+1)} &= AZ^{(i)} + (\beta_{i+1}-\alpha_i)A(A-\beta_{i+1}I)^{-1}Z^{(i)} \\
&= AZ^{(i)} + (\beta_{i+1}-\alpha_i)Z^{(i)} + (\beta_{i+1}-\alpha_i)\beta_{i+1}(A-\beta_{i+1}I)^{-1}Z^{(i)} \\
&= AZ^{(i)} + (\beta_{i+1}-\alpha_i)Z^{(i)} + \beta_{i+1}(Z^{(i+1)} - Z^{(i)}) \\
&= AZ^{(i)} - \alpha_i Z^{(i)} + \beta_{i+1}Z^{(i+1)} \\
&= AZ^{(i-1)} - \alpha_{i-1}Z^{(i-1)} + \beta_i Z^{(i)} - \alpha_i Z^{(i)} + \beta_{i+1}Z^{(i+1)} \\
&= \cdots \\
&= AZ^{(1)} - \alpha_1 Z^{(1)} + \prod_{j=2}^{i}(\beta_j - \alpha_j)Z^{(j)} + \beta_{i+1}Z^{(i+1)} \\
&= G + \prod_{j=1}^{i}(\beta_j - \alpha_j)Z^{(j)} + \beta_{i+1}Z^{(i+1)}
\end{aligned}$$

which proves (4.4). Similarly

$$\begin{aligned}
{Y^{(1)}}^* B &= F^*(B-\alpha_1 I)^{-1}B \\
&= F^* + \alpha_1 F^*(B-\alpha_1 I)^{-1} \\
&= F^* + \alpha_1 {Y^{(1)}}^*, \\
{Y^{(i+1)}}^* B &= {Y^{(i)}}^* B + (\alpha_{i+1}-\beta_i){Y^{(i)}}^*(B-\alpha_{i+1}I)^{-1}B \\
&= {Y^{(i)}}^* B + (\alpha_{i+1}-\beta_i){Y^{(i)}}^* + (\alpha_{i+1}-\beta_i)\alpha_{i+1}{Y^{(i)}}^*(B-\alpha_{i+1}I)^{-1} \\
&= {Y^{(i)}}^* B + (\alpha_{i+1}-\beta_i){Y^{(i)}}^* + \alpha_{i+1}({Y^{(i+1)}}^* - {Y^{(i)}}^*) \\
&= {Y^{(i)}}^* B - \beta_i {Y^{(i)}}^* + \alpha_{i+1}{Y^{(i+1)}}^* \\
&= F^* + \prod_{j=1}^{i}(\alpha_j - \beta_j){Y^{(j)}}^* + \alpha_{i+1}{Y^{(i+1)}}^*
\end{aligned}$$

which proves (4.5). ∎

The objective function in (4.3) is the Frobenius norm of an $m \times n$ matrix. Recall that the Sylvester equations we are interested have large m and n. Potentially (4.3) is as difficult as the original equation itself. Using the results of Theorem 4.1, we can reduced the size of the problem to at most $(k+1)r \times (k+1)r$. We shall now explain how. Assume that both U_k and V_k have orthonormal columns to begin with. Now orthogonalize G against the columns of U_k, and F against the columns of V_k to get $(U_k, \widehat{U})$ and $(V_k, \widehat{V})$, both having orthonormal columns. It can be seen that both $\widehat{U}$ and $\widehat{V}$ have no more than r

columns. Theorem 4.1 implies

$$AU_k = (U_k, \widehat{U})A_k, \quad V_k^* B = B_k(V_k, \widehat{V})^*$$

for some matrices A_k and B_k. Then

$$\begin{aligned} &A(U_k W_k V_k^*) - (U_k W_k V_k^*)B - C \\ &= (U_k, \widehat{U})A_k W_k V_k^* - U_k W_k B_k (V_k, \widehat{V})^* - C \end{aligned}$$

from which one can see that the solution W_k of (4.3) is the same as that of

$$\min_{W_k} \left\| A_k W_k (I, 0) - \begin{pmatrix} I \\ 0 \end{pmatrix} W_k B_k - (U_k, \widehat{U})^* G [(V_k, \widehat{V})^* F]^* \right\|_{\mathrm{F}}, \tag{4.8}$$

a much smaller problem than (4.3). This problem can be solved by borrowing the idea of Hu and Reichel [27, p.293]. But still its cost of doing so is much higher than solving (4.2) as the result of the simple Galerkin projection, nonetheless.

