A NEW ITERATIVE METHOD FOR SOLVING LARGE-SCALE LYAPUNOV MATRIX EQUATIONS*

V. SIMONCINI†

Abstract. In this paper we propose a new projection method to solve large-scale continuous-time Lyapunov matrix equations. The new method projects the problem onto a much smaller approximation space, generated as a combination of Krylov subspaces in A and A^{-1}. The reduced problem is then solved by means of a direct Lyapunov scheme based on matrix factorizations. The reported numerical results show the competitiveness of the new method, compared to a state-of-the-art approach based on the factorized Alternating Direction Implicit (ADI) iteration.

1. Introduction. Given the time-invariant linear system

$$\mathbf{x}'(t) = A\mathbf{x}(t) + B\mathbf{u}(t), \qquad \mathbf{x}(0) = x_0, \tag{1.1}$$

with $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times s}$ with $s \le n$, the solution of the associated continuous-time Lyapunov equation

$$AX + XA^T + BB^T = 0 \tag{1.2}$$

has a fundamental role in applied fields such as signal processing and system and control theory. We assume that $s \ll n$, and that A is dissipative, that is, $x^T(A + A^T)x < 0$ for all real vectors x. The symmetric solution X, called the reachability Gramian, carries important information on the stability and energy of the linear system (1.1) and on the feasibility of order reduction techniques [1], [4], [8].

The numerical solution of the Lyapunov matrix equation has been addressed in a large body of literature, and well established methods exist for small dimensional problems [2], [19]. Research has recently focused mostly on schemes for medium to large-scale Lyapunov equations, for which methods based on matrix factorizations become too demanding, both in terms of computational costs and memory requirements. For dense problems, efforts have been devoted to the development of matrix iterative methods based on the sign function, possibly exploiting the problem structure; see, e.g., [3], [5], and references therein.

In the case of sparse matrix A, projection-type methods based on Krylov subspaces, either using polynomials or rational functions of A, have been employed; see, e.g., [22], [23], [24], [25], [27], [31], and the general treatment in [1]. In particular, since the work by Ellner and Wachspress in [11], the ADI iteration has found great applicability in the solution of (1.2), and software implementing an efficient variant is now available [29]. A common feature of all these projection methods is that the approximate solution $\widehat{X}$ is given in factored form, $\widehat{X} = ZZ^T$ with Z of low column rank. The possibility of obtaining a good low rank approximate solution is supported by recent theoretical results showing the rapid decay of the eigenvalues of the symmetric and positive semidefinite exact solution; see [28], [15], [33]. This property can be fully exploited when dealing with large problems, since storing the whole, in general full matrix $\widehat{X}$ would require a significant amount of memory space. Other methods have been investigated, such as that in [21], which however do not build a low rank approximation.

*Version of May 29, 2006.

†Dipartimento di Matematica, Università di Bologna, Piazza di Porta S. Donato 5, I-40127 Bologna, Italy and CIRSA, Ravenna, Italy (valeria@dm.unibo.it).

In spite of these recent efforts, projection methods that are directly based on Krylov subspace approaches (cf. [31], [22]) may be slow, requiring the generation of a very large approximation space to obtain a sufficiently accurate solution; see the experiments reported in [27]. On the other hand, ADI-type methods need the a-priori evaluation of some parameters which may be hard to tune. In addition, memory and CPU time requirements remain an issue, for these methods perform various large matrix factorizations.

In this paper we propose a new projection-type method for large-scale problems, that is in general faster than approaches based on the ADI iteration, and does not require estimation of parameters. The new method projects the problem onto a small approximation space, generated as a combination of Krylov subspaces in A and A^{-1}. The reduced problem is then solved by means of a standard scheme for small Lyapunov equations. The method is equipped with a natural stopping criterion, which can be computed inexpensively. The performance of the new method is compared on benchmark problems to that of the cyclic implementation of the factorized ADI method available in [29].

The paper is organized as follows. Section 2 describes the general subspace projection framework, which characterizes the new approach and several other reduction methods. Section 3 and its subsections describe the new method, and its algorithmic aspects. Section 4 reviews the ADI iteration and its efficient variant, to prepare the notation for the numerical experiments reported in section 5. Finally, in section 6 we provide some conclusive remarks.

The following notation is used throughout. The square identity and zero matrices are denoted by I and O, respectively; rectangular portions will be identified by using subscripts. The only exceptions are the (block) columns of the identity, denoted by e_i, ith column (resp. E_i, $2i-1$st and $2i$th column of the identity matrix of even dimension). Matlab notation is used whenever possible [26]. $\|\cdot\|$ indicates the 2-norm for vectors and the induced norm for matrices, while $\|\cdot\|_F$ denotes the Frobenius norm, that is $\|A\|_F^2 = \sum_{i=1}^n \sum_{j=1}^m a_{i,j}^2$ for $A \in \mathbb{R}^{n\times m}$. Range$(V)$ indicates the space spanned by the columns of V. Finally, exact arithmetic is assumed throughout.

2. Projection methods for the Lyapunov equation. In this section we review a general framework associated with projection methods for the Lyapunov equation. This will allow us to simplify the presentation of the new approach in section 3.

Consider a subspace K of $\mathbb{R}^n$, and let V be a matrix whose orthonormal columns span K, with $B = VE$ for some matrix E, so that $B \in K$. We project the Lyapunov equation onto this subspace K: we set $H = V^TAV$, and seek an approximate solution in K in the form $\widehat{X} = VYV^T$. The projected problem may be obtained by imposing that the residual $R = A\widehat{X} + \widehat{X}A^T + BB^T$ be orthogonal to the approximation space K, the so-called Galerkin condition. In matrix terms, this can be written as[1] $V^TRV = O$. Since V has orthonormal columns, imposing $V^TRV = 0$ gives the following projected Lyapunov equation

$$V^TAVY + YV^TA^TV + EE^T = 0, \tag{2.1}$$

whose solution Y defines $\widehat{X}$. Since we assume that A is dissipative, V^TAV is stable and the equation (2.1) admits a unique solution. The projection is effective if a good approximation is attained for a small dimensional subspace, so that (2.1) can be

[1] This condition can be derived by first defining the matrix inner product $\langle X, Y\rangle = \mathrm{tr}(XY^T)$ and then imposing $\langle R, P\rangle = 0$ for any $P = VGV^T$ [31].

cheaply solved by means of procedures based on matrix factorizations; cf. [2], [19]. In turn, this effectiveness is more likely to take place whenever the exact solution can be accurately approximated by low rank matrices. An additional theoretical motivation for using such technique is given by the exponential form of the analytical solution,

$$X = \int_0^\infty e^{tA} B B^T e^{tA^T} dt. \tag{2.2}$$

Note that the use of this formulation has lead to different approximation procedures such as that in [16]. The quantity $W = e^{tA}B$ can be approximated in K as $\widehat{W} = Ve^{tH}E$. Therefore, an approximation to X can be obtained as

$$\widehat{X} = \int_0^\infty \widehat{W}\widehat{W}^T \, dt = V \left(\int_0^\infty e^{tH} E E^T e^{tH^T} \, dt \right) V^T =: VYV^T.$$

Using the fact that $H = V^T A V$ is stable, the following result can be readily established.

PROPOSITION 2.1. *The matrix* $Y = \int_0^\infty e^{tH} E E^T e^{tH^T} dt$ *is the unique solution to the Lyapunov equation (2.1).*

Although the results above hold independently of the specific approximation space K, the *effectiveness* of the approach crucially depends on K. The presentation above and also Proposition 2.1 were first introduced in [31] for $B \in \mathbb{R}^n$ and K corresponding to the Krylov subspace $\mathcal{K}_k(A, B) = \text{span}\{B, AB, \ldots, A^{k-1}B\}$. Further developments of Krylov subspace methods for $B \in \mathbb{R}^{n\times s}$ and using conditions on the residual different from the Galerkin one were discussed in [22]. Reasons for expecting good accuracy in the approximation for sufficiently large dimension of the Krylov subspace comes from the approximation properties of the exponential. Let V_k be such that $\text{Range}(V_k) = \mathcal{K}_k(A, B)$, and $H_k = V_k^T A V_k$. If $\widehat{W}_k = V_k e^{tH} E$ is a good approximation to $W = e^{tA}B$, we expect the corresponding integral to approximate X. In particular, assuming for the sake of the presentation $t = 1$, if the numerical range of A is contained in the complex disk $|z + \rho| \le \rho$ for some $\rho > 0$, then a bound of the form $\| \exp(A)v - V_k \exp(H_k)e_1\| = O((\frac{e\rho}{k})^k)$, holds, for $k \ge 2\rho$; see [20]. More accurate bounds can be obtained in the symmetric case [20], [36], [9].

