Received: 4 February 2020 | Revised: 7 July 2020 | Accepted: 31 August 2020

DOI: 10.1002/nla.2339

RESEARCH ARTICLE

WILEY

Compress-and-restart block Krylov subspace methods for Sylvester matrix equations

Daniel Kressner[1] | **Kathryn Lund**[2] | **Stefano Massei**[3] | **Davide Palitta**[4]

[1]Institute for Mathematics, EPF Lausanne, Lausanne, Switzerland

[2]Department of Numerical Mathematics, Faculty of Mathematics and Physics, Charles University, Prague, Czech Republic

[3]Centre for Analysis, Scientific Computing and Applications (CASA), TU/e, Eindhoven, The Netherlands

[4]Research Group Computational Methods in Systems and Control Theory (CSC), Max Planck Institute for Dynamics of Complex Technical Systems, Magdeburg, Germany

Correspondence
Davide Palitta, Research Group Computational Methods in Systems and Control Theory (CSC), Max Planck Institute for Dynamics of Complex Technical Systems, Magdeburg, Germany.
Email: palitta@mpi-magdeburg.mpg.de

Funding information
SNSF: Fast algorithms from low-rank updates, grant number: 200020 178806; Charles University, Grant/Award Number: PRIMUS/19/SCI/11

Summary

Block Krylov subspace methods (KSMs) comprise building blocks in many state-of-the-art solvers for large-scale matrix equations as they arise, for example, 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.

KEYWORDS

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

MOS SUBJECT CLASSIFICATION

65F10; 65N22; 65J10; 65F30; 65F50

1 | INTRODUCTION

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

$$AX + XB + \boldsymbol{C}\boldsymbol{D}^* = 0, \tag{1}$$

with coefficient matrices $A, B \in \mathbb{R}^{n\times n}$, and $\boldsymbol{C}, \boldsymbol{D} \in \mathbb{R}^{n\times s}$. The superscript * in (1) denotes the conjugate transpose of a matrix. 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 article, we will assume $s \ll n$ and view $\boldsymbol{C},\boldsymbol{D}$ as block vectors. This not only implies that the constant term $\boldsymbol{C}\boldsymbol{D}^*$ 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, for example, References 1-3.

The Sylvester equation (1) arises in a variety of applications. In particular, model reduction in Reference 4 and robust/optimal control[5] 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) under suitable separability and smoothness assumptions.[6] Another source of applications are linearizations of nonlinear problems, such as the algebraic Riccati equation,[7] and dynamic stochastic general equilibrium models,[8] where the solution of (1) is needed in iterative solvers. We refer to the surveys[9,10] 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,[11] or when A, B are actually dense, as for discretized boundary integral equations.[12] In these situations, the methods available for solving (1) essentially boil down to Krylov subspace methods. One of the most common variants, due to Saad[13] and Jaimoukha and Kasenally,[14] 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.[15,16] 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, for example, FOM[17] or GMRES.[18] In principle, the idea of restarting can be directly applied to Sylvester equations because (1) can be recast as a linear system of size $n^2 \times n^2$. However, such an approach inevitably involves the construction of a subspace of $\mathbb{R}^{n^2}$ with a consequent increase in both the memory consumption and the computational effort of the overall scheme. Moreover, neglecting the matrix equation formulation of the algebraic problem (1) does not lead to benefits.[9,10] In particular, Krylov subspace methods applied to the $n^2 \times n^2$ linear system stemming from (1) often compute a subspace which contains some spectral redundancy, and a delay in the convergence of the adopted iterative procedure is frequently detected; see, for example, Reference 19, section 4. In this work, we propose and analyze an algorithm that constructs only subspaces of $\mathbb{R}^n$, as it is customary in projection methods for Sylvester equations, and combines restarting with compression.

The *ADI method*[20] and *rational Krylov subspace methods (RKSM)*[21] 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 Reference 22. 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 References 23,24 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)*.[25,26] EKSM is a particular case of RKSM, where half of the shifts are set to 0 and the remaining are set to ∞. Other 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[27] and the restarted parameterized Arnoldi method in Reference 14.