5 Application to Lyapunov equation

fADI in Section 2 is a natural extension of the LR-CF ADI [30, 33, 35] for the Lyapunov Equation

$$AX + XA^* = C, \tag{5.1}$$

where A, C, and unknown X are all $n \times n$, and C is Hermitian. Since, Lyapunov Equation (5.1) is a special case of Sylvester equation (1.1) with $B = -A^*$, previous developments apply upon substituting $B = -A^*$ and $\beta_i = -\bar{\alpha}_i$, and most expressions can be much simplified, too.

In the case of Lyapunov equation it holds $Y^{(k)} = Z^{(k)}$, thus instead of (2.1) and (2.2) we have

$$\begin{aligned} &Z_k = \left(Z^{(1)}\ Z^{(2)}\ \cdots\ Z^{(k)} \right), \\ &\text{with} \quad \begin{cases} Z^{(1)} &= (A + \bar{\alpha}_1 I)^{-1} G, \\ Z^{(i+1)} &= (A - \alpha_i I)(A + \bar{\alpha}_{i+1} I)^{-1} Z^{(i)} \\ &= Z^{(i)} - (\bar{\alpha}_{i+1} + \alpha_i)(A + \bar{\alpha}_{i+1} I)^{-1} Z^{(i)}, \end{cases} \end{aligned} \tag{5.2}$$

while (2.3) is given by

$$X_k = Z_k D_k Z_k^*, \quad D_k = -2\,\mathrm{diag}\left(\Re(\alpha_1) I_r, \ldots, \Re(\alpha_k) I_r \right). \tag{5.3}$$

Based on (5.2) and (5.3), an fADI for Lyapunov equation $AX+XA^*+GG^*=0$ is obtained as in Algorithm 3.

Algorithm 3 (fADI for Lyapunov equation $AX+XA^*+GG^*=0$)

```
Input:  (a) $A_{(m\times m)}$, and $G_{(m\times r)}$;
        (b) ADI shifts $\{\alpha_1, \alpha_2, \ldots\}$;
        (c) $k$, the number of ADI steps;
Output: $Z_{(m\times kr)}$ and $D_{(kr\times kr)}$ such that $ZDZ^*$ approximately
        solves Lyapunov equation $AX+XA^*+GG^*=0$;
        $Z_{(:,1:r)} = (A+\bar{\alpha}_1 I)^{-1}G$;
        for $i=1,2,\ldots,k$ do
            $Z_{(:,ir+1:(i+1)r)} = Z_{(:,(i-1)r+1:ir)} - (\bar{\alpha}_{i+1}+\alpha_i)(A+\bar{\alpha}_{i+1})^{-1}Z_{(:,(i-1)r+1:ir)}$;
        end for;
        $D = -2\,\mathrm{diag}\,(\Re(\alpha_1)I_r, \ldots, \Re(\alpha_k)I_r)$.
```

For a stable Lyapunov equation, this essentially gives the so-called Cholesky Factor ADI (CF-ADI) of Li and White [30] and Low Rank ADI of Penzl [33], except that in CF-ADI/LR-ADI matrices D_i are embedded into Z_i. The difference is that here we have a matrix D. An advantage of doing so is that the algorithm no longer requires all $\Re(\alpha_i) > 0$, as must have by [30, 33]. Thus Algorithm 3 has a larger domain of applicability than its earlier versions.