Assume now that a sequence of approximation subspaces K_k of dimension k is given, with $K_k \subset K_{k+1}$, and satisfying $AK_k \subset K_{k+1}$. Note that these conditions on K_k are reasonably general.

Let the columns of V_k span K_k, $\underline{H}_k = V_{k+1}^T A V_k$, and let $X_k = V_k Y_k V_k^T$ be the approximate solution to (1.2) in K_k, obtained by imposing the Galerkin condition. The associated residual $R_k = AX_k + X_k A^T + BB^T$ satisfies

$$R_k = V_{k+1}\underline{H}_k V_k Y_k V_k^T + V_k Y_k \underline{H}_k^T V_{k+1}^T + V_k E E^T V_k^T =: V_{k+1}\widehat{R}V_{k+1}^T. \tag{2.3}$$

The selection $K_k = \mathcal{K}_k(A, B)$ fulfils all conditions above, and in addition it can be shown that $\|R_k\| = \|E_{k+1}^T \underline{H}_k Y_k\|$, so that the quality of the approximation X_k can be measured by computing the residual norm, without explicitly evaluating the residual matrix. This setting has been heavily exploited to devise competitive approaches, and to derive theoretical properties; see, e.g., [30], [23], [22].

3. The new iterative procedure. The general description of the previous section suggests that the approximation space K could be enriched to contain more information than that of the regular Krylov subspace $\mathcal{K}_k(A, B)$. To this end, starting

with the pair $\{B, A^{-1}B\}$, we propose to generate a sequence of approximation spaces that contain information on both A and A^{-1}, by adding two vectors at the time, one multiplied by A, and one by A^{-1}, that is

$$\text{span}\{B, A^{-1}B, AB, A^{-2}B, A^2B, A^{-3}B, \ldots\}.$$

As discussed in section 2, once the approximation space is generated, the original problem is projected onto this space, and the resulting small dimensional problem is solved. To simplify the presentation, we derive the new method for $\text{rank}(B) = 1$, however the generalization to larger rank is immediate. Implementation details of the approach are given in section 3.1.

Given B, A, set $V_1 = \text{gram\_sh}([B, A^{-1}B])$, $\mathcal{V}_0 = \emptyset$.
For $m = 1, 2, \ldots,$
1. $\mathcal{V}_m = [\mathcal{V}_{m-1}, V_m]$
2. Set $\mathcal{T}_m = \mathcal{V}_m^T A \mathcal{V}_m$ and $E = \mathcal{V}_m^T B$
3. Solve $\quad \mathcal{T}_m Y + Y \mathcal{T}_m^T + EE^T = 0$ and set $Y_m = Y$
4. If converged then $X_m = \mathcal{V}_m Y_m \mathcal{V}_m^T$ and stop
5. $V'_{m+1} = [AV_m^{(1)}, A^{-1}V_m^{(2)}]$
6. $\widehat{V}_{m+1} \leftarrow$ orthogonalize V'_{m+1} w.r.to $\mathcal{V}_m$
7. $V_{m+1} = \text{gram\_sh}(\widehat{V}_{m+1})$

At each iteration of this process, two new vectors are added to the space, so that the space dimension increases by two. Unless breakdown occurs, at the mth iteration the method has constructed an orthonormal basis of dimension $2m$, given by the columns of the matrix $\mathcal{V}_m = [V_1, V_2, \ldots, V_m]$, $V_i \in \mathbb{R}^{n\times 2}$, and $\mathbf{K}_m := \text{Range}(\mathcal{V}_m)$. The orthogonalization is performed first with respect to the previous basis vectors, and then within the new block of two vectors (function gram_sh, see section 3.1). At each iteration (cf. step 3), a small dimensional Lyapunov equation with matrices of size $2m \times 2m$ is solved. Since B is a vector, $E = E_1 V_1^T B = [e_1\beta, 0] \in \mathbb{R}^{2m\times 2}$, with $\beta = \|B\|$. Only at convergence (step 4), the approximate solution to X is constructed as

$$X_m = \mathcal{V}_m Y_m \mathcal{V}_m^T, \tag{3.1}$$

with Y_m symmetric. The following important properties are a consequence of the subspace definition.

PROPOSITION 3.1. *For any $m \geq 1$, the space $\mathbf{K}_m$ satisfies $\mathbf{K}_m = \mathcal{K}_{2m}(A, A^{-m}B)$. In particular, it follows that*

$$A\mathbf{K}_m \subseteq \mathbf{K}_{m+1}. \tag{3.2}$$

We observe that the results of Proposition 3.1 ensure that $\mathcal{T}_m$ is block upper Hessenberg, since $V_k^T A V_j = O$ for $k > j + 1$, $j = 1, 2, \ldots$. In particular, property (3.2) allows us to employ the framework of section 2. Moreover, since A is dissipative, $\mathcal{T}_m$ is stable, so that Y_m is positive (semi-)definite.

We next derive some recursive relations that can be used to significantly reduce the computational costs of the basic algorithm. Let $\underline{\mathcal{H}}_m = [\mathcal{H}_m; \chi_m E_m^T] \in \mathbb{R}^{2(m+1)\times 2m}$ collect all orthonormalization coefficients, that is

$$\widehat{V}_{m+1} = [AV_m^{(1)}, A^{-1}V_m^{(2)}] - \mathcal{V}_m \mathcal{H}_m [e_{2m-1}, e_{2m}] \tag{3.3}$$

$$V_{m+1}\chi_m = \widehat{V}_{m+1}; \tag{3.4}$$

note that $\mathcal{H}_m$ is block upper Hessenberg with 2×2 blocks. Let also $\underline{\mathcal{T}}_m := \mathcal{V}_{m+1}^T A\mathcal{V}_m$. The computation of $\mathcal{T}_m$ seems to require additional matrix-vector products with A and extra inner products of long vectors. We next derive a recursion for $\underline{\mathcal{T}}_m$ that completely avoids this expensive step.

PROPOSITION 3.2. *Let $\ell^{(k)} = (\ell_{ij})$ be the 2×2 matrix such that $V_k = \hat{V}_k\ell^{(k)}$ in (3.3), $k = 1,\dots,m$. Let*

$$\underline{\mathcal{T}}_m = (t_{i,j})_{i=1,\dots,2m+2,j=1,\dots,2m} \quad \textit{and} \quad \mathcal{H}_m = (h_{i,j})_{i=1,\dots,2m,j=1,\dots,2m}.$$

Then (odd columns)

$$t_{:,2k-1} = h_{:,2k-1}, \quad k = 1,\dots,m,$$

while (even columns)

$$\begin{aligned}
(k=1)\quad & t_{:,2} = \frac{1}{\ell_{11}^{(1)}}(h_{:,1}\ell_{12}^{(1)} + e_1\ell_{22}^{(1)}) \qquad t_{:,4} = (e_2 - \underline{\mathcal{T}}_1 h_{1:2,2})\ell_{22}^{(2)}, \quad \rho^{(2)} = \frac{\ell_{12}^{(2)}}{\ell_{22}^{(2)}} \\
(1<k\le m)\quad & t_{:,2k} = t_{:,2k} + t_{:,2k-1}\rho^{(k)} \\
& t_{:,2k+2} = (e_{2k} - \underline{\mathcal{T}}_k h_{1:2k,2k})\ell_{22}^{(k+1)}, \qquad \rho^{(k+1)} = \frac{\ell_{12}^{(k+1)}}{\ell_{22}^{(k+1)}}.
\end{aligned}$$