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 Reference 28, 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 Reference 29, 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 article 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) seek an approximate solution of the form

$$\tilde{X} = \mathcal{U}_m \tilde{Y} \mathcal{V}_m^*, \tag{2}$$

where the columns of $\mathcal{U}_m$ and $\mathcal{V}_m$ form orthonormal bases of suitably chosen (low-dimensional) subspaces. The small core factor $\tilde{Y}$ is determined by, for example, imposing a Galerkin condition on the residual matrix $R = A\tilde{X} + \tilde{X}B + \boldsymbol{C}\boldsymbol{D}^*$; see, for example, Reference 10.

In SKSM, the subspaces determining (2) are block Krylov subspaces. More specifically, $\mathcal{U}_m = [\boldsymbol{U}_1 | \cdots | \boldsymbol{U}_m]$, $\mathcal{V}_m = [\boldsymbol{V}_1 | \cdots | \boldsymbol{V}_m] \in \mathbb{R}^{n \times ms}$, $\boldsymbol{U}_i, \boldsymbol{V}_i \in \mathbb{R}^{n \times s}$, and $i = 1, \ldots, m$, are such that

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

and, analogously, $\operatorname{range}(\mathcal{V}_m) = \mathcal{K}_m(B^*, \boldsymbol{D})$.[1] If the block Arnoldi process is used for obtaining these bases, then the following block Arnoldi relations hold:

$$A\mathcal{U}_m = \mathcal{U}_m \mathcal{H}_m + \boldsymbol{U}_{m+1} H_{m+1,m} \boldsymbol{E}_m^*, \quad B^* \mathcal{V}_m = \mathcal{V}_m \mathcal{G}_m + \boldsymbol{V}_{m+1} G_{m+1,m} \boldsymbol{E}_m^*, \tag{3}$$

with $ms \times ms$ block Hessenberg matrices $\mathcal{H}_m$ and $\mathcal{G}_m$; $s \times s$ matrices $H_{m+1,m}$ and $G_{m+1,m}$; $\boldsymbol{E}_m^* = [0_s | \ \ldots \ | 0_s | I_s] = \boldsymbol{e}_m^* \otimes I_s$, where 0_s and I_s denote the $s \times s$ zero and identity matrices, respectively. We refer to Reference 30 for more details on block Krylov subspaces. The matrix $\tilde{Y} \in \mathbb{R}^{ms \times ms}$ is determined from the Galerkin condition

$$0 = \mathcal{U}_m^* R \mathcal{V}_m = \mathcal{U}_m^* (A\mathcal{U}_m \tilde{Y} \mathcal{V}_m^* + \mathcal{U}_m \tilde{Y} \mathcal{V}_m^* B + \boldsymbol{C}\boldsymbol{D}^*) \mathcal{V}_m.$$

Combined with the relations (3), this yields the projected equation

$$0 = \mathcal{H}_m \tilde{Y} + \tilde{Y} \mathcal{G}_m^* + \left(\mathcal{U}_m^* \boldsymbol{C}\right) \left(\mathcal{V}_m^* \boldsymbol{D}\right)^*, \tag{4}$$

[1] 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; see, for example, Reference 30. 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 $\mathbb{R}^n$.

which is again a Sylvester equation of the form (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,[31] can be employed for its solution.

Throughout the above discussion, we assumed that none of the block basis vectors $\boldsymbol{U}_i$ or $\boldsymbol{V}_i$, $i = 1, \dots, m$, 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, for example, References 32,33.