6 Numerical Experiments

In this section, we shall report several numerical examples to demonstrate that the Galerkin projection via ADI subspaces can lead to more accurate solutions than ADI alone.

Example 6.1 This is essentially [9, Example 1], except for C which will be set to some random rank-1 matrix. Depending on parameters a, b, s and the dimension n, matrices A, B, and C are generated as follows. First, set

$$\begin{aligned}
\widehat{A} &= \mathrm{diag}(-1, -a, -a^2, \ldots, -a^{n-1}),\\
\widehat{B} &= \mathrm{diag}(1, b, b^2, \ldots, b^{n-1}),\\
\widehat{C} &= \widehat{G}\widehat{F}^*,
\end{aligned}$$

where $\widehat{G}$ and $\widehat{F}$ are $n\times 1$ and generated randomly as by `randn`$(n,1)$ in MATLAB. Parameters a and b regulate the distribution of the spectra of A and B, respectively, and therefore their separation. The entries of the solution matrix

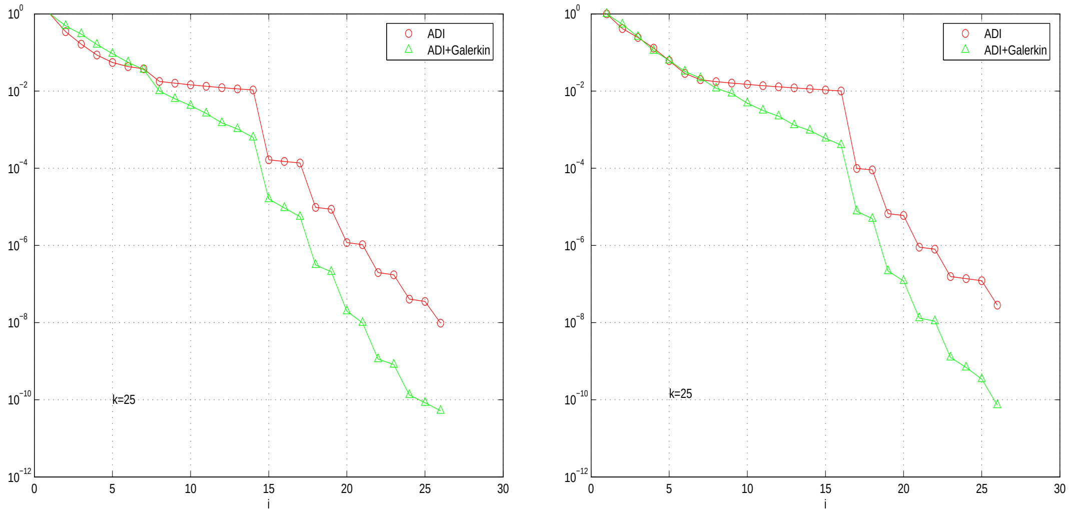

Fig. 6.1. Relative residual errors for ADI solutions and solutions by Galerkin projection via ADI subspaces for two different runs for Example 6.1 with (6.1).

to $\widehat{A}\widehat{X} - \widehat{X}\widehat{B} = \widehat{C}$ are then given by

$$\widehat{X}_{(i,j)} = \frac{\widehat{C}_{(i,j)}}{\widehat{A}_{(i,i)} - \widehat{B}_{(j,j)}}.$$

Next we employ a transformation matrix to define

$$A = T^{-T}\widehat{A}T^{\mathrm{T}},\ B = T\widehat{B}T^{-1},\ G = T^{-T}\widehat{G},\ F = \widehat{F}T^{-1},$$

where $T = H_2 S H_1 \in \mathbb{C}^{n\times n}$ is defined through

$$\begin{aligned} H_1 &= I_n - \frac{2}{n}h_1 h_1^{\mathrm{T}}, \quad h_1 = (1, 1, \ldots, 1)^{\mathrm{T}}, \\ H_2 &= I_n - \frac{2}{n}h_2 h_2^{\mathrm{T}}, \quad h_2 = (1, -1, \ldots, (-1)^{n-1})^{\mathrm{T}}, \\ S &= \mathrm{diag}(1, s, \ldots, s^{n-1}). \end{aligned}$$

