COMPRESS-AND-RESTART BLOCK KRYLOV SUBSPACE METHODS FOR SYLVESTER MATRIX EQUATIONS

DANIEL KRESSNER[*], KATHRYN LUND[†], STEFANO MASSEI[‡], AND DAVIDE PALITTA[§]

Abstract. Block Krylov subspace methods (KSMs) comprise building blocks in many state-of-the-art solvers for large-scale matrix equations as they arise, e.g., from the discretization of partial differential equations. While extended and rational block Krylov subspace methods provide a major reduction in iteration counts over polynomial block KSMs, they also require reliable solvers for the coefficient matrices, and these solvers are often iterative methods themselves. It is not hard to devise scenarios in which the available memory, and consequently the dimension of the Krylov subspace, is limited. In such scenarios for linear systems and eigenvalue problems, restarting is a well explored technique for mitigating memory constraints. In this work, such restarting techniques are applied to polynomial KSMs for matrix equations with a compression step to control the growing rank of the residual. An error analysis is also performed, leading to heuristics for dynamically adjusting the basis size in each restart cycle. A panel of numerical experiments demonstrates the effectiveness of the new method with respect to extended block KSMs.

Key words. linear matrix equations, block Krylov subspace methods, low-rank compression, restarts

AMS subject classifications. 65F10, 65N22, 65J10, 65F30, 65F50

1. Introduction.

This work is concerned with numerical methods for solving large-scale Sylvester matrix equations of the form

$$AX + XB + \boldsymbol{C}\boldsymbol{D}^* = 0, \tag{1.1}$$

with coefficient matrices $A, B \in \mathbb{C}^{n\times n}$ and $\boldsymbol{C}, \boldsymbol{D} \in \mathbb{C}^{n\times s}$. Although it is essential that both A, B are square, it is not essential that they are of the same size. The latter assumption has been made to simplify the exposition; our developments easily extend to square coefficients A, B of different sizes.

Throughout this paper, we will assume $s \ll n$ and view $\boldsymbol{C}, \boldsymbol{D}$ as block vectors. This not only implies that the right-hand side $\boldsymbol{C}\boldsymbol{D}^*$ has low rank, but it also implies that, under suitable additional conditions on the coefficients, such as the separability of the spectra of A and $-B$, the desired solution X admits accurate low-rank approximations; see, e.g., [3, 23, 43].

The Sylvester equation (1.1) arises in a variety of applications. In particular, model reduction [2] and robust/optimal control [53] for control problems governed by discretized partial differential equations (PDEs) give rise to Sylvester equations that feature very large and sparse coefficients A, B. Also, discretized PDEs themselves can sometimes be cast in the form (1.1) under suitable separability and smoothness assumptions [40]. Another source of applications are linearizations of nonlinear problems, such as the algebraic Riccati equation [11], and dynamic stochastic general

[*]EPF Lausanne, Switzerland, daniel.kressner@epfl.ch

[†]Charles University, Prague, Czech Republic, kathryn.lund@karlin.mff.cuni.cz. Supported in part by the Charles University PRIMUS grant, project no. PRIMUS/19/SCI/11, and in part by the SNSF research project *Fast algorithms from low-rank updates*, grant number: 200020_178806.

[‡]EPF Lausanne, Switzerland, stefano.massei@epfl.ch. Supported by the SNSF research project *Fast algorithms from low-rank updates*, grant number: 200020_178806.

[§]Research Group Computational Methods in Systems and Control Theory (CSC), Max Planck Institute for Dynamics of Complex Technical Systems, Sandtorstraße 1, 39106 Magdeburg, Germany, palitta@mpi-magdeburg.mpg.de.

equilibrium models [12], where the solution of (1.1) is needed in iterative solvers. We refer to the surveys [10, 48] for further applications and details.

Several of the applications mentioned above potentially lead to coefficients A, B that are too large to admit any sophisticated operation besides products with (block) vectors. In particular, the sparse factorization of A, B and, consequently, the direct solution of linear systems involving these matrices may be too expensive. This is the case, for example, when sparse factorizations lead to significant fill-in, as for discretized three-dimensional PDEs [17], or when A, B are actually dense, as for discretized boundary integral equations [46]. In these situations, the methods available for solving (1.1) essentially boil down to Krylov subspace methods. One of the most common variants, due to Saad [44] and Jaimoukha and Kasenally [32], first constructs orthonormal bases for the two block Krylov subspaces associated with the pairs $A, \boldsymbol{C}$ and $B^*, \boldsymbol{D}$, respectively, and then obtains a low-rank approximation to X via a Galerkin condition. In the following, we will refer to this method as the standard Krylov subspace method (SKSM).

The convergence of SKSM can be expected to be slow when the separation between the spectra of A and $-B$ is small. This happens, for example, when A and B are symmetric positive definite and ill-conditioned, which is usually the case for discretized elliptic PDEs. Slow convergence makes SKSM computationally expensive. As the number of iterations increases, the memory requirements as well as the cost for orthogonalization and extracting the approximate solution increase. Restarting is a common way to mitigate these effects when solving linear systems with, e.g., GMRES [45]. In principle, the idea of restarting can be directly applied to Sylvester equations because (1.1) can be recast as a linear system of size $n^2 \times n^2$. However, as we will discuss in more detail in Section 2, such an approach increases the ranks of the right-hand side and is, in turn, by itself not well suited for SKSM. In this work, we propose and partly analyze an algorithm that avoids this problem by combining restarting with compression.

The *ADI method* [21] and *rational Krylov subspace methods (RKSM)* [9] often converge faster in situations where SKSM struggles. However, this improvement comes at the expense of having to solve (shifted) linear systems with A, B in each iteration. As we are bound to the use of matrix-vector products with A, B, iterative methods, such as CG and GMRES, need to be used for solving these linear systems. These inner solvers introduce another error, and the interplay between this inexactness and the convergence of the outer method has been recently analyzed in [36]. Two major advantages of the inner-outer paradigm are that it allows for the use of well established linear solvers and the straightforward incorporation of preconditioners. On the other hand, it also has the disadvantage that it is difficult to reuse information from one outer iteration to the next, often leading to a large number of matrix-vector products overall. Moreover, the convergence of the outer scheme strongly depends on the selection of the shift parameters that define the rational functions used within ADI and RKSM. In the case that the chosen shifts do not result in fast convergence, then the disadvantages of SKSM, such as high memory requirements, still persist. Several strategies for choosing the shifts adaptively have been proposed; see [8, 18] and the references therein. An a priori strategy for selecting shift parameters, which does not take spectral information of A, B into account, is adopted in the so-called *extended Krylov subspace method (EKSM)* [14, 47]. EKSM is a particular case of RKSM,[1] where half of the shifts are set to 0 and the remaining are set to ∞. Other

[1]EKSM is, however, often implemented using block Gram-Schmidt, unlike RKSM, which builds

large-scale methods for Sylvester equations that require the solution of shifted linear systems and could be combined with iterative solvers in an inner-outer fashion include the approximate power iteration from [29] and the restarted parameterized Arnoldi method in [1].

Besides the inner-outer solvers discussed above, only a few attempts have been made to design mechanisms to limit the memory requirements of Krylov subspace methods for Sylvester equations. In [52], a connection to implicitly restarted Arnoldi method for eigenvalue problems is made. For symmetric (positive definite) A, B, a two-pass SKSM, such as the one discussed in [34], could be used. During the first pass, only the projected equation is constructed and solved; in the second pass, the method computes the product of the Krylov subspace bases with the low-rank factors of the projected solution.

The rest of this paper is structured as follows. In Section 2, we describe the compress-and-restart method proposed in this work for Sylvester equations as well as its adaptation to the special case of Lyapunov matrix equations. Section 3 provides an error analysis for the approximate solution obtained with our method. Finally, in Section 4, the performance of our method is demonstrated and compared to combinations of EKSM with iterative linear solvers.

2. Restarted SKSM with compression for linear matrix equations.

We first recall the general framework of projection methods for Sylvester equations and then explain our new restarted variant.

2.1. Projection methods and SKSM.

Projection methods for solving the Sylvester equation (1.1) seek an approximate solution of the form

$$\widetilde{X} = \boldsymbol{\mathcal{U}}_m \widetilde{Y} \boldsymbol{\mathcal{V}}_m^*, \tag{2.1}$$

where the columns of $\boldsymbol{\mathcal{U}}_m$ and $\boldsymbol{\mathcal{V}}_m$ form orthonormal bases of suitably chosen (low-dimensional) subspaces. The small core factor $\widetilde{Y}$ is determined by, e.g., imposing a Galerkin condition on the residual matrix $R = A\widetilde{X} + \widetilde{X}B + \boldsymbol{C}\boldsymbol{D}^*$; see, e.g., [48].

In SKSM, the subspaces determining (2.1) are block Krylov subspaces. More specifically, $\boldsymbol{\mathcal{U}}_m = [\boldsymbol{U}_1|\cdots|\boldsymbol{U}_m]$, $\boldsymbol{\mathcal{V}}_m = [\boldsymbol{V}_1|\cdots|\boldsymbol{V}_m] \in \mathbb{R}^{n\times ms}$; $\boldsymbol{U}_i, \boldsymbol{V}_i \in \mathbb{R}^{n\times s}$; and $i = 1, \dots, m$, are such that

$$\operatorname{range}(\boldsymbol{\mathcal{U}}_m) = \mathcal{K}_m(A, \boldsymbol{C}) = \operatorname{colspan}\{\boldsymbol{C}, A\boldsymbol{C}, A^2\boldsymbol{C}, \dots, A^{m-1}\boldsymbol{C}\} \subset \mathbb{C}^n$$

and, analogously, $\operatorname{range}(\boldsymbol{\mathcal{V}}_m) = \mathcal{K}_m(B^*, \boldsymbol{D})$.[2] If the block Arnoldi process is used for obtaining these bases, then the following block Arnoldi relations hold:

$$A\boldsymbol{\mathcal{U}}_m = \boldsymbol{\mathcal{U}}_m \mathcal{H}_m + \boldsymbol{U}_{m+1} H_{m+1,m} \boldsymbol{E}_m^*, \qquad B^*\boldsymbol{\mathcal{V}}_m = \boldsymbol{\mathcal{V}}_m \mathcal{G}_m + \boldsymbol{V}_{m+1} G_{m+1,m} \boldsymbol{E}_m^*, \tag{2.2}$$

with $ms \times ms$ block Hessenberg matrices $\mathcal{H}_m$ and $\mathcal{G}_m$; $s \times s$ matrices $H_{m+1,m}$ and $G_{m+1,m}$; and $\boldsymbol{E}_m^* = [0_s|\cdots|0_s|I_s] = \boldsymbol{e}_m^* \otimes I_s$, where 0_s and I_s denote the $s \times s$ zero and identity matrices, respectively. We refer to, e.g., [22, 24] for more details on block Krylov subspaces. The matrix $\widetilde{Y}$ is obtained by solving the projected equation

$$\mathcal{H}_m Y + Y\mathcal{G}_m^* + \left(\boldsymbol{\mathcal{U}}_m^* \boldsymbol{C}\right)\left(\boldsymbol{\mathcal{V}}_m^* \boldsymbol{D}\right)^* = 0. \tag{2.3}$$

one column at a time.