Proof. For $k \ge 1$, we write $V_k = [V_k^{(1)}, V_k^{(2)}] \in \mathbb{R}^{n\times 2}$. Using (3.3), we immediately have $AV_k^{(1)} = V_{k+1}\chi_k e_1 + \mathcal{V}_k\mathcal{H}_k e_{2k-1} = \mathcal{V}_{k+1}\underline{\mathcal{H}}_m e_{2k-1}$, so that the odd columns of $\mathcal{T}_m$ satisfy

$$\mathcal{V}_m^T A\mathcal{V}_m e_{2k-1} = \mathcal{V}_m^T AV_k^{(1)} = \begin{bmatrix} I_{2k+1} \\ O \end{bmatrix} \underline{\mathcal{H}}_k e_{2k-1} = \mathcal{H}_m e_{2k-1}.$$

The even columns need to take into account that the recurrence comes from multiplications by A^{-1}, and thus the term $AV_k^{(2)}$ is not explicitly available. However, from (3.3) we obtain

$$V_k^{(2)} = A\widehat{V}_{k+1}^{(2)} + A\mathcal{V}_k\mathcal{H}_k e_{2k},$$

which gives a recurrence for the $(2k+2)$th column $A\widehat{V}_{k+1}^{(2)}$. Hence,

$$\mathcal{V}_{m+1}^T A\widehat{V}_{k+1}^{(2)} = \mathcal{V}_{m+1}^T V_k^{(2)} - \mathcal{V}_{m+1}^T A\mathcal{V}_k\mathcal{H}_k e_{2k} = e_{2k} - \begin{pmatrix} \underline{\mathcal{T}}_k \\ O_{m-k,2k} \end{pmatrix} \mathcal{H}_k e_{2k}.$$

To complete the derivation, we recall that $V_{k+1} = \widehat{V}_{k+1}\ell^{(k+1)}$, with $\ell^{(k+1)} = (\ell_{ij})$ upper triangular, so that $V_{k+1}^{(2)} = \widehat{V}_{k+1}^{(1)}\ell_{12} + \widehat{V}_{k+1}^{(2)}\ell_{22}$, from which $AV_{k+1}^{(2)} = A\widehat{V}_{k+1}^{(1)}\ell_{12} + A\widehat{V}_{k+1}^{(2)}\ell_{22} = AV_{k+1}^{(1)}\ell_{11}^{-1}\ell_{12} + A\widehat{V}_{k+1}^{(2)}\ell_{22}$. Therefore,

$$\mathcal{V}_{m+1}^T AV_{k+1}^{(2)} = \mathcal{V}_{m+1}^T\mathcal{V}_{k+2}\underline{\mathcal{H}}_{k+1}e_{2k+1}\ell_{11}^{-1}\ell_{12} + \left(e_{2k} - \begin{pmatrix} \underline{\mathcal{T}}_k \\ O_{m-k,2k} \end{pmatrix}\mathcal{H}_k e_{2k}\right)\ell_{22}.$$

The recursion thus follows. □

The recurrence for $\underline{\mathcal{T}}_m$ shows that the lower diagonal 2×2 block has zero second row. This fact can be used to simplify the computation of the residual norm.

PROPOSITION 3.3. *For $m \geq 1$, let $\mathbf{K}_m = \mathrm{Range}(\mathcal{V}_m)$ and let X_m be the approximate solution obtained as in (3.1) with associated residual $R_m = AX_m + X_m A^T + BB^T$. Then Range($R_m$) = Range($\mathcal{V}_{m+1}$) and $\|R_m\| = \|e_{2m+1}^T \underline{\mathcal{T}}_m Y_m\|$.*

Proof. Since $A\mathcal{V}_m = \mathcal{V}_{m+1}L$, for some matrix L, the first result follows from (2.3). Therefore, $R_m = \mathcal{V}_{m+1}\widehat{R}_m\mathcal{V}_{m+1}^T$, with

$$\widehat{R}_m = \mathcal{V}_{m+1}^T R_m \mathcal{V}_{m+1} = \begin{pmatrix} 0 & \mathcal{V}_m^T R_m V_{m+1} \\ V_{m+1}^T R_m \mathcal{V}_m & 0 \end{pmatrix}.$$

Using $V_{m+1}^T R_m \mathcal{V}_m = E_{m+1}^T \underline{\mathcal{T}}_m Y_m$, it follows $\|R_m\| = \|\widehat{R}_m\| = \|V_{m+1}^T R_m \mathcal{V}_m\| = \|E_{m+1}^T \underline{\mathcal{T}}_m Y_m\| = \|e_{2m+1}^T \underline{\mathcal{T}}_m Y_m\|$. □

The matrix Y_m in the solution $X_m = \mathcal{V}_m Y_m \mathcal{V}_m^T$ is often numerically positive *semi*-definite [28], [15], [33]. If this is so, it is possible to generate a lower rank X_m as $X_m = Z_m Z_m^T$ with Z_m of rank lower than $2m$, the dimension of Y_m. More precisely, let $Y_m = WDW^T$ be the eigendecomposition of the $2m \times 2m$ matrix Y_m with D diagonal having diagonal entries sorted in decreasing order, and let $W_k D_k W_k^T$ be the approximate decomposition obtained by truncating to the first k columns of W, with $k \ll 2m$. Then $Y_m \approx W_k D_k W_k^T$ and the approximation is determined by using $Z_m = V_m W_k D_k^{\frac{1}{2}}$. The number k of columns to retain may be determined by inspecting the diagonal elements of D. In our implementation, we discarded the columns of W corresponding to diagonal elements of D less than 10^{-12}; a more conservative tolerance could also be employed.

Due to the form of the approximate solution, it is also possible to monitor at low cost the amount by which the approximate solution varies as the iterations proceed. Indeed, letting $Y_m = [\widehat{Y}_m, y_m; y_m^T, \eta_m]$ with $\widehat{Y}_m$ of size $(2m-2) \times (2m-2)$ we have

$$X_m - X_{m-1} = [\mathcal{V}_{m-1}, V_m] \begin{bmatrix} \widehat{Y}_m - Y_{m-1} & y_m \\ y_m^T & \eta_m \end{bmatrix} [\mathcal{V}_{m-1}, V_m]^T =: \mathcal{V}_m G_m \mathcal{V}_m^T,$$

so that

$$\frac{\|X_m - X_{m-1}\|_F}{\|X_{m-1}\|_F} = \frac{\|G_m\|_F}{\|Y_{m-1}\|_F}, \tag{3.5}$$

where both G_m and Y_{m-1} have small dimension. We do not advocate using this quantity as stopping criterion, however we shall sometime refer to it in our experiments for comparison purposes.