The scalar s is used here to regulate the conditioning of T. Because of the way they are constructed, each linear system with shifted A or B costs $O(n)$ flops to solve. In all our tests reported here, $k = 25$ and $n = 500$. We tested Algorithms 1 and 2 on two sets of parameter values:

$$a = 1.03,\ b = 1.008,\ s = 1.001; \tag{6.1}$$
$$a = 1.03e^{\iota\theta},\ b = 1.008e^{\iota\theta},\ s = 1.001, \tag{6.2}$$

where $\iota = \sqrt{-1}$ and $\theta = \pi/(2n)$. The values given in (6.1) were the ones used in [9]. In applying Algorithm 2, each Arnoldi run takes 35 steps and 17 best

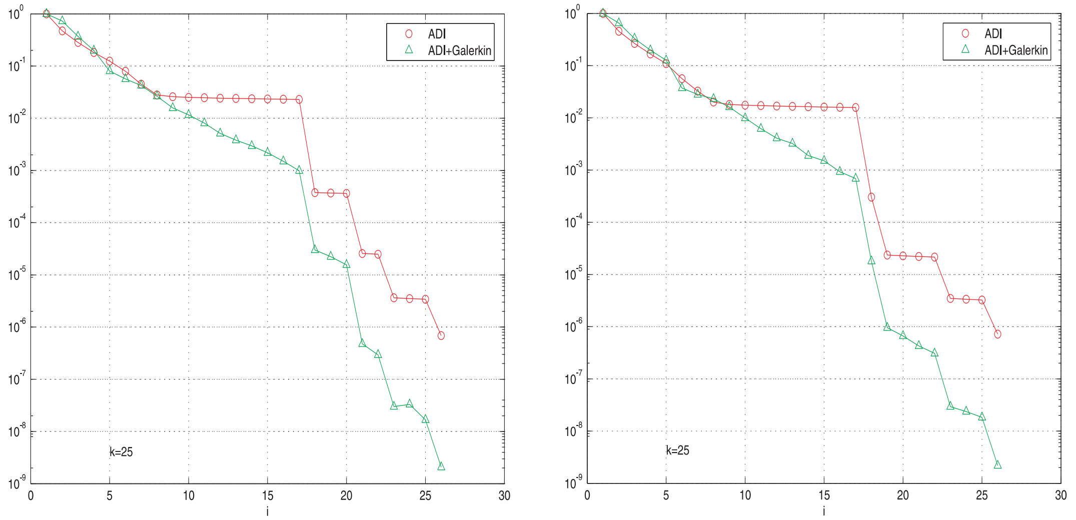

Fig. 6.2. Relative residual errors for ADI solutions and solutions by Galerkin projection via ADI subspaces for two different runs for Example 6.1 with (6.2).

Ritz values are taken, and thus both $\mathbb{E}$ and $\mathbb{F}$ have 34 values among which 25 are selected in the end. Our fADI produces an approximation $X_k = Z_k D_k Y_k^*$, along with intermediate approximations $X_i = Z_i D_i Y_i^*$ for $i \le k$. For two runs with different random F and G, Figures 6.1 and 6.2 plot the relative residual errors

$$\frac{\|AX_i - X_iB - GF^*\|_{\rm F}}{\|GF^*\|_{\rm F}}$$

(marked as "ADI"), as well as the relative residual errors for the approximations $U_iW_iV_i^*$ by Galerkin projection (4.2) (marked as "ADI+Galerkin"). We have run tests on each parameter set many times with different random F and G, and the residual behaviors are all similar to those plotted in Figures 6.1 and 6.2. In both figures, Galerkin projection via ADI subspaces produces better approximations after $i \ge 7$ and the improvements are up to more than 2 decimal digits. $\diamond$