[2] Note that in the block Krylov subspace literature, often a distinction is made between the "true" block Krylov subspace with elements in $\mathbb{C}^{n\times s}$ and the span of all the columns of the basis vectors [22, 24]. For the sake of simplifying notation, $\mathcal{K}_m(A, \boldsymbol{C})$ and $\mathcal{K}_m(B^*, \boldsymbol{D})$ will always denote the latter here, meaning they are subspaces of $\mathbb{C}^n$.

This is again a Sylvester equation of the form (1.1) and, since m should be much smaller than n, the equation is of moderate size. Therefore a direct solver, such as the Bartels-Stewart method [5], can be employed for its solution.

Throughout the above discussion, we assumed that none of the block basis vectors $\boldsymbol{U}_i$ or $\boldsymbol{V}_i$, $i = 1, \ldots, m$, is rank-deficient. Numerical rank-deficiency is rare in practice, but so-called "inexact" rank-deficiency may benefit from a deflation or column-replacement procedure; see, e.g., [13, 51].

Having the solution $\widetilde{Y}$ of (2.3) at hand, we can compute the Frobenius norm of the residual matrix at a low cost as explained in [33, 48]:

$$\|R\|_F^2 = \left\|H_{m+1,m}\boldsymbol{E}_m^*\widetilde{Y}\right\|_F^2 + \left\|\widetilde{Y}\boldsymbol{E}_m G_{m+1,m}^*\right\|_F^2. \tag{2.4}$$

Similarly, for the spectral norm we have

$$\|R\|_2 = \max\left\{\left\|H_{m+1,m}\boldsymbol{E}_m^*\widetilde{Y}\right\|_2, \left\|\widetilde{Y}\boldsymbol{E}_m G_{m+1,m}^*\right\|_2\right\}. \tag{2.5}$$

Once the residual norm is sufficiently small, the approximation $\widetilde{X}$ from (2.1) is returned. Note that $\widetilde{X}$ is a large, dense matrix, which is always kept in factored form. Specifically, given a factorization $\widetilde{Y} = \boldsymbol{Y}_L\boldsymbol{Y}_R^*$, which can be computed, e.g., via a truncated singular value decomposition (SVD), we store the low-rank factors $\boldsymbol{X}_L = \boldsymbol{\mathcal{U}}_m\boldsymbol{Y}_L$ and $\boldsymbol{X}_R = \boldsymbol{\mathcal{V}}_m\boldsymbol{Y}_R$ of $\widetilde{X} = \boldsymbol{X}_L\boldsymbol{X}_R^*$, where L and R here simply denote "left" and "right," respectively.

2.2. Restarts and compression. We now present the new restarted procedure. Suppose that m_0 iterations of SKSM have been performed, leading to an approximate, possibly quite inaccurate solution $X^{(0)} = \boldsymbol{X}_L^{(0)}\left(\boldsymbol{X}_R^{(0)}\right)^*$ with

$$Y^{(0)} = \boldsymbol{Y}_L^{(0)}\left(\boldsymbol{Y}_R^{(0)}\right)^*, \quad \boldsymbol{X}_L^{(0)} = \boldsymbol{\mathcal{U}}_{m_0}\boldsymbol{Y}_L^{(0)}, \text{ and } \boldsymbol{X}_R^{(0)} = \boldsymbol{\mathcal{V}}_{m_0}\boldsymbol{Y}_R^{(0)}.$$

For a linear system, a correction to a given approximate solution is obtained by solving the linear system (approximately) with the right-hand side replaced by the residual; see, e.g., [28]. Applying this principle to (1.1), one first solves

$$AZ^{(1)} + Z^{(1)}B + R^{(0)} = 0, \tag{2.6}$$

with $R^{(0)} := A\boldsymbol{X}_L^{(0)}\left(\boldsymbol{X}_R^{(0)}\right)^* + \boldsymbol{X}_L^{(0)}\left(\boldsymbol{X}_R^{(0)}\right)^*B + \boldsymbol{C}\boldsymbol{D}^*$, and then adds the correction $Z^{(1)}$ to $X^{(0)}$ in order to obtain $X^{(1)}$.

The Arnoldi relations (2.2) imply that the residual matrix $R^{(0)}$ admits the following low-rank representation:

$$\begin{aligned}
R^{(0)} &= A\,\boldsymbol{\mathcal{U}}_{m_0}Y^{(0)}\boldsymbol{\mathcal{V}}_{m_0}^* + \boldsymbol{\mathcal{U}}_{m_0}Y^{(0)}\boldsymbol{\mathcal{V}}_{m_0}^*B + \boldsymbol{C}\boldsymbol{D}^* \\
&= \boldsymbol{\mathcal{U}}_{m_0}(\mathcal{H}_{m_0}Y^{(0)} + Y^{(0)}\mathcal{G}_{m_0}^* + (\boldsymbol{\mathcal{U}}_{m_0}^*\boldsymbol{C})(\boldsymbol{\mathcal{V}}_{m_0}^*\boldsymbol{D})^*)\boldsymbol{\mathcal{V}}_{m_0}^* \\
&\quad + \boldsymbol{U}_{m_0+1}H_{m_0+1,m_0}\boldsymbol{E}_{m_0}^*Y^{(0)}\boldsymbol{\mathcal{V}}_{m_0}^* + \boldsymbol{\mathcal{U}}_{m_0}Y^{(0)}\boldsymbol{E}_{m_0}G_{m_0+1,m_0}^*\boldsymbol{V}_{m_0+1}^* \\
&= \left[\boldsymbol{U}_{m_0+1}H_{m_0+1,m_0} \,|\, \boldsymbol{\mathcal{U}}_{m_0}Y^{(0)}\boldsymbol{E}_{m_0}G_{m_0+1,m_0}^*\right]\left[\boldsymbol{\mathcal{V}}_{m_0}Y^{(0)}\boldsymbol{E}_{m_0} \,|\, \boldsymbol{V}_{m_0+1}\right]^* \\
&:= \boldsymbol{C}^{(1)}\left(\boldsymbol{D}^{(1)}\right)^*.
\end{aligned}$$

Because $\boldsymbol{C}^{(1)}$, $\boldsymbol{D}^{(1)}$ have $2s$ columns, this shows that $R^{(0)}$ has rank at most $2s$ and, in turn, we can again employ a block Krylov subspace method for solving (2.6). Note, however, that the Krylov subspaces are different from the ones used for $X^{(0)}$. In order to distinguish them more clearly, we will from now on add the superscript $^{(0)}$ to quantities generated by SKSM for constructing $X^{(0)}$; that is, we will write $\boldsymbol{\mathcal{U}}_{m_0}^{(0)}, \boldsymbol{\mathcal{V}}_{m_0}^{(0)}$ instead of $\boldsymbol{\mathcal{U}}_{m_0}, \boldsymbol{\mathcal{V}}_{m_0}$. SKSM applied to the correction equation (2.6) constructs orthonormal bases $\boldsymbol{\mathcal{U}}_{m_1}^{(1)}$ and $\boldsymbol{\mathcal{V}}_{m_1}^{(1)}$ for $\mathcal{K}_{m_1}(A, \boldsymbol{C}^{(1)})$ and $\mathcal{K}_{m_1}(B^*, \boldsymbol{D}^{(1)})$, respectively. The correction is again returned in factorized form as $Z^{(1)} = \boldsymbol{Z}_L^{(1)}\big(\boldsymbol{Z}_R^{(1)}\big)^*$, with $Y^{(1)} = \boldsymbol{Y}_L^{(1)}\big(\boldsymbol{Y}_R^{(1)}\big)^*$, $\boldsymbol{Z}_L^{(1)} = \boldsymbol{\mathcal{U}}_{m_1}^{(1)}\boldsymbol{Y}_L^{(1)}$, $\boldsymbol{Z}_R^{(1)} = \boldsymbol{\mathcal{V}}_{m_1}^{(1)}\boldsymbol{Y}_R^{(1)}$.

The procedure can be iterated; i.e., we can perform multiple restarts. After k restarts, the approximate solution is given by

$$X^{(k)} := \sum_{j=0}^{k} Z^{(j)} = \big[\boldsymbol{\mathcal{U}}_{m_0}^{(0)}\boldsymbol{Y}_L^{(0)} \,|\, \cdots \,|\, \boldsymbol{\mathcal{U}}_{m_k}^{(k)}\boldsymbol{Y}_L^{(k)}\big]\big[\boldsymbol{\mathcal{V}}_{m_0}^{(0)}\boldsymbol{Y}_R^{(0)} \,|\, \cdots \,|\, \boldsymbol{\mathcal{V}}_{m_k}^{(k)}\boldsymbol{Y}_R^{(k)}\big]^*. \tag{2.7}$$

The residual for $X^{(k)}$ is equal to the one provided by the last term $Z^{(k)}$ in the summation. Indeed,

$$\begin{aligned} AX^{(k)} + X^{(k)}B + \boldsymbol{C}\boldsymbol{D}^* &= A\Big(\sum_{j=0}^{k} Z^{(j)}\Big) + \Big(\sum_{j=0}^{k} Z^{(j)}\Big)B + \boldsymbol{C}\boldsymbol{D}^* \\ &= A\Big(\sum_{j=1}^{k} Z^{(j)}\Big) + \Big(\sum_{j=1}^{k} Z^{(j)}\Big)B + R^{(0)} \\ &= \dots \\ &= AZ^{(k)} + Z^{(k)}B + R^{(k-1)} = R^{(k)}. \end{aligned}$$

This relationship can be exploited to design efficient convergence checks within the restarted procedure; see also the discussion in Section 3. See [7] for a similar procedure within the squared Smith method for certain large-scale Smith equations.