Having the solution $\tilde{Y}$ of (4) at hand, we can compute the Frobenius norm of the residual matrix at a low cost as explained in References 10,14

$$\|R\|_F^2 = \left\| H_{m+1,m} \boldsymbol{E}_m^* \tilde{Y} \right\|_F^2 + \left\| \tilde{Y} \boldsymbol{E}_m G_{m+1,m}^* \right\|_F^2. \tag{5}$$

Similarly, for the spectral norm we have

$$\|R\|_2 = \max \left\{ \left\| H_{m+1,m} \boldsymbol{E}_m^* \tilde{Y} \right\|_2, \left\| \tilde{Y} \boldsymbol{E}_m G_{m+1,m}^* \right\|_2 \right\}. \tag{6}$$

Once the residual norm is sufficiently small, the approximation $\tilde{X}$ from (2) is returned. Note that $\tilde{X}$ is a large, dense matrix, which is always kept in factored form. Specifically, given a factorization $\tilde{Y} = \boldsymbol{Y}_L \boldsymbol{Y}_R^*$, which can be computed, for example, via a truncated singular value decomposition (SVD), we store the low-rank factors $\boldsymbol{X}_L = \mathscr{U}_m \boldsymbol{Y}_L$ and $\boldsymbol{X}_R = \mathscr{V}_m \boldsymbol{Y}_R$ of $\tilde{X} = \boldsymbol{X}_L \boldsymbol{X}_R^*$, 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

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

Note that we have dropped the subscripts on $\mathscr{U}$ and $\mathscr{V}$ and instead use the superscript $^{(0)}$ to denote both the restart cycle index and associated basis size m_0.

For a linear system, a correction to a given approximate solution is obtained by solving the linear system (approximately) with the constant term replaced by the residual; see, for example, Reference 34. Adding the exact solution of this so-called correction equation to the approximate solution would yield the exact solution of the linear system. Applying this principle to (1), one first solves the correction equation

$$AZ^{(1)} + Z^{(1)}B + R^{(0)} = 0, \tag{7}$$

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 obtained correction $Z^{(1)}$ to $X^{(0)}$ in order to obtain $X^{(1)}$. As its exact solution would be too expensive, the correction equation (7) is only solved approximately.

The Arnoldi relations (3) imply that the residual matrix $R^{(0)}$ admits the following low-rank representation:

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

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 approximating the solution of (7). In turn, this can be viewed as restarting the computations. We emphasize, however, that the next Krylov subspaces will be different from the ones used to build $X^{(0)}$, and we therefore

denote them with the superscript $^{(1)}$. After $m_1 \le m_0$ iterations, SKSM applied to the correction equation (7) constructs orthonormal bases $\mathcal{U}^{(1)}$ and $\mathcal{V}^{(1)}$ 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 $Z^{(1)} = \boldsymbol{Z}_L^{(1)}\left(\boldsymbol{Z}_R^{(1)}\right)^*$, with $Y^{(1)} = \boldsymbol{Y}_L^{(1)}\left(\boldsymbol{Y}_R^{(1)}\right)^*$, $\boldsymbol{Z}_L^{(1)} = \mathcal{U}^{(1)}\boldsymbol{Y}_L^{(1)}$, $\boldsymbol{Z}_R^{(1)} = \mathcal{V}^{(1)}\boldsymbol{Y}_R^{(1)}$.

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

$$X^{(k)} := \sum_{j=0}^{k} Z^{(j)} = \left[\mathcal{U}^{(0)}\boldsymbol{Y}_L^{(0)} \mid \cdots \mid \mathcal{U}^{(k)}\boldsymbol{Y}_L^{(k)}\right] [\mathcal{V}^{(0)}\boldsymbol{Y}_R^{(0)} \mid \cdots \mid \mathcal{V}^{(k)}\boldsymbol{Y}_R^{(k)}]^*, \tag{8}$$

where $\mathcal{U}^{(k)}$ and $\mathcal{V}^{(k)}$ have each been generated by m_k iterations of SKSM.

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

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

This relationship can be exploited to design efficient convergence checks within the restarted procedure; see also the discussion in Section 3. See Reference 35 for a similar procedure within the squared Smith method for certain large-scale Stein 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, for example, Reference 36, section 2.2.1. We report such a scheme in Algorithm 1 for completeness. Note that there is some flexibility in the choice of the 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 (8) is updated. The impact of these compressions on the quality of the computed solution is discussed in Section 3.

Algorithm 1. Compression of $\boldsymbol{C}\boldsymbol{D}^*$

1: **procedure** COMPRESS($\boldsymbol{C}$, $\boldsymbol{D}$, δ)
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 δ
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**

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 $\mathcal{U}^{(k)}$ and $\mathcal{V}^{(k)}$, 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.