Example 6.2 Chahlaoui and Van Dooren [12] compiled a collection of benchmark examples for model reduction. Except those for descriptor systems, these examples give rise to Lyapunov equations $AX + XA^* + GG^* = 0$. Simplified versions of Algorithms 1 and 2 upon substituting $B = -A^*$ and $F = -G^*$ can be applied. We have tested ADI and Galerkin projection via ADI subspaces on these equations, and found out both performs badly when A is highly non-normal in the sense that $\|A\|_{\rm F}^2$ and $\|A\|_{\rm F}^2 - \sum_j |\lambda_j|^2$ are of the same magnitude and $\sum_j |\lambda_j|^2 \ll \|A\|_{\rm F}^2$, where $\{\lambda_j\}$ consists of all A's eigenvalues. In the collection, there are two other examples whose A are in fact normal, i.e., $\|A\|_{\rm F}^2 = \sum_j |\lambda_j|^2$. We now report our numerical results on them. The first

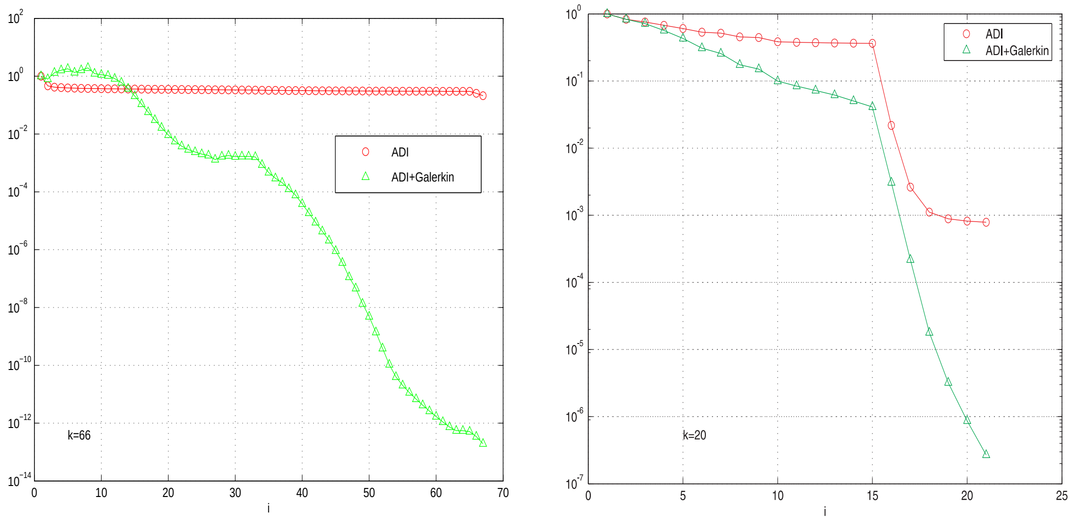

Fig. 6.3. Relative residual errors for ADI solutions and solutions by Galerkin projection via ADI subspaces. *Left:* FOM in [12]; *Right:* HEAT in [12].

example is FOM: $A = \operatorname{diag}(A_1, A_2, A_3, A_4)$ with

$$A_1 = \begin{pmatrix} -1 & 100 \\ -100 & -1 \end{pmatrix}, \; A_2 = \begin{pmatrix} -1 & 200 \\ -200 & -1 \end{pmatrix}, \; A_3 = \begin{pmatrix} -1 & 400 \\ -400 & -1 \end{pmatrix},$$

$A_4 = \operatorname{diag}(-1, -2, \ldots, -1000)$, and $G = (\underbrace{10, \ldots, 10}_{6}, \underbrace{1, \ldots, 1}_{1000})^{\mathrm{T}}$. So $n = 1006$ and each linear system with shifted A or A^* takes $O(n)$ flops to solve. Apply Algorithms 1 and 2 with $k = 66$, where for Algorithm 2, each Arnoldi run takes 76 steps and 38 best Ritz values are taken, and thus $\mathbb{E}$ has 76 values among which 66 are selected in the end. The left of Figure 6.3 plots the relative residual errors for ADI solutions and solutions by Galerkin projection via ADI subspaces, as explained in Example 6.1. For this example, ADI barely does anything while projection ADI subspace method does extremely well.