An evident shortcoming of the described procedure is that every time we restart, the rank of the residual may double, thus leading to increasingly large Krylov bases that will inevitably exceed memory capacity. This issue can be mitigated by compressing the residual factors before constructing the Krylov subspace in each cycle. For this task, a well-known QR-SVD-based technique can be employed; see, e.g., [35, Section 2.2.1]. We report such a scheme in Algorithm 2.1 for completeness. Note that there is some flexibility in the choice of truncation criterion in line 5.

In this work, we consider two possibilities. Truncating all singular values below a chosen tolerance $\delta > 0$ implies that the spectral norm truncation error is bounded by δ. Alternatively, the Frobenius norm of the error is bounded by δ if we make sure that the Euclidean norm of the truncated singular value remains below δ.

The described compression is not only applied to the residual but also each time when the approximate solution (2.7) is updated. The impact of these compressions on the quality of the computed solution is discussed in Section 3.

Another measure to make sure that memory consumption stays moderate is to impose a maximum number of iterations m_k in each cycle of SKSM. As the memory requirements are primarily dictated by the number of basis vectors that need to be stored in $\boldsymbol{\mathcal{U}}_{m_k}^{(k)}$ and $\boldsymbol{\mathcal{V}}_{m_k}^{(k)}$, we set

$$m_k := \lfloor \mathtt{mem}_{\max}/(2s_k)\rfloor,$$

where $\mathtt{mem}_{\max}$ is the user-defined, maximum number of basis vectors that can be allocated at once, and s_k is the number of columns of the residual factors $\boldsymbol{C}^{(k)}, \boldsymbol{D}^{(k)}$ after truncation.

Algorithm 2.1 Compression of $\boldsymbol{C}\boldsymbol{D}^*$

```
1: procedure COMPRESS($\boldsymbol{C}$, $\boldsymbol{D}$, $\delta$)
2:     Compute economy-size QR decomposition $\boldsymbol{C} = Q_C R_C$
3:     Compute economy-size QR decomposition $\boldsymbol{D} = Q_D R_D$
4:     Compute SVD $R_C R_D^* = U\Sigma V^*$
5:     Truncate $U\Sigma V^* \approx \widetilde{U}\widetilde{\Sigma}\widetilde{V}^*$ up to $\delta$
6:     return $\widetilde{\boldsymbol{C}} := Q_C\widetilde{U}\widetilde{\Sigma}^{1/2}$ and $\widetilde{\boldsymbol{D}} := Q_D\widetilde{V}\widetilde{\Sigma}^{1/2}$
7: end procedure
```

The whole procedure, which combines restarting with compression, is reported in Algorithm 2.2. Note that our pseudocode is not written to optimize memory allocation overall. For best performance, one should a priori estimate the storage needed for the final solution components $\boldsymbol{X}_L$ and $\boldsymbol{X}_R$ and take this into account along with $\texttt{mem}_{\max}$.

2.3. The Lyapunov equation. The Lyapunov equation

$$AX + XA^* + \boldsymbol{C}\boldsymbol{C}^* = 0, \tag{2.8}$$

is an important special case of the more general Sylvester equation (1.1). Indeed, in control and system theory [2], it is more common to find (2.8) than the more general case. In principle, Algorithm 2.2 could be directly applied to solve (2.8). However, as we will see in this section, it is possible to enhance the algorithm by taking into account the specific structure of (2.8).

Thanks to the symmetry of (2.8), general projection techniques for Lyapunov equations generate Hermitian approximations $\widetilde{X} = \boldsymbol{\mathcal{U}}_m \widetilde{Y} \boldsymbol{\mathcal{U}}_m^*$, with range$(\boldsymbol{\mathcal{U}}_m) = \mathcal{K}_m(A, \boldsymbol{C})$ and where the symmetric matrix $\widetilde{Y}$ is computed by solving the projected Lyapunov equation

$$\mathcal{H}_m Y + Y\mathcal{H}_m^* + \big(\boldsymbol{\mathcal{U}}_m^*\boldsymbol{C}\big)\big(\boldsymbol{\mathcal{U}}_m^*\boldsymbol{C}\big)^* = 0, \tag{2.9}$$

with $\mathcal{H}_m = \boldsymbol{\mathcal{U}}_m^* A \boldsymbol{\mathcal{U}}_m$. The norm of the residual matrix $R = A\widetilde{X} + \widetilde{X}A^* + \boldsymbol{C}\boldsymbol{C}^*$ is computed as

$$\|R\|_F = \sqrt{2}\,\|H_{m+1,m}\boldsymbol{E}_m^* Y_m\|_F \;\text{ or }\; \|R\|_2 = \|H_{m+1,m}\boldsymbol{E}_m^* Y_m\|_2\,. \tag{2.10}$$

Employing only one approximation space is quite appealing; it potentially permits skipping the construction of the second Arnoldi relation at line 7 of Algorithm 2.2. However, some additional considerations are needed after the first cycle because the residual matrix $R^{(k)}$ becomes in general indefinite for $k \geq 1$. To address this, the residual is expressed in LDLT form. Following the discussion in the previous section, at the kth restart the residual matrix $R^{(k)} = AZ^{(k)} + Z^{(k)}A^* + R^{(k-1)}$ can be written as

$$\begin{aligned} R^{(k)} &= [\boldsymbol{U}_{m_k+1}^{(k)} H_{m_k+1,m_k} \,|\, \boldsymbol{\mathcal{U}}_{m_k}^{(k)} Y^{(k)} \boldsymbol{E}_{m_k}] \begin{bmatrix} 0 & I \\ I & 0 \end{bmatrix} [\boldsymbol{U}_{m_k+1}^{(k)} H_{m_k+1,m_k} \,|\, \boldsymbol{\mathcal{U}}_{m_k}^{(k)} Y^{(k)} \boldsymbol{E}_{m_k}]^* \\ &= \boldsymbol{C}^{(k+1)} \begin{bmatrix} 0 & I \\ I & 0 \end{bmatrix} \big(\boldsymbol{C}^{(k+1)}\big)^*. \end{aligned} \tag{2.11}$$

Algorithm 2.2 Restarted Sylvester solver

```
1: procedure RESTARTED_SYLV($A$, $B$, $\boldsymbol{C}$, $\boldsymbol{D}$, $\mathtt{mem}_{\max}$, $k_{\max}$, $\varepsilon$, $\delta$)
2:     Initialize $\boldsymbol{X}_L = [\,]$, $\boldsymbol{X}_R = [\,]$, $\boldsymbol{C}^{(0)} = \boldsymbol{C}$, $\boldsymbol{D}^{(0)} = \boldsymbol{D}$, $\mathtt{flag}_{\text{conv}} = 0$
3:     for $k = 0, \dots, k_{\max}$ do
4:         Set $m_k = \lfloor \mathtt{mem}_{\max}/(2s_k) \rfloor - 2$, $s_k = \operatorname{rank}(\boldsymbol{C}^{(k)}) = \operatorname{rank}(\boldsymbol{D}^{(k)})$
5:         for $j = 1, \dots, m_k$ do
6:             Compute (incrementally) the Arnoldi relation for $\mathcal{K}_j(A, \boldsymbol{C}^{(k)})$ and
               store $\boldsymbol{\mathcal{U}}_{j+1}^{(k)} = [\boldsymbol{U}_1 | \cdots | \boldsymbol{U}_{j+1}]$, $\mathcal{H}_j^{(k)}$, and $H_{j+1,j}^{(k)}$
7:             Compute (incrementally) the Arnoldi relation for $\mathcal{K}_j(B, \boldsymbol{D}^{(k)})$ and
               store $\boldsymbol{\mathcal{V}}_{j+1}^{(k)} = [\boldsymbol{V}_1 | \cdots | \boldsymbol{V}_{j+1}]$, $\mathcal{G}_j^{(k)}$, and $G_{j+1,j}^{(k)}$
8:             Compute $Y^{(k)}$ as the solution of the projected equation
                   $\mathcal{H}_j^{(k)} Y + Y \big(\mathcal{G}_j^{(k)}\big)^* + \big(\boldsymbol{\mathcal{U}}_j^{(k)}\big)^* \boldsymbol{C}^{(k)} \big(\boldsymbol{\mathcal{V}}_j^{(k)} \boldsymbol{D}^{(k)}\big)^* = 0$
9:             Compute residual norm $\left\| R_j^{(k)} \right\|$ as in (2.4) or (2.5)
10:            if $\left\| R_j^{(k)} \right\| \le \varepsilon$ then
11:                $m_k \leftarrow j$, $\mathtt{flag}_{\text{conv}} = 1$, and go to 14
12:            end if
13:        end for
14:        Factor $Y^{(k)} = \boldsymbol{Y}_L^{(k)} \big(\boldsymbol{Y}_R^{(k)}\big)^*$
15:        Compute $\boldsymbol{Z}_L^{(k)} = \boldsymbol{\mathcal{U}}_{m_k}^{(k)} \boldsymbol{Y}_L^{(k)}$ and $\boldsymbol{Z}_R^{(k)} = \boldsymbol{\mathcal{V}}_{m_k}^{(k)} \boldsymbol{Y}_R^{(k)}$
16:        Update $\boldsymbol{X}_L = [\boldsymbol{X}_L \,|\, \boldsymbol{Z}_L^{(k)}]$ and $\boldsymbol{X}_R = [\boldsymbol{X}_R \,|\, \boldsymbol{Z}_R^{(k)}]$
17:        $[\boldsymbol{X}_L,\ \boldsymbol{X}_R] \leftarrow$ COMPRESS($\boldsymbol{X}_L$, $\boldsymbol{X}_R$, $\delta$)
18:        if $\mathtt{flag}_{\text{conv}}$ then
19:            return $\boldsymbol{X}_L$ and $\boldsymbol{X}_R$
20:        end if
21:        Set $\boldsymbol{C}^{(k+1)} = [\boldsymbol{U}_{m_k+1}^{(k)} H_{m_k+1,m_k}^{(k)} \,|\, \boldsymbol{\mathcal{U}}_{m_k}^{(k)} Y_{m_k}^{(k)} \boldsymbol{E}_{m_k}]$
22:        Set $\boldsymbol{D}^{(k+1)} = [\boldsymbol{\mathcal{V}}_{m_k}^{(k)} \big(Y_{m_k}^{(k)}\big)^* \boldsymbol{E}_{m_k} \,|\, \boldsymbol{V}_{m_k+1}^{(k)} G_{m_k+1,m_k}^{(k)}]$
23:        $[\boldsymbol{C}^{(k+1)},\ \boldsymbol{D}^{(k+1)}] \leftarrow$ COMPRESS($\boldsymbol{C}^{(k+1)}$, $\boldsymbol{D}^{(k+1)}$, $\delta$)
24:    end for
25: end procedure
```