The whole procedure, which combines restarting with compression, is reported in Algorithm 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 $\boldsymbol{X}_R$ (e.g., using the results in Reference 37), the number of restarts, and take this into account along with $\texttt{mem}_{\max}$. Providing a bound for the number of restarts is still an open problem.

Finally, we remark that keeping the solution and the residual in factorized form allows Algorithm 2 to deal with the case of square coefficients A and B of different sizes, with essentially no changes. Indeed, both the generation of the orthonormal bases and the compression routine do not assume that the final solution has a square shape.

Algorithm 2. Restarted Sylvester solver

procedure RESTARTED_SYLV(A, B, $\boldsymbol{C}$, $\boldsymbol{D}$, $\texttt{mem}_{\max}$, $k_{\max}$, ε, δ)
  Initialize $\boldsymbol{X}_L = [\,]$, $\boldsymbol{X}_R = [\,]$, $\boldsymbol{C}^{(0)} = \boldsymbol{C}$, $\boldsymbol{D}^{(0)} = \boldsymbol{D}$, $\texttt{flag}_{\text{conv}} = 0$
  for $k = 0, \dots, k_{\max}$ **do**
    Set $m_k = \lfloor \texttt{mem}_{\max}/(2s_k) \rfloor - 2$, $s_k = \operatorname{rank}(\boldsymbol{C}^{(k)}) = \operatorname{rank}(\boldsymbol{D}^{(k)})$
    for $j = 1, \dots, m_k$ **do**
      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 $H_{j+1,j}^{(k)}$
      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 $G_{j+1,j}^{(k)}$
      Compute $Y^{(k)}$ as the solution of the projected equation

$$\mathcal{H}_j^{(k)} Y + Y\left(\mathcal{G}_j^{(k)}\right)^* + \left(\boldsymbol{\mathcal{U}}_j^{(k)}\right)^* \boldsymbol{C}^{(k)} \left(\boldsymbol{D}^{(k)}\right)^* \boldsymbol{\mathcal{V}}_j^{(k)} = 0$$

      Compute residual norm $\left\| R_j^{(k)} \right\|$ as in (5) or (6)
      if $\left\| R_j^{(k)} \right\| \le \varepsilon$ **then**
        $m_k \leftarrow j$, $\texttt{flag}_{\text{conv}} = 1$, and go to 14
      end if
    end for
    Factor $Y^{(k)} = \boldsymbol{Y}_L^{(k)} \left(\boldsymbol{Y}_R^{(k)}\right)^*$
    Compute $\boldsymbol{Z}_L^{(k)} = \boldsymbol{\mathcal{U}}^{(k)} \boldsymbol{Y}_L^{(k)}$ and $\boldsymbol{Z}_R^{(k)} = \boldsymbol{\mathcal{V}}^{(k)} \boldsymbol{Y}_R^{(k)}$
    Update $\boldsymbol{X}_L = [\boldsymbol{X}_L \,|\, \boldsymbol{Z}_L^{(k)}]$ and $\boldsymbol{X}_R = [\boldsymbol{X}_R \,|\, \boldsymbol{Z}_R^{(k)}]$
    $[\boldsymbol{X}_L, \boldsymbol{X}_R] \leftarrow$ COMPRESS($\boldsymbol{X}_L, \boldsymbol{X}_R, \delta$)
    if $\texttt{flag}_{\text{conv}}$ **then**
      return $\boldsymbol{X}_L$ and $\boldsymbol{X}_R$
    end if
    Set $\boldsymbol{C}^{(k+1)} = [\boldsymbol{U}_{m_k+1}^{(k)} H_{m_k+1,m_k}^{(k)} \,|\, \boldsymbol{\mathcal{U}}^{(k)} Y^{(k)} \boldsymbol{E}_{m_k}]$
    Set $\boldsymbol{D}^{(k+1)} = [\boldsymbol{\mathcal{V}}^{(k)} \left(Y^{(k)}\right)^* \boldsymbol{E}_{m_k} \,|\, \boldsymbol{V}_{m_k+1}^{(k)} G_{m_k+1,m_k}^{(k)}]$
    $[\boldsymbol{C}^{(k+1)}, \boldsymbol{D}^{(k+1)}] \leftarrow$ COMPRESS($\boldsymbol{C}^{(k+1)}, \boldsymbol{D}^{(k+1)}, \delta$)
  end for
end procedure