The other example is from discretizing the 1-D heat equation. A is real symmetric tridiagonal. Thus each linear system with shifted A or A^* takes $O(n)$ flops to solve. The size of A can be made as large as one wishes. As in [12], we take $n = 200$, and A has diagonal entries -808 and off-diagonal entries 404, and all of G's entries are zero, except $G_{(67)} = 1$. With $k = 20$ for Algorithm 2, each Arnoldi run takes 30 steps and 15 best Ritz values are taken, and thus $\mathbb{E}$ has 30 values among which 20 are selected in the end. The right of Figure 6.3 plots the relative residual errors for ADI solutions and for solutions by Galerkin projection via projection ADI subspaces method. $\diamond$

Example 6.3 This is a Sylvester equation $AX - XB = GF^*$ with real symmetric A and B, both taken from the Harwell-Boeing Collection. In fact,

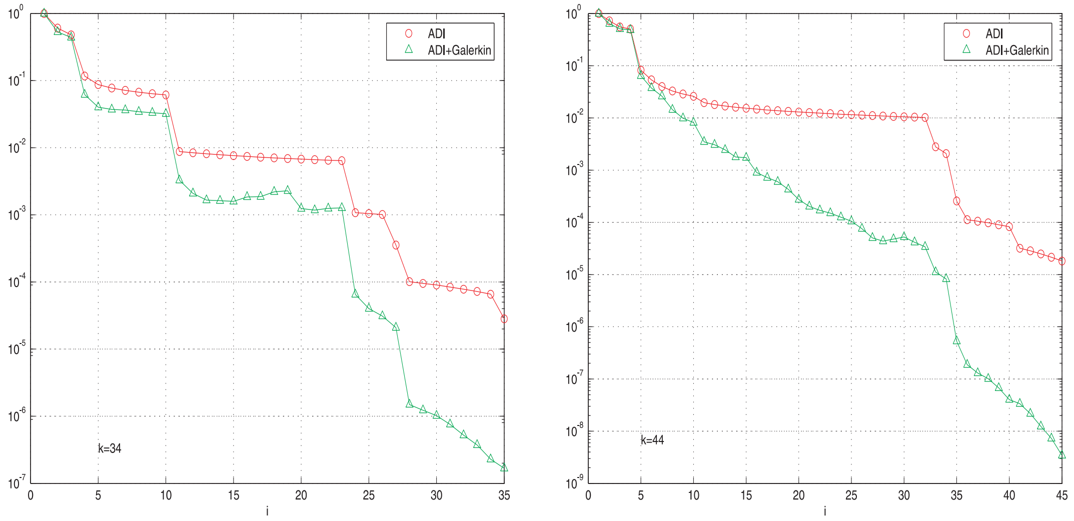

Fig. 6.4. Relative residual errors for ADI solutions and solutions by Galerkin projection via ADI subspaces for Example 6.3. *Left:* $k = 34$; *Right:* $k = 44$.

$A(675 \times 675)$ is NOS6 and $B(468 \times 468)$ is negative NOS5 from Set LANPRO[4]. G and F are taken to be random vectors. Sylvester equations so constructed are solely for our testing purpose because there is no physical background yet for combining the two matrices together in one Sylvester equation. Both A and B are sparse and in fact very narrow-banded, and each linear system with shifted A or B costs $O(m)$ or $O(n)$ flops to solve, respectively.