3.1. Implementation details. An implementation of the algorithm associated with the procedure outlined above is given by the function Krylov-Plus-Inverted-Krylov (hereafter K-PIK).

```
function Z=KPIK(A,v,m_max,k_max,tol)

nrmb=norm(rhs,'fro')^2;
nrma=norm(A,'fro');


[LA,UA]= lu(A);              % For sym matrices use UA=chol(-A); LA=-UA';
[ibeta,U(:,1:2)] = gram_sh([rhs, UA\ (LA\ rhs)]);

for j=1:m_max,

  % Expand the approximation space
```

```
  jms=(j-1)*2+1;j1s=(j+1)*2;js=j*2;js1=js+1;
  Up(:,1) = A*U(:,jms);     Up(:,2) = UA\ (LA\ U(:,js));

  %New Basis Block
  for il=1:2           % Reorthogonalization
      k_min=max(1,j-k_max);
      for kk=k_min:j
         k1=(kk-1)*2+1; k2=kk*2;
         coef= U(1:n,k1:k2)'*Up;
         H(k1:k2,jms:js) = H(k1:k2,jms:js)+ coef;
         Up = Up - U(:,k1:k2)*coef;
      end
  end
  if (j<=m)
      [hinv,U(1:n,js1:j1s)]=gram_sh(Up);    H(js1:j1s,jms:js)=inv(hinv);
  end
  I = speye(js+2);

  % Update the T recurrence
  if (j==1),
    l(1:js+1,j)=[ H(1:3,1), speye(3,1)]*ibeta(1:2,2)/ibeta(1,1);
  else
    l(1:js+2,j)=l(1:js+2,j) + H(1:js+2,js-1)*rho;
  end
  T(1:js+2,1:2:js)=H(1:js+2,1:2:js); T(1:js+2,2:2:js)=l(1:js+2,1:j);
  l(1:js+2,j+1)=( I(1:js+2,j*2)-T(1:js+2,1:js)*H(1:js,js))*hinv(2,2);
  rho = hinv(1,2)/hinv(1,1);

  % Solve small Lyapunov equation
  Y = lyap(T(1:js,1:js),beta^2*speye(js,1)*speye(js,1)');

  % Compute residual and test for convergence
  cc(1:2,1) = H(js1:j1s,js-1); cc(1:2,2) = l(js1:j1s,j);
  nrmx = norm(Y,'fro');
  er2(j) = norm(cc*Y(js-1:js,:))/(nrmb+nrma*nrmx);
  if (er2(j)<tol), break, end

end

% Reduced rank solution   tol = 1e-12
[uY,sY]=eig(Y); [sY,id]=sort(diag(sY));
sY=flipud(sY); uY=uY(:,id(end:-1:1)); is=sum(abs(sY)>1e-12);
Y0 = uY(:,1:is)*diag(sqrt(sY(1:is)));
Z = U(:,1:js)*Y0;
```

A few comments are in order. The LU factorization of A is employed for medium size matrices, and the Cholesky factorization of $-A$ should be used for A symmetric. For large matrices, a preconditioned iterative method could be employed to solve systems with A at each iteration, where the preconditioner could be generated once for all.

Function `gram_sh` performs the modified Gram-Schmidt orthogonalization of the new block [13]. The same process is used for the orthogonalization with respect to older vectors in the basis. Note that this reduces to a procedure similar to the block Lanczos algorithm (see [14]) when the problem is symmetric, by setting `k_max = 2`, so that only orthogonalization with respect to the previous two blocks is enforced. In particular, for A symmetric, the whole matrix $\mathcal{V}_m$ is only needed at convergence to recover the solution factor. Therefore, one can either store the columns that are not needed in the orthogonalization process, or use a "two-pass" procedure, in which the

basis vectors are first discarded. At convergence, the short-term iteration is performed once again to recover the basis vectors and update the final solution. Although this procedure is expensive, essentially doubling the computational cost of the generation of the approximation space, the memory requirements become extremely limited, allowing one to tackle very large problems, in the symmetric case.

The function `lyap` computes the numerical solution of the small dimensional Lyapunov equation by means of a Schur decomposition type method; in our experiments, we used the algorithm proposed by Sorensen and Zhou [34], which implements the Bartels-Stewart method with only real arithmetic, leading to significant computational savings compared to the original method [2]; see also [32] for a recent evaluation of solvers from the SLICOT package ([37]) for Lyapunov and other matrix equations stemming from control system design. Since the matrix $\mathcal{T}_m$ is stable, other procedures such as the Hammarling method could also be employed [19]. Note also that the fact that $\mathcal{T}_m$ is block upper Hessenberg significantly reduces the computational work of the method. We should remark, however, that as the approximation subspace increases, this step becomes the most computational demanding task of the algorithm.

The stopping criterion is given by

$$\frac{\|AX_m + X_m A^T + BB^T\|}{2\|A\|_F\|Y_m\|_F + \|B\|_F^2} \leq \texttt{tol} \tag{3.6}$$

where `tol` is a chosen fixed threshold. The numerator is obtained by using the computationally inexpensive relation of Proposition 3.3. We observe that the output matrix is the memory-conserving factor Z_m of $X_m = Z_m Z_m^T$.

3.2. Finite Termination and convergence. In exact arithmetic, the procedure just described has finite termination, since for m such that $2m \geq n$ the generated vectors span the whole space. However, since two vectors at the time are added to the current basis, loss of rank may occur during the orthogonalization with respect to the older basis vectors, so that the next basis pair using V_{m+1} cannot be built. We next show that this implies convergence, as expected by the result of Proposition 3.1.

PROPOSITION 3.4. *Assume that $m-1$ iterations of the K-PIK method have been taken, with $B \in \mathbb{R}^n$. At the mth iteration, assume that $\widehat{V}_{m+1}$ in (3.3) has rank less than two. Then $\mathcal{V}_m \cup \{\widehat{V}_{m+1}\}$ is an invariant subspace of A with respect to B. Therefore, Range($\mathcal{V}_m \cup \{\widehat{V}_{m+1}\}$) contains the exact solution.*

Proof. We recall that $\text{Range}(\mathcal{V}_m) = \mathcal{K}_{2m}(A, A^{-m}B)$. Therefore, the given assumption requires that there exist $\alpha_i \in \mathbb{R}$, $i = -m-1, \ldots, m$ not all zero such that

$$\begin{aligned}\alpha_{-m}A^{-m}B + \ldots + \alpha_{-1}A^{-1}B + \alpha_0 B + \alpha_1 AB + \\ \ldots + \alpha_{m-1}A^{m-1}B + \alpha_m A^m B + \alpha_{-m-1}A^{-m-1}B = 0.\end{aligned}$$

Assume first $\alpha_{-m-1} = 0$ and $\alpha_m \neq 0$. Then, $A(A^{m-1}B) = \sum_{j=0}^{2m-1} \beta_j A^j A^{-m} B$, which shows that $A\mathcal{V}_m = \mathcal{V}_m \mathcal{T}_m$, and thus $\mathcal{V}_m$ is invariant for A. Using $X_m = \mathcal{V}_m Y_m \mathcal{V}_m^T$, we obtain

$$AX_m + X_m A^T + BB^T = \mathcal{V}_m(\mathcal{T}_m Y_m + Y_m \mathcal{T}_m + EE^T)\mathcal{V}_m^T = O.$$

Next, assume that $\alpha_{-m-1} \neq 0$ and $\alpha_m \neq 0$. Then,

$$A^{-1}A^{-m}B = \sum_{j=0}^{2m-1} \beta_j A^j A^{-m} B + \beta_{2m} A^m B,$$

or, equivalently, $A^{-m}B - \sum_{j=1}^{2m} \beta_j A^j A^{-m} B = \beta_{2m} A(A^m B)$, so that $A[\mathcal{V}_m, V_m^{(1)}] = [\mathcal{V}_m, V_m^{(1)}]\widehat{\mathcal{T}}_m$ where $\widehat{\mathcal{T}}_m$ is the restriction of $\mathcal{T}_{m+1}$ to the first $(2m+1)$ columns and rows. Therefore, $[\mathcal{V}_m, V_m^{(1)}]$ is invariant for A. As in the previous case, writing $X'_m = [\mathcal{V}_m, V_m^{(1)}] Y'_m [\mathcal{V}_m, V_m^{(1)}]^T$, where Y'_m is the solution to the Lyapunov equation with coefficient matrix $\widehat{\mathcal{T}}_m$, we obtain $AX'_m + X'_m A^T + BB^T = [\mathcal{V}_m, V_m^{(1)}](\widehat{\mathcal{T}}_m Y'_m + Y'_m \widehat{\mathcal{T}}_m + EE^T)[\mathcal{V}_m, V_m^{(1)}]^T = O$, and the proof is completed. □

If B is not a vector, then loss of rank while generating the basis may occur without convergence. In this case, deflation strategies should be employed, as in stardard block Krylov subspace methods [18].