2.3 | The Lyapunov equation

The Lyapunov equation

$$AX + XA^* + \boldsymbol{C}\boldsymbol{C}^* = 0, \tag{9}$$

is an important special case of the more general Sylvester equation (1). Indeed, in control and system theory,[4] it is more common to find (9) than the more general case. In principle, Algorithm 2 could be directly applied to solve (9).

However, as we will see in this section, it is possible to enhance the algorithm by taking into account the specific structure of (9).

Thanks to the symmetry of (9), general projection techniques for Lyapunov equations generate Hermitian approximations $\tilde{X} = \mathcal{U}_m \tilde{Y} \mathcal{U}_m^*$, with $\operatorname{range}(\mathcal{U}_m) = \mathcal{K}_m(A, \boldsymbol{C})$ and where the symmetric matrix $\tilde{Y}$ is computed by solving the projected Lyapunov equation

$$\mathcal{H}_m Y + Y \mathcal{H}_m^* + \left(\mathcal{U}_m^* \boldsymbol{C}\right)\left(\mathcal{U}_m^* \boldsymbol{C}\right)^* = 0, \tag{10}$$

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

$$\|R\|_F = \sqrt{2}\|H_{m+1,m}\boldsymbol{E}_m^* Y_m\|_F \quad \text{or} \quad \|R\|_2 = \|H_{m+1,m}\boldsymbol{E}_m^* Y_m\|_2. \tag{11}$$

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. 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 symmetric factorized 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

$$\begin{aligned} R^{(k)} &= [\boldsymbol{U}_{m_k+1}^{(k)} H_{m_k+1,m_k} \mid \mathcal{U}^{(k)} Y^{(k)} \boldsymbol{E}_{m_k}] \begin{bmatrix} 0 & I \\ I & 0 \end{bmatrix} [\boldsymbol{U}_{m_k+1}^{(k)} H_{m_k+1,m_k} \mid \mathcal{U}^{(k)} Y^{(k)} \boldsymbol{E}_{m_k}]^* \\ &= \boldsymbol{C}^{(k+1)} \begin{bmatrix} 0 & I \\ I & 0 \end{bmatrix} \left(\boldsymbol{C}^{(k+1)}\right)^*. \end{aligned} \tag{12}$$

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

$$\mathcal{H}_j^{(k)} Y + Y\left(\mathcal{H}_j^{(k)}\right)^* + \tilde{\boldsymbol{C}} \begin{bmatrix} 0 & I \\ I & 0 \end{bmatrix} \tilde{\boldsymbol{C}}^* = 0, \quad \tilde{\boldsymbol{C}} = \left(\mathcal{U}_j^{(k)}\right)^* \boldsymbol{C}^{(j)}. \tag{13}$$

This equation can again be solved with the Bartels–Stewart method or with Hammarling's method[38] after splitting the constant term as discussed, for example, in Reference 39, sec. 2.3. In both cases, the Hermitian structure of $\tilde{Y}$ is preserved and can be exploited.

To avoid the occurrence of complex arithmetic, a symmetric factorization approach must be employed also during the truncation strategy, and we suggest to call Algorithm 3 in place of Algorithm 1 at lines 17 and 23 of Algorithm 2. See, for example, Reference 40 for similar considerations in the context of differential Riccati equations. Applying the described modifications to Algorithm 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 3. Compression of $\boldsymbol{C}S\boldsymbol{C}^*$

1: **procedure** COMPRESS_SYM($\boldsymbol{C}$, S, δ)
2: 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 δ
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 4.