Figure 6.4 plots the relative residual errors for ADI solutions and solutions by Galerkin projection via ADI subspaces for $k = 34$ (*left*) and $k = 44$ (*right*) by applying Algorithms 1 and 2. For $k = 34$, in Algorithm 2 each Arnoldi run takes 44 steps and 22 best Ritz values are taken, and thus both $\mathbb{E}$ and $\mathbb{F}$ have 44 values among which 34 are selected in the end; and for $k = 44$, in Algorithm 2 each Arnoldi run takes 54 steps and 27 best Ritz values are taken, and thus both $\mathbb{E}$ and $\mathbb{F}$ have 54 values among which 44 are selected in the end. $\diamond$

7 Conclusions

We have presented a factored ADI for Sylvester equation $AX - XB = GF^*$, Lyapunov equation as a special case included. It is based on a set of formulas which generalize corresponding ones in the CF-ADI for Lyapunov equation. They enable one to compute the columns of the left factor and the rows of the right factor one block per step. We also demonstrate that often much more accurate solutions than the ADI solutions can be gotten by performing

[4] `http://math.nist.gov/MatrixMarket/data/Harwell-Boeing/lanpro/lanpro.html.`

Galerkin projection using the column spaces and row spaces of the computed approximate solutions.

References

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

[2] L. Bao, Y. Lin and Y. Wei, A new projection method for solving large Sylvester equations, Appl. Numer. Math. 57 (5–7) (2007) 521–532.

[3] R. H. Bartels and G. W. Stewart, Algorithm 432: The solution of the matrix equation $AX - BX = C$, Commun. ACM (8) (1972) 820–826.

[4] U. Baur and P. Benner, Cross-gramian based model reduction for data-sparse systems, Tech. rep., Fakultät für Mathematik, TU Chemnitz, 09107 Chemnitz, FRG, submitted for publication 2007.

[5] P. Benner, Solving large-scale control problems, IEEE Control Systems Magazine 14 (1)(2004) 44–59.

[6] P. Benner, The matrix factorization paradigm in solving matrix equations, Householder Symposium XVI, Seven Springs, PA, available electronically at `http://www.tu-chemnitz.de/~benner/talks/hh05.pdf` (May 2005).

[7] P. Benner, J.-R. Li and T. Penzl, Numerical solution of large Lyapunov equations, Riccati equations, and linear-quadratic control problems, Numer. Lin. Alg. Appl., to appear, 2008.

[8] P. Benner, H. Mena and J. Saak, On the parameter selection problem in the Newton-ADI iteration for large-scale Riccati equations, Electr. Trans. Num. Anal. 29 (2008) 136–149.

[9] P. Benner, E. Quintana-Ortí and G. Quintana-Ortí, Solving stable Sylvester equations via rational iterative schemes, J. Sci. Comput. 28 (2006) 51–83.

[10] R. Bhatia and P. Rosenthal, How and why to solve the operator equation $AX - XB = Y$, Bull. London Math. Soc. 29 (1997) 1–21.

[11] D. Calvetti and L. Reichel, Application of ADI iterative methods to the restoration of noisy images, SIAM J. Matrix Anal. Appl. 17 (1996) 165–186.

[12] Y. Chahlaoui and P. Van Dooren, A collection of benchmark examples for model reduction of linear time invariant dynamical systems, SLICOT Working Notes 2002-2, February 2002, available at `www.win.tue.nl/niconet/NIC2/benchmodred.html`.

[13] B. Datta, Numerical Methods for Linear Control Systems, Elsevier Academic Press, 2004.

[14] J. Demmel, Applied Numerical Linear Algebra, SIAM, Phildelphia, 1997.

[15] F. Ding and T. Chen, Hierarchical gradient-based identification of multivariable discrete-time systems. Automatica, 41 (2) (2005) 315–325.

[16] F. Ding and T. Chen, Hierarchical least squares identification methods for multivariable systems. IEEE Transactions on Automatic Control, 50 (3) (2005) 397–402.

[17] F. Ding and T. Chen, On Iterative Solutions of General Coupled Matrix Equations, SIAM J. Cont. Opt. 44 (6) (2005) 2269–2284.

[18] F. Ding and T. Chen, Gradient based iterative algorithms for solving a class

of matrix equations, IEEE Transactions on Automatic Control 50 (8) (2005) 1216–1221.