In section 2 we discussed the close relation between the approximation to the exponential and the projected Lyapunov equation. The new approach implicitly approximates $W = \exp(A)B$ with $\widehat{W} = \mathcal{V}_m \exp(\mathcal{T}_m)E$ in $\mathbf{K}_m = \mathcal{K}_{2m}(A, A^{-m}B)$, and results in this context can be used to theoretically motivate our approach. For A symmetric and B a vector, it is shown by Druskin and Knizhnerman in [10] that the choice $\mathcal{K}_{2m}(A, A^{-m}B)$ yields an improved approximation to the exponential, over the standard Krylov subspace $\mathcal{K}_{2m}(A, B)$. More precisely, they show that the quality of the approximation to the exponential (and in fact to a whole class of functions) in $\mathcal{K}_{2m}(A, B)$ corresponds to that in $\mathcal{K}_{\sqrt{2m}}(A, A^{-m}B)$. Therefore, using $\mathcal{K}_{\sqrt{2m}}(A, A^{-m}B)$ allows one to achieve the same approximation quality as a regular Krylov space with a much lower subspace dimension.

We explicitly observe that the method proposed in [10] aims at approximating functions of matrices (in the symmetric case). In addition, the authors suggest to fix a-priori the number k of matrix-vector products with A^{-1}, so as to implicitly build the approximation space $\text{span}\{A^{-k}v, A^{-k+1}v, \ldots, Av, \ldots, A^{m-1}v, \ldots\}$. Here, we continue increasing the subspace in "both directions" until convergence. This can be easily done thanks to the recurrence in Proposition 3.2 for the matrix $\mathcal{T}_m$.

4. The Cholesky-Factorized ADI iteration. The Lyapunov equation was shown to be a "model problem" for the classical ADI iteration in [11] and an implementation for sparse symmetric positive definite A was proposed in [38]. Since then, several papers have discussed the practical implementation of the method, including the case of complex spectrum [12], [25]. Much more recently, a Cholesky-factorized version has made the method more appealing for large-scale computation, for the approximate solution can be written as the product of two low-rank matrices [24]. We next review the method, which will be used for numerical comparisons in section 5. Finally, we should mention that for the CF-ADI method to be derived, A is only required to be stable, that is its eigenvalues should lie in the left-half plane, whereas projection-type methods need dissipativity, to ensure that the projected and restricted coefficient matrix in (2.1) is stable.

Given a set of parameters $p_1, p_2, \ldots, p_\ell$, the basic ADI iteration determines an approximate solution as (cf., e.g., [24])

$$X_0 = 0,\ X_j = -2p_j(A + p_j I)^{-1} BB^T (A + p_j I)^{-T} + (A + p_j I)^{-1}(A - p_j I) X_{j-1} (A - p_j I)^T (A + p_j I)^{-T}, \quad j = 1, \ldots, \ell.$$

It can be shown that X_j is symmetric and positive (semi-)definite so that it can be represented as $X_j = Z_j Z_j^T$. This Cholesky-type factorization with the possibly low-rank matrix Z_j allows one to iteratively form the columns of Z_j one at the time.

Setting $Z_1 = \sqrt{-2p_1}(A + p_1 I)^{-1} B$, the iteration is given by

$$Z_j = [\sqrt{-2p_j}(A + p_j I)^{-1} B, (A + p_j I)^{-1}(A - p_j I) Z_{j-1}] \in \mathbb{R}^{n \times j},\ j = 1, \ldots, \ell.$$

The actual implementation, hereafter CF-ADI, requires only one solve with $A + p_j I$ per iteration; see, e.g., [24, Algorithm 2]. A cyclic procedure is obtained by continuing the iteration above and cyclically using the parameters, and this is also referred to as the cyclic low-rank Smith(ℓ) method [27]. It is interesting to notice that the columns of the factor Z_j span a special rational Krylov subspace generated by alternating the parameters as poles in the construction of the space [24].

For small size matrices, the recurrence can be stopped as soon as the residual matrix $AZ_jZ_j^T + Z_jZ_j^TA^T + BB^T$ is sufficiently small in some norm. In fact, the Frobenius norm of the residual matrix can be more cheaply computed using (see [27])

$$\begin{aligned} \|AZ_jZ_j^T + Z_jZ_j^TA^T + BB^T\|_F &= \|[AZ_j, Z_j, B][Z, AZ_j, B]^T\|_F \\ &= \|r\mathcal{I}r^T\|_F, \end{aligned} \tag{4.1}$$

where r is the triangular square matrix of the "economy-size" QR factorization of the matrix $[AZ_j, Z_j, B]$, and $\mathcal{I}$ is a permutation matrix. Due to memory limitations, the previous quantities cannot be computed explicitly for large problems, therefore, a criterion based on the current iterate is commonly employed. More precisely, if $Z_j = [Z_{j-1}, z_j]$, then using $\|Z_jZ_j^T - Z_{j-1}Z_{j-1}^T\|_F = \|z_j\|^2$, the iteration is stopped if $\|z_j\|$ (absolute) or $\|z_j\|/\|Z_{j-1}\|_F$ (relative) is sufficiently small, which implies that the recurrence stagnates. Note that neither of these last two stopping criteria ensures that convergence has taken place.

The ADI parameters represent the key ingredient for the success of the method, and the performance can wildly vary if the parameters are not chosen sufficiently well. Optimal ADI parameters are a function of ℓ and solve the following min-max problem ([38])

$$\min_{p_1,\ldots,p_\ell} \max_{x \in \Omega} \left| \prod_{j=1}^{\ell} \frac{(p_j - x)}{(p_j + x)} \right|,$$

where $\Omega \subseteq \mathbb{C}^-$ contains the spectrum of A. Optimal parameters are known analytically when A has real spectrum; see, e.g., the discussion in [12]. The situation is significantly more complicated for complex spectrum. In both cases, however, quite accurate information on the region Ω is required. The computation of quasi-optimal parameters has raised considerable attention, both as a general approximation theory problem, as well as in more application oriented contexts; see, e.g., [35], [12] and references therein. More recently an economical procedure was proposed by Penzl in [27], which is based on the classical and shift-and-invert Arnoldi procedures and using the associated Ritz values to compute suboptimal ADI parameters. In our experiments, we used the function[2] `lp_para` in the LYAPACK package ([29]) to compute these parameters.

Another important aspect associated with the implementation of the CF-ADI iteration is related to the solves with $A + p_j I$, $j = 1, \ldots, \ell$. For very sparse and structured matrices, e.g., banded with small bandwidth, each matrix $A + p_j I$ can be

[2]The Arnoldi procedures in `lp_para` were modified to include reorthogonalization, so as to limit the occurrence of unstable shifts.

factorized once for all and the Cholesky factor stored and used cyclically when p_j is employed; this approach is suggested for instance in [29]. For the sake of comparison, we report the performance of such implementation in Example 5.1 where factorizing the matrix requires limited memory resources. Since this procedure is clearly too memory consuming in large-scale applications, we opted for solving systems with $A + p_j I$ repeatedly at occurrence.

An additional difficulty with CF-ADI is given by the use of complex arithmetic when A is nonsymmetric and the ADI parameters are complex conjugate. In the original CF-ADI iteration, conjugate parameters can be coupled [24]. In the cyclic version, coupling does not seem to be doable, and the whole computation is carried out in complex arithmetic to get a complex Z_j. Note that it is possible to recover a real factor Z_j from the complex counterpart at the cost of additional computation [29]. Due to the added computational difficulties caused by complex ADI parameters, Gugercin et al. suggest only dealing with real parameters, even when the matrix has complex spectrum [17]. These considerations show that the performance of the method is crucially affected by the choices taken in the computation involving the ADI parameters. To fully explore the properties of the method, in our experiments with the cyclic CF-ADI method we decided to use complex parameters and thus allowed complex arithmetic throughout the iteration. Further considerations regarding the typical behavior of this method are reported in the next section.

5. Numerical experiments. In this section we report on our numerical experience with the new method described in section 3, compared with the cyclic CF-ADI method. All reported experiments were obtained using Matlab 7.1 (R3) ([26]), on a PC with a 2Ghz processor, and 2GBytes of RAM.