In turn, the subspace $\mathcal{K}_m(A, \boldsymbol{C}^{(k+1)})$ can be used as an approximation space for the subsequent restart. The presence of the matrix $\begin{bmatrix} 0 & I \\ I & 0 \end{bmatrix}$ only affects the definition of the projected equation solved at line 8, which now takes the form

$$\mathcal{H}_j^{(k)} Y + Y \big(\mathcal{H}_j^{(k)}\big)^* + \widetilde{\boldsymbol{C}} \begin{bmatrix} 0 & I \\ I & 0 \end{bmatrix} \widetilde{\boldsymbol{C}}^* = 0, \qquad \widetilde{\boldsymbol{C}} = \big(\boldsymbol{\mathcal{U}}_j^{(k)}\big)^* \boldsymbol{C}^{(j)}. \tag{2.12}$$

This equation can again be solved with the Bartels-Stewart method or with Hammarling's method [26] after splitting the right-hand side as discussed, e.g., in [6, Sec. 2.3]. In both cases, the Hermitian structure of $\widetilde{Y}$ is preserved and can be exploited.

To avoid the occurrence of complex arithmetic, an LDLT approach must be employed also during the truncation strategy, and we suggest to call Algorithm 2.3 in place of Algorithm 2.1 at lines 17 and 23 of Algorithm 2.2. See, e.g., [37] for similar considerations in the context of differential Riccati equations. Applying the described modifications to Algorithm 2.2 returns an approximate solution of the form $X^{(k)} = \boldsymbol{Z}_L S \boldsymbol{Z}_L^* \approx X$ with a (small) Hermitian matrix S.

Algorithm 2.3 Compression of $\boldsymbol{C}S\boldsymbol{C}^*$

```
1: procedure COMPRESS_SYM($\boldsymbol{C}$, $S$, $\delta$)
2:     Compute Compute economy-size QR decomposition $\boldsymbol{C} = Q_C R_C$
3:     Compute eigendecomposition $R_C S R_C^* = W \Sigma W^*$
4:     Truncate $W \Sigma W^* \approx \widetilde{W}\widetilde{\Sigma}\widetilde{W}^*$ up to $\delta$
5:     return $\widetilde{\boldsymbol{C}} := Q_C \widetilde{W}$ and $\widetilde{S} := \widetilde{\Sigma}$
6: end procedure
```

The final Lyapunov algorithm is stated as Algorithm 2.4.

Algorithm 2.4 Restarted Lyapunov solver

```
1: procedure RESTARTED_LYAP($A$, $\boldsymbol{C}$, $\texttt{mem}_{\max}$, $k_{\max}$, $\varepsilon$, $\delta$)
2:     Initialize $\boldsymbol{X}_L = [\,]$, $\boldsymbol{C}^{(0)} = \boldsymbol{C}$, $\texttt{flag}_{\text{conv}} = 0$, $D = I$
3:     for $k = 0, \dots, k_{\max}$ do
4:         Set $m_k = \lfloor \texttt{mem}_{\max}/s_k \rfloor - 1$, $s_k = \operatorname{rank}(\boldsymbol{C}^{(k)})$
5:         for $j = 1, \dots, m_k$ do
6:             Compute (incrementally) the Arnoldi relation for $\mathcal{K}_j(A, \boldsymbol{C}^{(k)})$ and
               store $\boldsymbol{\mathcal{U}}_{j+1}^{(k)} = [\boldsymbol{U}_1 | \cdots | \boldsymbol{U}_{j+1}]$, $\mathcal{H}_j^{(k)}$, and $H_{j+1,j}^{(k)}$
7:             Compute $Y^{(k)}$ as the solution of the projected equation
                   $$H_j^{(k)} Y + Y \big(H_j^{(k)}\big)^* + \big(\boldsymbol{\mathcal{U}}_j^{(k)}\big)^* \boldsymbol{C}^{(k)} D \big(\boldsymbol{\mathcal{U}}_j^{(k)} \boldsymbol{C}^{(k)}\big)^*,$$
8:             Compute the residual norm $\left\| R_j^{(k)} \right\|$ as in (2.10)
9:             if $\left\| R_j^{(k)} \right\| \le \varepsilon$ then
10:                $m_k \leftarrow j$, $\texttt{flag}_{\text{conv}} = 1$ and go to 13
11:            end if
12:        end for
13:        Factor $Y^{(k)} = \widetilde{Y}^{(k)} S (\widetilde{Y}^{(k)})^*$
14:        Update $\boldsymbol{X}_L = [\boldsymbol{X}_L \,|\, \boldsymbol{\mathcal{U}}_{m_k}^{(k)} \widetilde{Y}^{(k)}]$
15:        $[\boldsymbol{X}_L, S] \leftarrow$ COMPRESS_SYM($\boldsymbol{X}_L$, $\operatorname{diag}(I, S)$, $\delta$)
16:        if $\texttt{flag}_{\text{conv}}$ then
17:            return $\boldsymbol{X}_L$ and $S$
18:        end if
19:        Set $\boldsymbol{C}^{(k+1)} = [\boldsymbol{U}_{m_k+1}^{(k)} H_{m_k+1,m_k}^{(k)} \,|\, \boldsymbol{\mathcal{U}}_{m_k}^{(k)} Y^{(k)} \boldsymbol{E}_{m_k}]$
20:        $[\boldsymbol{C}^{(k+1)}, D] \leftarrow$ COMPRESS_SYM($\boldsymbol{C}^{(k+1)}$, $[0, I; I, 0]$, $\delta$)
21:    end for
22: end procedure
```

2.3.1. Positive semidefinite approximations. A peculiar property of the Lyapunov equation (2.8) is that the solution X is Hermitian positive semidefinite (SPSD) whenever A is stable; in other words, all the eigenvalues of A are in the open left half-plane $\mathbb{C}_-$ [50]. It is desirable to retain this property in an approximate solution. In the context of projection methods for Lyapunov equations, this property is ensured when A is negative definite; i.e., not only A but also its Hermitian part $(A+A^*)/2$ is stable. In particular, in SKSM it would follow that the matrix $\mathcal{H}_m = \boldsymbol{\mathcal{U}}_m^* A \boldsymbol{\mathcal{U}}_m$ is stable for every m and, therefore, that the solution Y of the projected equation $\mathcal{H}_m Y + Y\mathcal{H}_m^* + (\boldsymbol{\mathcal{U}}_m^*\boldsymbol{C})(\boldsymbol{\mathcal{U}}_m^*\boldsymbol{C})^* = 0$, as well as the corresponding approximation $\widetilde{X} = \boldsymbol{\mathcal{U}}_m Y \boldsymbol{\mathcal{U}}_m^*$, are SPSD.

In our framework, the same arguments show that $X^{(0)}$ is SPSD if A is negative definite. However, the subsequent $Z^{(k)}$, $0 < k \le k_{\max}$, are in general indefinite (although still symmetric), due to the indefiniteness of the residual matrices $R^{(k)}$. Nevertheless, it is reasonable to expect the approximate solution $X^{(\bar{k})} = \sum_{k=0}^{\bar{k}} Z^{(k)}$, $0 < \bar{k} \le k_{\max}$, to be close to a positive semidefinite matrix. In particular, if

$$X^{(\bar{k})} = [U_+ \,|\, U_- \,|\, U_0] \begin{bmatrix} \Lambda_+ & & \\ & \Lambda_- & \\ & & 0 \end{bmatrix} [U_+ \,|\, U_- \,|\, U_0]^* \tag{2.13}$$

is the eigendecomposition of $X^{(\bar{k})}$, partitioned according to the sign of the eigenvalues, then one can consider $X_+^{(\bar{k})} := U_+\Lambda_+U_+^*$ as an SPSD approximation to the solution. In practice, $X_+^{(\bar{k})}$ is obtained by applying a slight modification of Algorithm 2.3, which neglects the part of the eigendecomposition corresponding to the negative eigenvalues, to the matrix $X^{(\bar{k})}$ returned by Algorithm 2.2. Such a step might deteriorate the accuracy of the computed solution. However, the next result shows that the error and the residual norm associated with $X_+^{(\bar{k})}$ would be close to the ones associated with $X^{(\bar{k})}$.

LEMMA 2.1. *Let the Lyapunov equation* (2.8) *have the SPSD solution* X*. For a Hermitian approximation* $X^{(\bar{k})}$*, let* $X_+^{(\bar{k})}$ *be defined as in* (2.13) *and set* $R = AX^{(\bar{k})} + X^{(\bar{k})}A^* + \boldsymbol{C}\boldsymbol{C}^*$, $R_+ = AX_+^{(\bar{k})} + X_+^{(\bar{k})}A^* + \boldsymbol{C}\boldsymbol{C}^*$*. Then,*

$$\begin{aligned} \|X - X_+^{(\bar{k})}\| &\le 2\|X - X^{(\bar{k})}\|, \\ \|R_+\| &\le \|R\| + 2\,\|A\|_2\,\|X - X^{(\bar{k})}\|, \end{aligned}$$

where $\|\cdot\|$ *corresponds to the Frobenius or the spectral norm.*

Proof. Because $X^{(\bar{k})}$ is Hermitian, $X_+^{(\bar{k})}$ verifies

$$X_+^{(\bar{k})} = \operatorname{argmin}_{G \text{ is SPSD}} \left\| X^{(\bar{k})} - G \right\|, \tag{2.14}$$

both for the Frobenius and the spectral norm [25, 27]. Therefore,

$$\|X - X_+^{(\bar{k})}\| \le \|X - X^{(\bar{k})}\| + \|X^{(\bar{k})} - X_+^{(\bar{k})}\| \le 2\|X - X^{(\bar{k})}\|,$$

where the last inequality follows from (2.14) by taking into account that X is SPSD.