[19] F. Ding and T. Chen, Iterative least squares solutions of coupled Sylvester matrix equations, Systems and Control Letters 54 (2) (2005) 95–107.

[20] A. El Guennouni, K. Jbilou and J. Riquet, Block Krylov subspace methods for solving large Sylvester equations, Numer. Algorithms 29 (2002) 75–96.

[21] N. S. Ellner and E. L. Wachspress, Alternating direction implicit iteration for systems with complex spectra, SIAM J. Num. Anal. 3 (1991) 859–870.

[22] W. Enright, Improving the efficiency of matrix operations in the numerical solution of stiff ordinary differential equations, ACM Trans. Math. Softw. 4 (1978) 127–136.

[23] F. Ding, P. X. Liub and J. Ding, Iterative solutions of the generalized Sylvester matrix equations by using the hierarchical identification principle, Applied Mathematics and Computation 197 (2008) 41-50.

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

[25] G. H. Golub and C. F. Van Loan, Matrix Computations, Johns Hopkins University Press, Baltimore, Maryland, 3rd ed., 1996.

[26] S. Gugercin, D. Sorensen and A. Antoulas, A modified low-rank Smith method for large-scale, Numerical Algorithms 32 (2003) 27–55.

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

[28] K. Jbilou, Low rank approximate solutions to large Sylvester matrix equations, Appl. Math. Comput. 177 (2006) 365–376.

[29] P. Lancaster and M. Tismenetsky, The Theory of Matrices, 2nd ed., Academic Press, Orlando, 1985.

[30] J.-R. Li and J. White, Low-rank solution of Lyapunov equations, SIAM J. Matrix Anal. Appl. 24 (2002) 260–280.

[31] ——, Low-rank solution of Lyapunov equations, SIAM Rev. 46 (2004) 693–713.

[32] A. Lu and E. Wachspress, Solution of Lyapunov equations by ADI iteration, Comp. Math. Appl. 21 (1991) 43–58.

[33] T. Penzl, A cyclic low-rank smith method for large sparse Lyapunov equations, SIAM J. Sci. Comput. 21 (2000) 1401–1418.

[34] ——, Eigenvalue decay bounds for solutions of Lyapunov equations: the symmetric case, Systems Control Lett. 40 (2000) 139–144.

[35] ——, LYAPACK: A MATLAB toolbox for large Lyapunov and Riccati equations, model reduction problems, and linear-quadratic optimal control problems, users' guide (ver. 1.0), available at `www.tu-chemnitz.de/sfb393/lyapack/`, 2000.

[36] J. Sabino, Solution of Large-Scale Lyapunov Equations via the Block Modified Smith Method, PhD thesis, Rice University, Houston, Texas, 2006.

[37] V. Simoncini, On the numerical solution of $AX - XB = C$, BIT 36 (1996) 814–830.

[38] ——, A new iterative method for solving large-scale Lyapunov matrix equations, SIAM J. Sci. Comput. 29 (2007) 1268–1288.

[39] D. Sorensen and A. Antoulas, The Sylvester equation and approximate bal-

anced reduction, Linear Algebra Appl. 351/352 (2002) 671–700.

[40] N. Truhar and R.-C. Li, On ADI Method for Sylvester Equations, Technical Report 2008-02, Department of Mathematics, University of Texas at Arlington, 2008, available at `http://www.uta.edu/math/preprint/rep2008_02.pdf`.

[41] E. L. Wachspress, Iterative solution of the Lyapunov matrix equation, Appl. Math. Lett. 1 (1988) 87–90.

[42] E. L. Wachspress, The ADI Model Problem, Windsor, CA, 1995.

[43] E. Wachspress, Trail to a Lyapunov equation solver, Computers & Mathematics with Applications 55 (2008) 1653–1659.

[44] E. Wachspress, Adi Iteration Parameters For Solving Lyapunov And Sylvester Equations, Technical Report, March, 2009