Both CPU time (in seconds) and number of iterations (as defined next) are used to measure the cost of the approaches. For the CF-ADI method, the number of iterations (counted as the number of cycles times the number of parameters) coincides with the number of shifted systems to be solved, and also with the number of long vectors that need to be stored. We recall here that we are assuming that B is a vector. When B is a matrix with $s > 1$ columns, then s new columns need to be stored at each iteration. For the K-PIK method, each (block) iteration involves one system solution with A, and two new vectors entering the basis. Therefore, for the Krylov approach the final memory requirements for the approximation space are given by twice the number of iterations. In summary, the number of iterations indicates the amount of system solves required by each method.

In all tests with the new method, we also report the rank of the final approximate solution as discussed in section 3. The cost of this computation is included in the overall cost of the method. For the CF-ADI iteration, we did not attempt to compute the factor rank to limit the computational costs. Nonetheless, we remark that a procedure has been proposed in [17] to more cheaply determine a lower-rank factor.

Unless explicitly stated, the CF-ADI complex parameters were determined with search parameters $(\ell, k, m) = (10, 40, 20)$, where ℓ is the number of ADI parameters[3], k is the number of Arnoldi iterations, and m is the number of inverted-Arnoldi iterations in the procedure `lp_para`; these are the parameters suggested in [27] for similar problems. We stress that the cost of generating the ADI parameters was *not* taken into account in the performance evaluation. Indeed, unless some tuning on (ℓ, k, m) is needed (cf. for instance Example 5.2), this pre-processing only employs a low per-

[3]This number may need to be adjusted to accommodate complex conjugates.

centage of the whole cost of the algorithm. Our numerical experience showed that the computation of the parameters was also sensitive to the choice of the starting vector that was used for generating spectral information. ADI parameters determined with the same values (ℓ, k, m) but with a different starting vector for the two Krylov subspaces were significantly different, affecting the performance of the ADI iteration.

Stopping criterion. As already mentioned, a comparison between the new method and the cyclic CF-ADI method faces the problem of the different stopping criteria used in the two cases, namely (3.6) and the one discussed in section 4. Indeed, the fact that the two stopping criteria are satisfied in the two cases, does not ensure that the quality of the approximate solution is comparable. For this reason we decided to take the "user point of view", that is, we measure what the user can really handle during the run of the algorithm. To this end, we fix a certain tolerance, and evaluate the performance of the two methods by monitoring the associated "standard" stopping criterion. More precisely,

$$\text{K-PIK:} \qquad \frac{\|AX_m + X_m A^T + BB^T\|}{2\|A\|_F\|Y_m\|_F + \|B\|_F^2} \le \texttt{tol} \tag{5.1}$$

$$\text{Cyclic CF-ADI:} \qquad \frac{\|z_j\|}{\|Z_{j-1}\|_F} \le \sqrt{\texttt{tol}}. \tag{5.2}$$

We recall that (5.2) corresponds to $\|X_j - X_{j-1}\|_F / \|X_{j-1}\|_F \le \texttt{tol}$. In all examples considered, we used $\texttt{tol} = 10^{-10}$. In some cases, the final residual norm (4.1) of the CF-ADI iteration was also computed, but this cost was not included in the total computational requirements of the method, because in most cases overwhelming. We found the CF-ADI residual norm often to be above the requested residual tolerance (cf. Example 5.4).

Example 5.1. This example is taken from [27, Example 6.3] and describes a model of heat flow with convection in the given domain. The associated parabolic equation is given by

$$\mathbf{x}' = \mathbf{x}_{xx} + \mathbf{x}_{yy} - 10x\mathbf{x}_x - 1000y\mathbf{x}_y + \mathbf{b}(x,y)\mathbf{u}(t)$$

on the unit square, with Dirichlet boundary conditions. The 4900×4900 matrix A resulting from the centered finite difference discretization of the differential terms is nonsymmetric with complex eigenvalues, and has 24220 nonzero entries. Vector b was taken to be the vector of all ones. This choice allows one to completely replicate the reported experiments. The results in the following table show the performance of the new method, compared to the cyclic CF-ADI iteration. For the sake of completeness, in this example we also report the performance of the CF-ADI method when all matrix factorizations are carried out at the beginning of the computation and thus stored, so that at each ADI iteration only triangular solves are performed (CF-ADI_m column).

	K-PIK	CF-ADI	CF-ADI_m
CPU time (s)	1.1	14	9
# iterations	19	80	80
dim. Approx. Space	38	80	80
Solution rank	35	80	80

The reported values show the competitiveness of the new method on this medium size problem, with low CPU time and memory requirements. A posteriori, we com-

puted the residual norm of the CF-ADI solution and found that it was above 10^{-5}. Following (3.5), we also computed the relative norm of the variation in the solution of the new method, which for this example was $5.7 \cdot 10^{-8}$. All these numbers show that the quality of the approximate solutions significantly differs, depending on the employed measure. In this particular case, we observed in a separate experiment that letting the K-PIK solution difference (3.5) go below the threshold 10^{-10} only took less than one additional second of CPU time.

EXAMPLE 5.2. This example is a three-dimensional variant of the previous problem, and the differential equation is given by

$$\mathbf{x}' = \mathbf{x}_{xx} + \mathbf{x}_{yy} + \mathbf{x}_{zz} - 10x\mathbf{x}_x - 1000y\mathbf{x}_y - 10\mathbf{x}_z + \mathbf{b}(x,y)\mathbf{u}(t)$$

on the unit cube, with Dirichlet boundary conditions. The nonsymmetric matrix resulting from centered finite difference discretization with 18 nodes in each direction is of size 5832×5832 and has 38880 nonzero entries. Once again, b was taken to be the vector of all ones. The results are shown in the first part of the following table.

n		K-PIK	CF-ADI (10,40,20)	CF-ADI (16,60,40)
5832	CPU time (s)	15	149	112
	# iterations	56	110	85
	dim. Approx. Space	112	110	85
	Solution rank	47	110	85
		KPIC	CF-ADI (16,60,40)	CF-ADI (30,60,60)
10648	CPU time (s)	43	336	236
	# iterations	45	85	65
	dim. Approx. Space	90	85	65
	Solution rank	45	85	65

The reported numbers are consistent with those of the previous experiments. It is worth noticing that at completion, for the new method the difference $\|X_m - X_{m-1}\|/\|X_{m-1}\|$ was equal to about $2 \cdot 10^{-8}$ in both problems (cf. (3.5)).

The behavior for different CF-ADI parameters is also reported. The computed parameters for the two choices are also depicted in Figure 5.1, where the symbol '$\times$' denotes the eigenvalues of A, 'o' the ADI parameters for $(\ell, k, m) = (10, 40, 20)$ and '*' the case $(\ell, k, m) = (16, 60, 40)$. The last choice is clearly more successful in capturing the relevant spectral boundary, suggesting better performance.

In the table we also report the results for a run of the same problem with a slightly finer discretization, leading to a 10648×10648 nonsymmetric matrix A with 71632 nonzero entries. The results are consistent with those for the coarser mesh, confirming the difficulty of the CF-ADI pre-processing in finding effective ADI parameters even for different discretizations of the same problem.