For the second inequality, applying again (2.14) yields

$$\begin{aligned} \|AX_+^{(\bar{k})} + X_+^{(\bar{k})}A^* + \boldsymbol{C}\boldsymbol{C}^*\| &= \|R - A(X^{(\bar{k})} - X_+^{(\bar{k})}) + (X^{(\bar{k})} - X_+^{(\bar{k})})A^*\| \\ &\le \|R\| + 2\,\|A\|_2\,\|X^{(\bar{k})} - X_+^{(\bar{k})}\| \\ &\le \|R\| + 2\,\|A\|_2\,\|X - X^{(\bar{k})}\|. \qquad \square \end{aligned}$$

3. Residual and error analysis of Algorithm 2.2.

The compression steps performed on the intermediate residuals and solutions introduce some inexactness. In the spirit of the analysis of inexact Krylov methods (as in, e.g., [36, 49]), we study how these compression steps affect the residual and error norms associated with the approximation returned by Algorithm 2.2. The relations retrieved in this section hold for both the Frobenius norm and the spectral norm.

Let us suppose that Algorithm 2.2 terminates after $\bar{k} < k_{\max}$ restarts, so that $\|R^{(\bar{k})}\| < \varepsilon$. Notice that the returned approximation $X^{(\bar{k})}$ satisfies

$$X^{(\bar{k})} = \sum_{j=0}^{\bar{k}} (Z^{(j)} - \Delta Z^{(j)}),$$

where the matrices $\Delta Z^{(j)}$ represent the components removed by the compression step at line 17. In particular, it holds that $\|\Delta Z^{(j)}\| \le \delta$ for $j = 0, \ldots, \bar{k}$.

Similarly, the sequence of residuals verifies

$$AZ^{(j)} + Z^{(j)}B + R^{(j-1)} - \Delta R^{(j-1)} = R^{(j)}, \qquad j = 0, \ldots, \bar{k}, \tag{3.1}$$

where $\Delta R^{(j)}$ takes into account the effect of the compression step at line 23; i.e., $\|\Delta R^{(j)}\| \le \delta$, $j = 0, \ldots, \bar{k}$.

Summing up (3.1) for $j = 0, \ldots, \bar{k}$, we retrieve

$$AX^{(\bar{k})} + X^{(\bar{k})}B + \boldsymbol{C}\boldsymbol{D}^* = R^{(\bar{k})} + \sum_{j=0}^{\bar{k}} \Delta R^{(j)} - A\sum_{j=0}^{\bar{k}} \Delta Z^{(j)} - \sum_{j=0}^{\bar{k}} \Delta Z^{(j)} B.$$

Therefore, the residual norm associated with $X^{(\bar{k})}$ is bounded by

$$\left\| AX^{(\bar{k})} + X^{(\bar{k})}B + \boldsymbol{C}\boldsymbol{D}^* \right\| \le \varepsilon + (\bar{k}+1)(\|A\| + \|B\| + 1)\delta. \tag{3.2}$$

Equation (3.2) shows how the truncation tolerance δ is connected with the final attainable accuracy from our restarted routine. Indeed, a reasonable way to choose δ in Algorithm 2.2 is such that $(k_{\max}+1)(\|A\| + \|B\| + 1)\delta \le \varepsilon$.

We conclude by estimating the distance from the true solution X. To this end, we remark that the difference $X^{(\bar{k})} + \sum_{j=0}^{\bar{k}} \Delta Z^{(j)} - X$ exactly solves the Sylvester equation

$$A\Big(X^{(\bar{k})} + \sum_{j=0}^{\bar{k}} \Delta Z^{(j)} - X\Big) + \Big(X^{(\bar{k})} + \sum_{j=0}^{\bar{k}} \Delta Z^{(j)} - X\Big)B = R^{(\bar{k})} + \sum_{j=0}^{\bar{k}} \Delta R^{(j)}. \tag{3.3}$$

The impact of perturbing the right-hand side on the solution can be estimated via the norm of the inverse of the solution operator. This is particularly simple, when A, B are normal and the spectra of A, $-B$ are separated by a vertical line in the complex plane.

Lemma 3.1. *Let $A, B \in \mathbb{C}^{n\times n}$ be normal matrices with eigenvalues $\lambda_{A,j}$ and $\lambda_{B,j}$, such that*

$$0 \le \mathrm{Re}(\lambda_{A,1}) \le \cdots \le \mathrm{Re}(\lambda_{A,n}), \qquad 0 \le \mathrm{Re}(\lambda_{B,1}) \le \cdots \le \mathrm{Re}(\lambda_{B,n}).$$

If Algorithm 2.2 terminates after $\bar{k} < k_{max}$ iterations then the returned solution $X^{(\bar{k})}$ verifies

$$\|X^{(\bar{k})} - X\| \le \frac{1}{\mathrm{Re}(\lambda_{A,1}) + \mathrm{Re}(\lambda_{B,1})}(\varepsilon + (\bar{k}+1)\delta) + (\bar{k}+1)\delta.$$

Proof. By applying Theorem 1.1 in [30] to the equation (3.3) we get

$$\left\| X^{(\bar{k})} + \sum_{j=0}^{\bar{k}} \Delta Z^{(j)} - X \right\| \leq \frac{1}{\mathrm{Re}(\lambda_{A,1}) + \mathrm{Re}(\lambda_{B,1})} (\varepsilon + (\bar{k}+1)\delta).$$

The claim follows by using the triangular inequality and the bound $\left\| \sum_{j=0}^{\bar{k}} \Delta Z^{(j)} \right\| \leq (\bar{k}+1)\delta$. ▯

4. Numerical experiments. To demonstrate the efficiency of our proposed methods, we examine their behavior with respect to other viable methods on two standard problems where the coefficient matrices stem from the discretization of certain differential operators. These other methods include the Extended Krylov Subspace Method (EKSM) (detailed, e.g., in [14, 47] but implemented like a Rational KSM with poles at zero and infinity) and the Standard Krylov Subspace Method (SKSM) of [41], which is tailored to equations with symmetric coefficient matrices.

EKSM can be implemented "exactly" in the sense that linear systems with A are solved at very high accuracy via, e.g., a direct sparse solver like the MATLAB *backslash* operator; or "inexactly," in which inversions of A are computed approximately via a block Krylov subspace method. We test both of these variants of EKSM. Instead of using *backslash* explicitly per call to A^{-1}, we precompute and store the Cholesky or LU factorization. In the particular case of iterative solves by A, we use a retooled (and possibly ILU preconditioned) block conjugate gradients (BCG) method from [19] when A is Hermitian positive definite, and (possibly ILU preconditioned) block GMRES otherwise. We employ a rather small tolerance on the relative residual norm when BCG and block GMRES are applied to solve the linear systems with A. In particular, we set this tolerance to 10^{-8}, namely two orders of magnitude smaller that the outer tolerance on the relative residual norm. However, the novel results about inexact procedures in the basis construction of extended (and rational) Krylov subspaces presented in [36] may be adopted to further reduce the computational cost of the basis construction. We also remind the reader that at each EKSM iteration $2s$ new basis vectors are added to the current basis so that, at the mth EKSM iteration, the computed extended Krylov subspace has dimension $2(m+1)s$.

We would like to underscore that the comparisons between our compress-and-restart scheme, the inexact variants of EKSM, and SKSM are the fairest. Indeed, in these families of methods, only the action of A on (block) vectors is allowed, making all these solvers potentially matrix-free. This is not the case for EKSM with a direct inner solver.

All methods make use of block Krylov subspace techniques and, consequently, sparse-matrix-matrix multiplication (SpMM) between A and block vectors $\boldsymbol{V}$ that could vary in size. Since the performance of such block operations depends on machine architecture (in particular, the level 3 cache [20]), memory hierarchies, and choice of libraries, we measure the performance of each method via the number calls to A ("A-calls") and the total number of columns or column vectors to which A is applied, which we refer here to as `matvecs`.[3] The number of A-calls is a crude measure of memory operations, whereas the number of `matvecs` is directly related to the amount of floating-point operations (FLOPs). The ratio between FLOPs and memory operations is known as *computational intensity*, and algorithms with high

[3]Note that, strictly speaking, a `matvec` is usually defined as the application of A to a single column vector $\boldsymbol{v}$.

computational intensity, i.e., many FLOPs to memory operations, are preferable for high-performance architectures; see, e.g., [4]. We approximate computational intensity by looking at the ratio between A-calls and `matvecs`, which we here refer to as "efficiency." For a more in-depth analysis of the performance of block operations and potential gains over column-by-column applications of A, see, e.g., the thesis by Birk [13].

Unless otherwise noted, all reported residual norms are relative and measured in the Frobenius norm. For all experiments, the residual tolerance is set to 10^{-6}.

All results were obtained by running MATLAB R2017b [39] on a standard node of the Linux cluster Mechthild hosted at the Max Planck Institute for Dynamics of Complex Technical Systems in Magdeburg, Germany.[4]

The full code to reproduce our numerical results can be found https://gitlab.com/katlund/compress-and-restart-KSM. For the original version of SKSM, which we have adapted, see https://zenodo.org/record/3252320#.XjM3UN-YVuR.

4.1. 2D Laplacian. In this example, A is the second-order, centered, finite-difference discretization of the two-dimensional Laplacian operator $-(\partial_{xx} + \partial_{yy})$ on the unit square unit cube $\Omega = (0,1)^2$. We take $n = 100$ grid points in each direction, resulting in a matrix of size $10,000 \times 10,000$. The matrix A is Hermitian positive definite. We solve

$$AX + XA + \boldsymbol{C}\boldsymbol{C}^* = 0,$$

where $\boldsymbol{C} \in \mathbb{C}^{n^2 \times 3}$ is drawn from a normal random distribution, and it is such that $\|\boldsymbol{C}\boldsymbol{C}^*\|_F = 1$. In this example we call RESTART_LYAP (Algorithm 2.4) and equipped with the enhancements described in section 2.3.