EXAMPLE 5.3. With this example we start our tests on symmetric problems. Here we consider the parabolic equation

$$\mathbf{x}' = \mathbf{x}_{xx} + \mathbf{x}_{yy} + \mathbf{x}_{zz} + \mathbf{b}(x,y)\mathbf{u}(t)$$

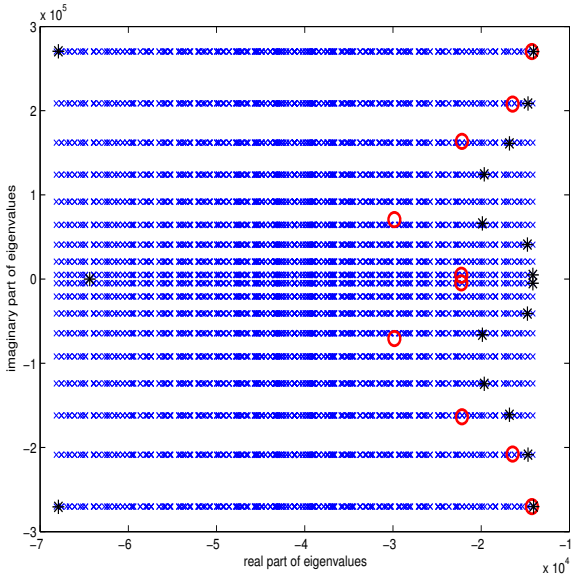

FIG. 5.1. *Example 5.2. Spectrum of matrix A ('$\times$'); ADI parameters for $(\ell, k, m) = (10, 40, 20)$ (symbol 'o'); and for $(\ell, k, m) = (16, 60, 40)$ (symbol '*').*

with homogeneous boundary conditions, and discretize the 3D Laplace operator with centered finite differences in the unit cube. The resulting symmetric matrix for the given discretization mesh has size $n = 27000$. The vector b is the vector of all ones. Note that both methods can take great advantage of the symmetry of A. In particular, the CF-ADI is all performed in real arithmetic, with $\ell = 10$ real parameters. The results are reported below.

	K-PIK	CF-ADI
CPU time (s)	132	202
# iterations	8	20
dim. Approx. Space	16	20
Solution rank	14	20

The numbers confirm the competitiveness of the new method, which does not require any pre-processing to gain spectral information.

EXAMPLE 5.4. We next consider a benchmark problem stemming from a semi-discretized Heat Transfer Problem for Optimal Cooling of Steel Profiles [6]. The associated linear time-invariant system is given by

$$\begin{aligned} M\mathbf{x}'(t) &= N\mathbf{x}(t) + B_0\mathbf{u}(t), \quad \text{with} \quad t > 0, \\ \mathbf{x}(0) &= \mathbf{x}_0, \\ \mathbf{y}(t) &= C_0\mathbf{x}(t). \end{aligned}$$

with M symmetric positive definite, N symmetric negative definite and B_0 of small column rank. We considered two tests available in the data set in [7], corresponding to different mesh resolutions, with dimensions $n = 1357$ (file `rail_1357_c60`) and

$n = 5177$ (`rail_5177_c60`). In our experiments, we used the first column of B_0 for each data set.

The numerical treatment of this problem requires the discussion of some additional details. For M nonsingular, the problem can be recast as in (1.1). Using $M = LL^T$, we have $A = L^{-1}NL^{-T}$, $B = L^{-1}B_0$ and $C = C_0L^{-1}$. Clearly, none of these matrices is computed explicitly, rather, the following operations are performed:

$$Av = L^{-1}(N(L^{-T}v)), \qquad A^{-1}v = L^T(N^{-1}(Lv)).$$

Thus, matrix-vector products with A require the sequential solution of two triangular systems. In the case of the ADI iteration, the system solution is given as $(A+pI)^{-1}v = L^T(N+pM)^{-1}Lv$, therefore requiring the solution of $\ell = 10$ systems with coefficient matrix $N + pM$ at each ADI cycle.

The results for both problems are summarized in the following table.

n		K-PIK	CF-ADI (10,40,20)	CF-ADI (20,40,40)
1357	CPU time (s)	5.67	4.36	6.96
	# iterations	59	90	140
	dim. Approx. Space	118	90	140
	Solution rank	12	90	140
	Final Residual F-norm	$< 10^{-10}$(†)	$2.8 \cdot 10^{-5}$	$2.8 \cdot 10^{-5}$
5177	CPU time (s)	50	62	32
	# iterations	86	220	120
	dim. Approx. Space	172	220	120
	Solution rank	13	220	120
	Final Residual F-norm	$< 10^{-10}$(†)	$9.9 \cdot 10^{-5}$	$9.9 \cdot 10^{-5}$

†: Before rank truncation

We observe that the first problem is quite small, and the K-PIK method is penalized, since its auxiliary costs (reorthogonalizations and the solution of a small Lyapunov equation) may be significant.

In this set of experiments, $\|B\| \approx 10^{-7}$ and $\|A\|_F \approx 10^{-3}$, so that $\|X\| \approx 10^{-3}$. These figures make the two stopping criteria behave very differently and seem to show that the CF-ADI method performs better than Krylov, when the parameters are properly chosen (Note, however, that the performance of the parameters is dimension dependent). We investigate further this example by reporting the convergence curves in Figure 5.2. In there, we show the behavior of the relative difference $\|X_m - X_{m-1}\|/\|X_{m-1}\|$ for both methods, and also the residual norm for the Krylov approach. Clearly, had the K-PIK convergence been measured in terms of the solution difference, the iteration would have been stopped much earlier, with significant savings both in memory and CPU time requirements. In the table we also report the final relative residual norm (cf. (5.1)) of the CF-ADI iteration. According to this measure, the accuracy of the CF-ADI solution is not satisfactory, for a tolerance `tol`$=10^{-10}$.

6. Conclusions. In this paper we have presented a new projection method for determining low-rank approximate solutions to large-scale Lyapunov algebraic equations with dissipative coefficient matrix. Our experiments show that the new algorithm is extremely competitive with the state-of-the-art solver cyclic CF-ADI, and

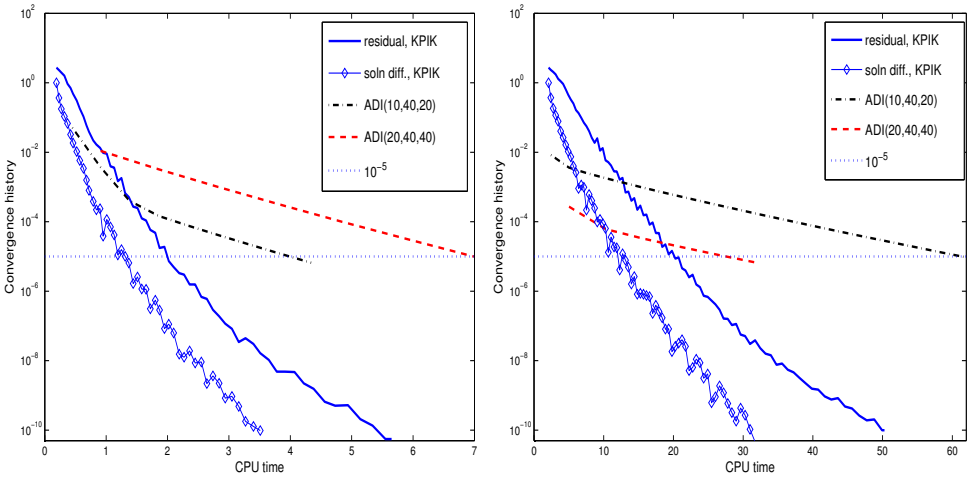

FIG. 5.2. *Example 5.4. Convergence history of the two methods. Left:* $n = 1357$. *Right:* $n = 5177$.

it does not require estimation of parameters. The new approach provides a natural, cheaply computable a-posteriori stopping criterion. Note that such a measure is missing in the cyclic CF-ADI iteration. Moreover, the new method provides an economical procedure to further decrease the rank of the final approximate factor. The approach effectively reduces the original problem to one of much smaller dimension, in a way that information on the problem augments as the dimension of the approximation space increases. This view point is completely different from CF-ADI type of iteration, where spectral information on the problem is gathered before-hand, and then used repeatedly. This difference is reminiscent of the differences between semi-iterative and fully non-stationary methods in the solution of systems of linear equations [13].

We have tested the new approach on medium size problems, while even greater benefits can be obtained when solving larger problems. In such setting, the inner system solves could be effectively carried out by a preconditioned iterative method. Note that this procedure is far less straightforward in the case of CF-ADI, for the presence of several complex shifts. We also mention that if the dimension of the approximation space becomes large, then the costly step of solving the projected Lyapunov equation can be done periodically, and not at every iteration. This small variant may significantly reduce the overall cost of the method.