Both the exact and inexact variants of EKSM need 15 iterations to meet the prescribed accuracy. As a result, a low-dimensional extended Krylov subspace of dimension 96 is constructed. We mimic such a feature by setting the memory buffer of the compress-and-restart procedure, i.e., $\texttt{mem}_{\max}$, equal to 96. We report the results in Table 4.1.

	Its (Restarts)	rank($X^{(k)}$)	A-calls	`matvecs`	efficiency	Time (s)
RESTARTED_LYAP	158 (20)	53	158	1845	11	4.02
EKSM (BCG)	15 (–)	56	2615	7845	3	8.29
EKSM (BCG+ILU)	15 (–)	56	900	2700	3	4.17
EKSM (exact)	15 (–)	56	30	90	3	0.34
SKSM (two-pass Lanczos)	148 (–)	65	295	885	3	4.08

TABLE 4.1
Example 4.1. Performance measures. $s = 3$, $\texttt{mem}_{\max} = 96$.

Our routine RESTARTED_LYAP needs 20 restarts to converge for a total number of 158 iterations. The compress-and-restart scheme lets us maintain a low storage demand and, at each restart, a polynomial Krylov subspace of dimension (at most) 96 is constructed, as in the case of the comparable EKSM. In SKSM with two-pass Lanczos, thanks to the symmetry of A, we can use short-term recurrences so that the whole basis is never stored, only three basis vectors at a time. The low-rank factors of the solution are recovered by means of a two-pass strategy; see [41] for more details. Despite the very low storage demands of SKSM, the number of A-calls exceeds that of the compress-and-restart scheme.

[4]See https://www.mpi-magdeburg.mpg.de/cluster/mechthild for further details.

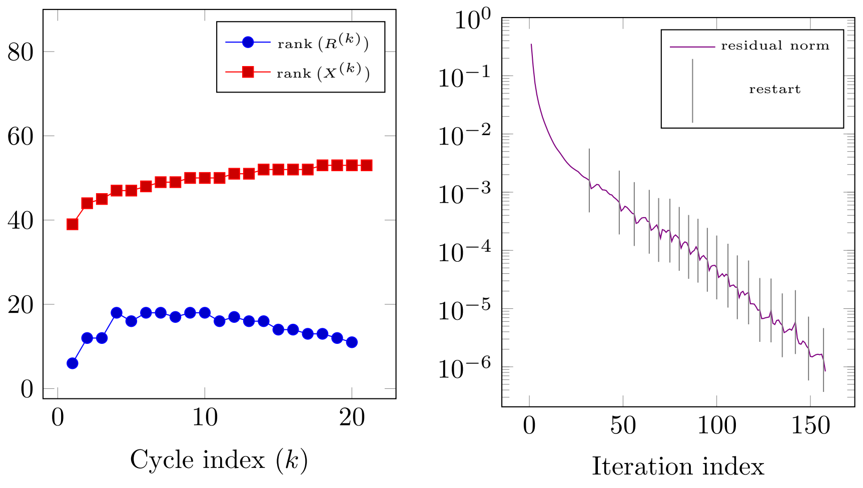

FIG. 4.1. *Example 4.1. Residual and solution ranks (left) and residual norms (right) for the compress-and-restart polynomial Krylov method. Vertical tick marks indicate the start of a new cycle.*

Both the numbers of A-calls and `matvecs` of RESTARTED_LYAP are much lower than those accrued by the inexact procedures EKSM (BCG) and EKSM (BCG+ILU). Moreover, our procedure is more efficient for block operations while maintaining a computational time comparable to that of the other SpMM-dominant routines, thus reinforcing its potential for further speed-ups in communication-dominant high-performance environments. Timings for EKSM (exact) largely benefit from having precomputed and stored the Cholesky factors of A.

All the routines we tested return a low-rank numerical solution and, for this example, the approximation computed by RESTARTED_LYAP has the lowest rank. In Figure 4.1 (left) we report the ranks of both the residual and the approximate solution computed at the end of each restart, together with the rank of the final solution. These results illustrate that the truncation strategy COMPRESS_SYM (Algorithm 2.3) is able to maintain a moderate rank in the residual $R^{(k)}$, for all $k = 0, \ldots, 20$. This is crucial for making the construction of the subsequent restart space feasible, when necessary. In Figure 4.1 (right) we also plot, in logarithmic scale, the relative residual norm history through all the 158 iterations performed by RESTARTED_LYAP. We can see that the relative residual norm does not have a smooth behavior. This is due to the a Galerkin condition we impose on the residual. Even though this phenomenon has not been extensively analyzed in the matrix equation literature yet, it is quite well understood in the linear system setting. See, e.g., [16, 15]. Imposing a minimal residual condition in place of a Galerkin condition might be beneficial, although such a strategy has some peculiar shortcomings in the matrix equation framework. See, e.g., [38, 31, 42].

We now illustrate some observations about the possible computation of an indefinite approximate solution. See also the end of section 2.3. In Figure 4.2 (left) the blue circles denote the minimum nonzero eigenvalue of $X^{(k)}$ for $k = 0, \ldots, 20$, and of the final solution. The black dashed line marks the x-axis. We can appreciate how these eigenvalues are all positive so that $X^{(k)}$ is positive semidefinite for all k.

We now consider the same Lyapunov equation as before but we do not normalize

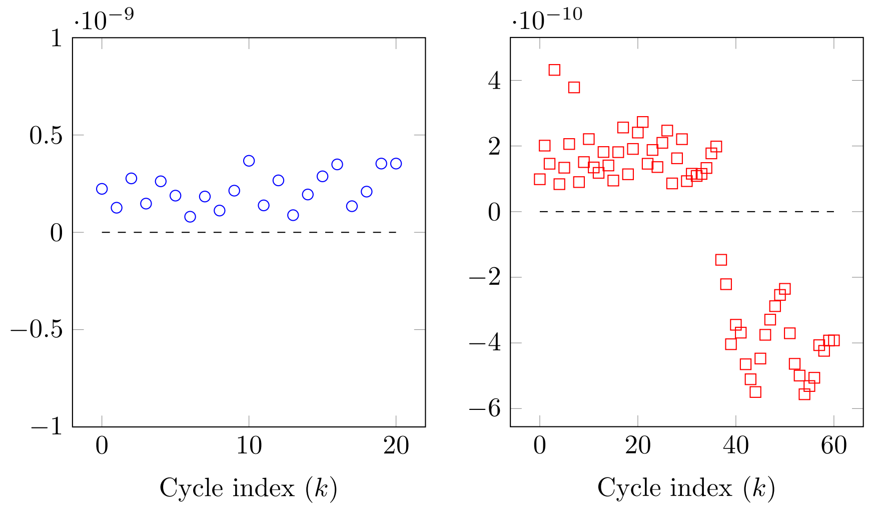

FIG. 4.2. *Example 4.1. Smallest nonzero eigenvalue of $X^{(k)}$ in the case of a normalized right-hand side (left) and non-normalized right-hand side (right).*

the random matrix $\boldsymbol{CC}^*$. This is now a harder problem for the polynomial Krylov subspace method and, with $\texttt{mem}_{\max} = 96$, RESTARTED_LYAP needs 60 restarts to converge.The large number of restarts is due to the large rank of $R^{(k)}$, especially for large k, which limits us to a couple of iterations per restart in order to stay within the memory buffer prescribed by $\texttt{mem}_{\max}$. In Figure 4.2 (right) we report the minimum nonzero eigenvalue of $X^{(k)}$ for $k = 0, \dots, 60$. The matrix $X^{(k)}$ stops being positive semidefinite for $k \geq 39$, and we thus apply the strategy presented at the end of section 2.3 to compute $X_+^{(k)}$, $k = 60$, in place of $X^{(k)}$.For this example, such a strategy has multiple advantages. Indeed, in addition to providing a positive semidefinite approximate solution, it reduces the rank of the computed solution while maintaining the prescribed accuracy. In particular, $\operatorname{rank}(X^{(k)}) = 81$ while $\operatorname{rank}\left(X_+^{(k)}\right) = 77$ and $\left\| AX_+^{(k)} + X_+^{(k)}A^* + \boldsymbol{CC}^* \right\|_F / \|\boldsymbol{CC}^*\|_F = 8.84 \times 10^{-7}$.

We conclude this example by showing one of the most remarkable features of our novel compress-and-restart strategy. The total memory demand of a Krylov subspace method is in general difficult to predict a priori, as it requires knowing the dimension of the subspace in which a satisfactory approximation can be found. If a situation calls for stringent memory management, then methods that increase the basis size every iteration, like EKSM, may not be able to reach the desired accuracy before exhausting memory resources. Table 4.2 demonstrates the superiority of RESTARTED_LYAP in precisely such a scenario, where $\texttt{mem}_{\max} = 250$ and the right-hand side $\boldsymbol{CC}^*$, $\boldsymbol{C} \in \mathbb{R}^{n^2 \times s}$, $s = 25$, is a low-rank approximation of the matrix $C \in \mathbb{R}^{n^2 \times n^2}$ representing the discretization of $\exp((x_1^p + x_2^p + x_3^p + x_4^p)^{1/p})$, $p = 2$, on the hypercube $[-1, 1]^4$. All the EKSM variants are forced to stop as soon as a space of dimension 250 is constructed; in tests not reported here, we found that $\texttt{mem}_{\max} = 400$ allows EKSM to reach the desired residual tolerance. SKSM with two-pass Lanczos seems to suffer from the large rank of $\boldsymbol{C}$. Indeed, the computed residual norm differs from the actual one by some orders of magnitude, likely due to the loss of orthogonality in the computed basis. A full, or perhaps even partial, re-orthogonalization of the basis

	Its (Restarts)	Rel. Res.
RESTARTED_LYAP	165 (33)	7.06×10^{-7}
EKSM (BCG)	5 (–)	1.51×10^{-4}
EKSM (BCG+ILU)	5 (–)	1.60×10^{-4}
EKSM (exact)	5 (–)	1.43×10^{-4}
SKSM (two-pass Lanczos)	69 (–)	5.76×10^{-4}

TABLE 4.2
Example 4.1. Performance measures. $s = 25$, $\texttt{mem}_{\max} = 250$.

may fix this issue but doing so in a memory-sensitive manner remains open. On the other hand, RESTARTED_LYAP successfully reaches the desired residual tolerance, thus demonstrating its potential in not only memory-limited situations but also for matrix equations whose right-hand side has high rank.

4.2. Convection-diffusion equation. We turn our attention to the main problem, a Sylvester equation of the form (1.1), where the coefficient matrices A and B stem from the second-order, centered, finite-difference discretization of the 3D convection-diffusion operators

$$\mathcal{L}_A(u) = -\varepsilon\Delta u + \vec{w}_A \cdot \nabla u, \quad \mathcal{L}_B(u) = -\varepsilon\Delta u + \vec{w}_B \cdot \nabla u, \tag{4.1}$$

on the unit cube $\Omega = (0,1)^3$, respectively. The viscosity parameter is $\varepsilon = 0.01$ while the convection vectors $\vec{w}_A$, $\vec{w}_B$ are defined as

$$\vec{w}_A = (x\sin(x), y\cos(y), e^{z^2-1}), \quad \vec{w}_B = (yz(1-x^2), 0, e^z).$$

If the operators in (4.1) are discretized with n equidistant nodes in each direction, then the nonsymmetric matrices A and B are each of dimension n^3. We consider two different problem sizes for (1.1), $n = 25$ and $n = 80$. The resulting problems are of dimension 15625 and 512000, respectively. The low-rank matrices $\boldsymbol{C}, \boldsymbol{D} \in \mathbb{C}^{n^3 \times 3}$ are once again drawn from a normal random distribution, and they are such that $\|\boldsymbol{C}\boldsymbol{D}^*\|_F = 1$.

For $n = 25$, all the EKSM variants we tested– EKSM (exact), EKSM (BGMRES), and EKSM (BGMRES+ILU)– need 21 iterations to converge, in which case, two extended Krylov subspaces of dimension 132 are constructed. As before, we set the memory buffer of our compress-and-restart routine equal to the memory consumption of EKSM; i.e., we set $\texttt{mem}_{\max}$ in Algorithm 2.2 equal to 264. With this setting, RESTARTED_SYLV needs 2 restarts for a total of 85 iterations to converge, and it computes an approximate solution of rank 57, equal to the rank of the solution returned by all the EKSM variants.

In Figure 4.3 (left) we depict the ranks of both the residual and of the approximate solution computed at each RESTARTED_SYLV restart, together with the rank of the final solution, while in Figure 4.3 (right), the entire relative residual norm history is reported. We again see that the low-rank compression procedure (Algorithm 2.1) is able to maintain a moderate rank in both the residual and the approximate solution.

We remind the reader that two different subspaces have to be generated, one by A and the other by B, when Sylvester equations are solved by projection techniques. The computational efforts devoted to such tasks may significantly differ from each other if

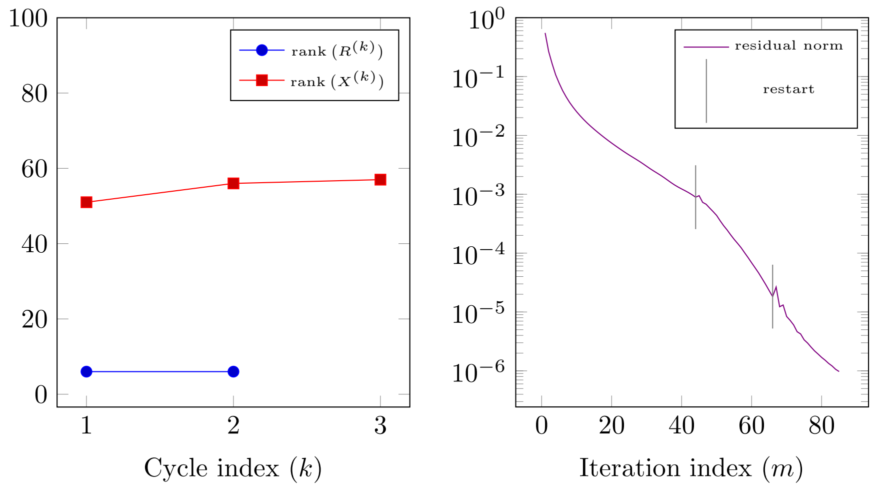

FIG. 4.3. *Example 4.2, $n = 25$. Residual and solution ranks (left) and residual norms (right) for the compress-and-restart polynomial Krylov method. Vertical tick marks indicate the start of a new cycle.*

	A-calls	`matvecs`	efficiency	B-calls	`matvecs`	efficiency
RESTARTED_SYLV	85	378	4	85	378	4
EKSM (BGMRES)	1819	5457	3	2445	7335	3
EKSM (BGMRES+ILU)	455	1365	3	376	1128	3
EKSM (exact)	42	126	3	42	126	3

TABLE 4.3
Example 4.2. Performance measures. $n = 25$, $s = 3$, $\texttt{mem}_{\max} = 264$.

the matrices A and B have dissimilar spectral properties. This can be appreciated by looking at the results in Table 4.3. In this example, iteratively solving linear systems with B requires more iterations than with A. Therefore, a larger number of B-calls is required in EKSM (BGMRES), even though the extended Krylov subspaces generated by A and B have the same dimensions. This phenomenon is reversed in EKSM (BGMRES+ILU), indicating that the ILU preconditioner for B performs better than the one for A. Our compress-and-restart procedure is not influenced by such issues, though, since by design, the spaces for A and B are computed to the same dimension (assuming no breakdowns). It is indeed possible, and in some cases desirable, to allow for different basis sizes for A and B, especially if the operators differ significantly in size or complexity. However, this flexibility is not trivial to implement, especially with the compression at step 23 of Algorithm 2.2, which could cause $\boldsymbol{C}^{(k+1)}$ and $\boldsymbol{D}^{(k+1)}$ to have different numbers of columns. For the simplicity of presentation and implementation, we therefore do not explore this option further in the present work.

The extra memory allocation required by the iterative solution of linear systems with A and B during the basis construction must be taken into account for precisely identifying the memory demands of EKSM (BGMRES) and EKSM (BGMRES+ILU). To the best of our knowledge, such an issue has not yet been rigorously explored in the literature, and it should not be underestimated. Increasing the density of the discretization grid to $n = 80$ (for a problem size of 512000), leads to 26 iterations for

	Time (s)	
	$n = 25$, $\texttt{mem}_{\max} = 264$	$n = 80$, $\texttt{mem}_{\max} = 324$
RESTARTED_SYLV	5.96	549.99
EKSM (BGMRES)	298.15	–
EKSM (BGMRES+ILU)	14.62	–
EKSM (exact)	3.44	589.72

TABLE 4.4
Example 4.2. Computational times for different values of n.

EKSM (exact) to converge and the construction of two extended Krylov subspaces of dimension 162 each. With $\texttt{mem}_{\max} = 324$, EKSM (BGMRES) and EKSM (BGMRES+ILU) are not able to achieve the prescribed accuracy, because at some point, the number of (outer) extended Krylov basis vectors already computed plus the number of vectors needed by the (inner) block polynomial Krylov subspace to accurately compute the next (outer) basis vector exceeds $\texttt{mem}_{\max}$.

We conclude this example by pointing out that RESTARTED_SYLV turns out to be competitive also in terms of computational time for both $n = 25$ and $n = 80$; see Table 4.4. Indeed, a preconditioner tailored to the problem may benefit EKSM with inexact solves, but the design of an effective preconditioner is a difficult task and largely problem-dependent. Our compress-and-restart procedure achieves excellent performance without the need for preconditioning while still managing severe memory limitations.

5. Conclusions.

Modern computing architectures pose many challenges demanding the optimization of not only operation counts (FLOPs) but also memory allocation and movement. Much work has been devoted to adapting iterative methods for large and sparse linear systems to these architectures, but straightforward extensions of successful strategies for linear systems to matrix equations are not always feasible. We have demonstrated how to apply a common and effective Krylov subspace technique for linear systems, namely restarts, by introducing a compression step to mitigate the growing rank of the residual. The resulting compress-and-restart method is viable for both Sylvester and Lyapunov matrix equations, and given a fixed memory requirement, it tunes the computable basis size automatically at each restart. Compared to extended Krylov subspace methods (EKSM), which require an inner solver for applications of A^{-1} and therefore either well-designed preconditioners or fast sparse Cholesky and LU factorizations, our compress-and-restart polynomial Krylov methods require very little set-up and converge with competitive timings in situations where (unrestarted) EKSM run out of memory.

Our compress-and-restart method is a success not only for matrix equations but potentially other classes of higher-order problems where memory resources are even more limited, due to the curse of dimensionality. Moreover, our compress-and-restart paradigm is not restricted to the use of block polynomial Krylov subspaces; different approximation spaces can be used as well.

Acknowledgments. The third and the fourth authors are members of the Italian INdAM Research group GNCS.

REFERENCES

[1] M. I. AHMAD, I. M. JAIMOUKHA, AND M. FRANGOS, *Krylov Subspace Restart Scheme for*

Solving Large-Scale Sylvester Equations, in 2010 Am. Control Conf. Marriott Waterfront, Balt. MD, 2010, pp. 5726–5731.

[2] A. C. Antoulas, *Approximation of Large-Scale Dynamical Systems*, SIAM Publications, Philadelphia, PA, 2005.

[3] J. Baker, M. Embree, and J. Sabino, *Fast singular value decay for Lyapunov solutions with nonnormal coefficients*, SIAM J. Matrix Anal. Appl., 36 (2015), pp. 656–668.

[4] G. Ballard, E. C. Carson, J. W. Demmel, M. Hoemmen, N. Knight, and O. Schwartz, *Communication lower bounds and optimal algorithms for numerical linear algebra*, Acta Numer., 23 (2014), pp. 1–155, https://doi.org/10.1017/S0962492914000038.