The general framework described in section 2 suggests that further acceleration techniques could be derived by constructing new approximation spaces that satisfy the given constraints. Moreover, several computational issues deserve deeper investigation, such as the possibility of implementing truncation, restarting or some other memory-conserving strategy.

Acknowledgements. We thank Peter Benner and Jens Saak for insightful conversations and for pointing to [7]. All reported results were obtained using the computer facilities of the SINCEM Laboratory at CIRSA, Ravenna.

REFERENCES

[1] A. C. Antoulas. *Approximation of large-scale Dynamical Systems*. Advances in Design and Control. SIAM, Philadelphia, 2005.

[2] R. H. Bartels and G. W. Stewart. Algorithm 432: Solution of the Matrix Equation AX+XB=C. *Comm. of the ACM*, 15(9):820–826, 1972.

[3] Ulrike Baur and Peter Benner. Factorized solution of Lyapunov equations based on hierarchical matrix arithmetic. Technical Report 05-xx, Technische Universität, Chemnitz, D, October 2005.

[4] P. Benner. Control Theory. Technical report, http://www.tu-chemnitz.de/~benner, 2006. to appear in 'Handbook of Linear Algebra'.

[5] P. Benner, E. S. Quintana-Ortí, and G. Quintana-Ortí. Solving stable Sylvester equations via rational iterative schemes. Technical Report 04-08, Technische Universität, Chemnitz, D, 2004. To appear in J. Scientific Computing.

[6] P. Benner and J. Saak. Efficient numerical solution of the LQR-problem for the heat equation. *PAMM, Proc. Appl. Math. Mech.*, 4(1):648–649, 2004.

[7] Benchmark Collection. Oberwolfach model reduction benchmark collection, 2003. `http://www.imtek.de/simulation/benchmark`.

[8] M. J. Corless and A. E. Frazho. *Linear systems and control - An operator perspective*. Pure and Applied Mathematics. Marcel Dekker, New York - Basel, 2003.

[9] V. Druskin and L. Knizhnerman. Two polynomial methods of calculating functions of symmetric matrices. *U.S.S.R. Comput. Math. Math. Phys.*, 29:112–121, 1989.

[10] V. Druskin and L. Knizhnerman. Extended Krylov subspaces: approximation of the matrix square root and related functions. *SIAM J. Matrix Anal. Appl.*, 19(3):755–771, 1998.

[11] Nancy S. Ellner and Eugene L. Wachspress. New ADI model problem applications. In *Proceedings of 1986 ACM Fall joint computer conference, Dallas, Texas, United States*, pages 528–534. IEEE Computer Society Press, Los Alamitos, CA, 1986.

[12] Nancy S. Ellner and Eugene L. Wachspress. Alternating Direction Implicit iteration for systems with complex spectra. *SIAM J. Numer. Anal.*, 23(3):859–870, 1991.

[13] G. Golub and C. F. Van Loan. *Matrix Computations*. The Johns Hopkins University Press, Baltimore, 3rd edition, 1996.

[14] G.H. Golub and R. Underwood. The block Lanczos method for computing eigenvalues. In J. R. Rice, editor, *Mathematical Software III*, pages 364–377. Academic Press, New York, 1977.

[15] L. Grasedyck. Existence of a low rank of h-matrix approximant to the solution of a Sylvester equation. *Numerical Linear Algebra with Appl.*, 11(4):371–389, 2004.

[16] T. Gudmundsson and A. J. Laub. Approximate solution of large sparse Lyapunov equations. *IEEE Trans. Automatic Control*, 39(5):1110–1114, 1994.

[17] S. Gugercin, D. C. Sorensen, and A. C. Antoulas. A modified low-rank Smith method for large-scale Lyapunov equations. *Numerical Algorithms*, 32:27–55, 2003.

[18] Martin H. Gutknecht. Block Krylov space methods for linear systems with multiple right-hand sides: an introduction. In Abul Hasan Siddiqi, Iain S. Duff, and Ole Christensen, editors, *Modern Mathematical Models, Methods and Algorithms for Real World Systems*, New Delhi, 2006. Anamaya Publishers. to appear.

[19] S. J. Hammarling. Numerical solution of the stable, non-negative definite Lyapunov equation. *IMA J. Numer. Anal.*, 2:303–323, 1982.

[20] M. Hochbruck and C. Lubich. On Krylov subspace approximations to the matrix exponential operator. *SIAM J. Numer. Anal.*, 34(5):1911–1925, 1997.

[21] M. Hochbruck and G. Starke. Preconditioned Krylov subspace methods for Lyapunov matrix equations. *SIAM Matrix Anal. and Appl.*, 16(1):156–171, 1995.

[22] I. M. Jaimoukha and E. M. Kasenally. Krylov subspace methods for solving large Lyapunov equations. *SIAM J. Numer. Anal.*, 31(1):227–251, Feb. 1994.

[23] K. Jbilou and A.J. Riquet. Projection methods for large Lyapunov matrix equations. *Linear Algebra and its Applications*, 415(2-3):344–358, 2006.

[24] J.-R. Li and J. White. Low-Rank solutions of Lyapunov equations. *SIAM J. Matrix Anal. Appl.*, 24(1):260–280, 2002.

[25] An Lu and Eugene L. Wachspress. Solution of Lyapunov equations by Alternating Direction Implicit iteration. *Computers Math. Applic.*, 21(9):43–58, 1991.

[26] The MathWorks, Inc. *MATLAB 7*, September 2004.

[27] T. Penzl. A cyclic low-rank Smith method for large sparse Lyapunov equations. *SIAM J. Sci. Comput.*, 21(4):1401–1418, 2000.

[28] T. Penzl. Eigenvalue decay bounds for solutions of Lyapunov equations: the symmetric case. *Systems and Control Letters*, 40(2):139–144, 2000.

[29] T. Penzl. Lyapack users guide. Technical Report SFB393/00-33, TU Chemnitz, 09107 Chemnitz, D, 2000. Available from `http://www.tu-chemnitz.de/sfb393/sfb00pr.html`.

[30] M. Robbé and M. Sadkane. A convergence analysis of GMRES and FOM for Sylvester equations. *Numerical Algorithms*, 30:71–89, 2002.

[31] Y. Saad. Numerical solution of large Lyapunov equations. In M. A. Kaashoek, J. H. van Schuppen, and A. C. Ran, editors, *Signal Processing, Scattering, Operator Theory, and Numerical Methods. Proceedings of the international symposium MTNS-89, vol III*, pages 503–511, Boston, 1990. Birkhauser.

[32] Martin Slowik, Peter Benner, and Vasile Sima. Evaluation of the linear matrix equation solvers in SLICOT. Technical report, 2004. To appear in Journal of Numerical Analysis, Inustrial and Applied Mathematics, 2006.

[33] D. Sorensen and Y. Zhou. Bounds on eigenvalue decay rates and sensitivity of solutions to Lyapunov equations. Technical Report 02-07, Rice University, Houston, Texas, 2002.

[34] D. C. Sorensen and Y. Zhou. Direct methods for matrix Sylvester and Lyapunov equations. *J. Appl. Math.*, 2003:277–303, 2003.

[35] G. Starke. Fields of values and the ADI method for non-normal matrices. *Lin. Alg. Appl.*, 180:199–218, 1993.

[36] H. Tal-Ezer. Spectral methods in time for parabolic problems. *SIAM J. Numer. Anal.*, 26:1–11, 1989.

[37] A. van den Boom, A. Brown, F. Dumortier, A. Geurts, S. Hammarling, R. Kool, M. Vanbegin, P. Van Dooren, and S. Van Huffel. SLICOT, a subroutine library for control and system theory. In *Preprints IFAC Symp. on CADCS*, pages 89–94, Swansea, UK, July 1991.

[38] E. L. Wachspress. Iterative Solution of the Lyapunov Matrix Equations. *Appl. Math. Lett.*, 1(1):87–90, 1988.