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

[6] P. Benner, P. Ezzatti, D. Kressner, E. S. Quintana-Ortí, and A. Remón, *A mixed-precision algorithm for the solution of Lyapunov equations on hybrid CPU-GPU platforms*, Parallel Comput., 37 (2011), pp. 439–450, https://doi.org/10.1016/j.parco.2010.12.002.

[7] P. Benner, G. E. Khoury, and M. Sadkane, *On the squared Smith method for large-scale Stein equations*, Numerical Linear Algebra with Applications, 21 (2014), pp. 645–665, https://doi.org/10.1002/nla.1918.

[8] P. Benner, P. Kürschner, and J. Saak, *Self-generating and efficient shift parameters in ADI methods for large Lyapunov and Sylvester equations*, Electron. Trans. Numer. Anal., 43 (2014), pp. 142–162.

[9] P. Benner, R.-C. Li, and N. Truhar, *On the ADI method for Sylvester equations*, J. Comput. Appl. Math., 233 (2009), pp. 1035–1045, https://doi.org/10.1016/j.cam.2009.08.108.

[10] P. Benner and J. Saak, *Numerical solution of large and sparse continuous time algebraic matrix Riccati and Lyapunov equations: a state of the art survey*, GAMM-Mitt., 36 (2013), pp. 32–52, https://doi.org/10.1002/gamm.201310003.

[11] D. A. Bini, B. Iannazzo, and B. Meini, *Numerical solution of algebraic Riccati equations*, vol. 9 of Fundamentals of Algorithms, SIAM, Philadelphia, PA, 2012.

[12] A. Binning, *Solving second and third-order approximations to DSGE models: A recursive Sylvester equation solution*, Norges Bank Working Paper 18, 2013.

[13] S. Birk, *Deflated shifted block Krylov subspace methods for Hermitian positive definite matrices*, PhD thesis, Fakultät für Mathematik und Naturwissenschaften, Bergische Universität Wuppertal, 2015.

[14] T. Breiten, V. Simoncini, and M. Stoll, *Low-rank solvers for fractional differential equations*, Electron. Trans. Numer. Anal., 45 (2016), pp. 107–132.

[15] J. K. Cullum, *Peaks, plateaus, numerical instabilities in a Galerkin minimal residual pair of methods for solving $Ax = b$*, Applied Numerical Mathematics, 19 (1995), pp. 255 – 278, https://doi.org/10.1016/0168-9274(95)00086-0.

[16] J. K. Cullum and A. Greenbaum, *Relations between Galerkin and norm-minimizing iterative methods for solving linear systems*, SIAM Journal on Matrix Analysis and Applications, 17 (1996), pp. 223–247, https://doi.org/10.1137/S0895479893246765.

[17] T. A. Davis, S. Rajamanickam, and W. M. Sid-Lakhdar, *A survey of direct methods for sparse linear systems*, Acta Numer., 25 (2016), pp. 383–566, https://doi.org/10.1017/S0962492916000076.

[18] V. Druskin and V. Simoncini, *Adaptive rational Krylov subspaces for large-scale dynamical systems*, Systems Control Lett., 60 (2011), pp. 546–560, https://doi.org/10.1016/j.sysconle.2011.04.013.

[19] A. A. Dubrulle, *Retooling the method of block conjugate gradients*, Electron. Trans. Numer. Anal., 12 (2001), pp. 216–233.

[20] I. S. Duff, M. Marrone, G. Radicati, and C. Vittoli, *Level 3 basic linear algebra subprograms for sparse matrices: a user-level interface*, ACM Trans. Math. Softw., 23 (1997), pp. 379–401, https://doi.org/10.1145/275323.275327.

[21] N. S. Ellner et al., *New ADI model problem applications*, in Proceedings of 1986 ACM Fall joint computer conference, IEEE Computer Society Press, 1986, pp. 528–534.

[22] A. Frommer, K. Lund, and D. B. Szyld, *Block Krylov subspace methods for functions of matrices*, Electron. Trans. Numer. Anal., 47 (2017), pp. 100–126.

[23] L. Grasedyck, *Existence of a low rank or $\mathcal{H}$-matrix approximant to the solution of a Sylvester equation*, Numer. Linear Algebra Appl., 11 (2004), pp. 371–389.

[24] M. H. Gutknecht, *Block Krylov space methods for linear systems with multiple right-hand sides: An introduction*, in Mod. Math. Model. Methods Algorithms Real World Syst., A. H. Siddiqi, I. S. Duff, and O. Christensen, eds., New Delhi, 2007, Anamaya, pp. 420–447.

[25] P. R. Halmos, *Positive approximants of operators*, Indiana Univ. Math. J., 21 (1971/72), pp. 951–960, https://doi.org/10.1512/iumj.1972.21.21076.

[26] S. Hammarling, *Numerical solution of the stable, nonnegative definite Lyapunov equation*, IMA J. Numer. Anal., 2 (1982), pp. 303–323.

[27] N. J. Higham, *Computing a nearest symmetric positive semidefinite matrix*, Linear Algebra Appl., 103 (1988), pp. 103–118, https://doi.org/10.1016/0024-3795(88)90223-6.

[28] N. J. Higham, *Accuracy and Stability of Numerical Algorithms*, SIAM, Philadelphia, PA, second ed., 2002.

[29] A. S. Hodel, B. Tenison, and K. R. Poolla, *Numerical solution of the Lyapunov equation by approximate power iteration*, Linear Algebra Appl., 236 (1996), pp. 205–230.

[30] R. A. Horn and F. Kittaneh, *Two applications of a bound on the Hadamard product with a Cauchy matrix*, vol. 3, 1998, pp. 4–12, https://doi.org/10.13001/1081-3810.1010.

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

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

[33] I. M. Jaimoukha and E. M. Kasenally, *Krylov subspace methods for solving large Lyapunov equations*, SIAM J. Numer. Anal., 31 (1994), pp. 227–251, https://doi.org/10.1137/0731012.

[34] D. Kressner, *Memory-efficient Krylov subspace techniques for solving large-scale Lyapunov equations*, Proc. IEEE Int. Symp. Comput. Control Syst. Des., (2008), pp. 613–618, https://doi.org/10.1109/CACSD.2008.4627370.

[35] D. Kressner and C. Tobler, *Low-rank tensor Krylov subspace methods for parametrized linear systems*, SIAM J. Matrix Anal. Appl., 32 (2011), pp. 1288–1316, https://doi.org/10.1137/100799010.

[36] P. Kürschner and M. Freitag, *Inexact methods for the low rank solution to large scale Lyapunov equations*, (2018). ArXiv preprint: 1809.06903.

[37] N. Lang, H. Mena, and J. Saak, *On the benefits of the LDL^T factorization for large-scale differential matrix equation solvers*, Linear Algebra Appl., 480 (2015), pp. 44–71, https://doi.org/10.1016/j.laa.2015.04.006.

[38] Y. Lin and V. Simoncini, *Minimal residual methods for large scale Lyapunov equations*, Appl. Numer. Math., 72 (2013), pp. 52–71, https://doi.org/10.1016/j.apnum.2013.04.004.

[39] MATLAB, *version 9.3.0.713579 (R2017b)*, The MathWorks Inc., Natick, Massachusetts, 2017.

[40] D. Palitta and V. Simoncini, *Matrix-equation-based strategies for convection-diffusion equations*, BIT, 56 (2016), pp. 751–776, https://doi.org/10.1007/s10543-015-0575-8.

[41] D. Palitta and V. Simoncini, *Computationally enhanced projection methods for symmetric Sylvester and Lyapunov matrix equations*, J. Comput. Appl. Math., 330 (2018), pp. 1–16, https://doi.org/10.1016/j.cam.2017.08.011, https://arxiv.org/abs/1602.05033.

[42] D. Palitta and V. Simoncini, *Optimality properties of Galerkin and Petrov-Galerkin methods for linear matrix equations*, (2019). To appear in Vietnam Journal of Mathematics.

[43] T. Penzl, *Eigenvalue decay bounds for solutions of Lyapunov equations: The symmetric case*, Syst. Control Lett., 40 (2000), pp. 139–144, https://doi.org/10.1016/S0167-6911(00)00010-4.

[44] Y. Saad, *Numerical solution of large Lyapunov equations*, in Signal processing, scattering and operator theory, and numerical methods (Amsterdam, 1989), vol. 5 of Progr. Systems Control Theory, Birkhäuser Boston, Boston, MA, 1990, pp. 503–511.

[45] Y. Saad and M. H. Schultz, *GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 7 (1986), pp. 856–869, https://doi.org/10.1137/0907058.

[46] S. A. Sauter and C. Schwab, *Boundary element methods*, vol. 39 of Springer Series in Computational Mathematics, Springer-Verlag, Berlin, 2011, https://doi.org/10.1007/978-3-540-68093-2.

[47] V. Simoncini, *A new iterative method for solving large-scale Lyapunov matrix equations*, SIAM J. Sci. Comput., 29 (2007), pp. 1268–1288, https://doi.org/10.1137/06066120X.

[48] V. Simoncini, *Computational methods for linear matrix equations*, SIAM Rev., 58 (2016), pp. 377–441, https://doi.org/10.1137/130912839.

[49] V. Simoncini and D. B. Szyld, *Theory of inexact Krylov subspace methods and applications to scientific computing*, SIAM Journal on Scientific Computing, 25 (2003), pp. 454–477.

[50] J. Snyders and M. Zakai, *On nonnegative solutions of the equation $AD + DA' = -C$*, SIAM J. Appl. Math., 18 (1970), pp. 704–714, https://doi.org/10.1137/0118063.

[51] K. M. Soodhalter, *Stagnation of block GMRES and its relationship to block FOM*, Electron. Trans. Numer. Anal., 46 (2017), pp. 162–189.

[52] D. C. Sorensen and A. C. Antoulas, *The Sylvester equation and approximate balanced reduction*, vol. 351/352, 2002, pp. 671–700, https://doi.org/10.1016/S0024-3795(02)00283-5.

Fourth special issue on linear systems and control.

[53] K. Zhou, J. C. Doyle, and K. Glover, *Robust and Optimal Control*, Prentice-Hall, Upper Saddle River, NJ, 1996.