Algorithm 4. Restarted Lyapunov solver

procedure RESTARTED_LYAP(A, $\boldsymbol{C}$, $\texttt{mem}_{\max}$, $k_{\max}$, ε, δ)
  Initialize $\boldsymbol{X}_L = [\,]$, $\boldsymbol{C}^{(0)} = \boldsymbol{C}$, $\texttt{flag}_{\text{conv}}\texttt{flag}_{\text{conv}} = 0$, $D = I$
  for $k = 0, \dots, k_{\max}$ **do**
    Set $m_k = \lfloor \texttt{mem}_{\max}/s_k \rfloor - 1$, $s_k = \operatorname{rank}(\boldsymbol{C}^{(k)})$
    for $j = 1, \dots, m_k$ **do**
      Compute (incrementally) the Arnoldi relation for $\mathcal{K}_j(A, \boldsymbol{C}^{(k)})$ and store $\boldsymbol{\mathcal{U}}^{(k)}_{j+1} = [\boldsymbol{U}_1 | \cdots | \boldsymbol{U}_{j+1}]$, $\mathcal{H}^{(k)}_j$, and $H^{(k)}_{j+1,j}$
      Compute $Y^{(k)}$ as the solution of the projected equation

$$H^{(k)}_j Y + Y\big(H^{(k)}_j\big)^* + \big(\boldsymbol{\mathcal{U}}^{(k)}_j\big)^* \boldsymbol{C}^{(k)} D \big(\boldsymbol{C}^{(k)}\big)^* \boldsymbol{\mathcal{U}}^{(k)}_j = 0,$$

      Compute the residual norm $\left\| R^{(k)}_j \right\|$ as in (11)
      if $\|R^{(k)}_j\| \le \varepsilon$ **then**
        $m_k \leftarrow j$, $\texttt{flag}_{\text{conv}}\texttt{flag}_{\text{conv}} = 1$ and go to 13
      end if
    end for
    Factor $Y^{(k)} = \widetilde{Y}^{(k)} S (\widetilde{Y}^{(k)})^*$
    Update $\boldsymbol{X}_L = [\boldsymbol{X}_L \mid \boldsymbol{\mathcal{U}}^{(k)} \widetilde{Y}^{(k)}]$
    $[\boldsymbol{X}_L, S] \leftarrow$ COMPRESS_SYM($\boldsymbol{X}_L$, $\operatorname{diag}(I, S)$, δ)
    if $\texttt{flag}_{\text{conv}}\texttt{flag}_{\text{conv}}$ **then**
      return $\boldsymbol{X}_L$ and S
    end if
    Set $\boldsymbol{C}^{(k+1)} = [\boldsymbol{U}^{(k)}_{m_k+1} H^{(k)}_{m_k+1,m_k} \mid \boldsymbol{\mathcal{U}}^{(k)} Y^{(k)} \boldsymbol{E}_{m_k}]$
    $[\boldsymbol{C}^{(k+1)}, D] \leftarrow$ COMPRESS_SYM($\boldsymbol{C}^{(k+1)}$, $[0, I; I, 0]$, δ)
  end for
end procedure

2.3.1 | Positive semidefinite approximations

A peculiar property of the Lyapunov equation (9) 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 $\mathbb{C}_-$.[41] 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, that is, 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 $\mathcal{H}_m Y + Y\mathcal{H}_m^* + (\boldsymbol{\mathcal{U}}_m^* \boldsymbol{C})\big(\boldsymbol{\mathcal{U}}_m^* \boldsymbol{C}\big)^* = 0$, as well as the corresponding approximation $\tilde{X} = \boldsymbol{\mathcal{U}}_m Y \boldsymbol{\mathcal{U}}_m^*$, 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

$$X^{(\bar{k})} = [U_+ \mid U_- \mid U_0] \begin{bmatrix} \Lambda_+ & & \\ & \Lambda_- & \\ & & 0 \end{bmatrix} [U_+ \mid U_- \mid U_0]^* \tag{14}$$

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 3, which neglects the part of the eigendecomposition corresponding to the negative eigenvalues, to the matrix

$X^{(\bar{k})}$ returned by Algorithm 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 1. *Let the Lyapunov equation (9) have the SPSD solution X. For a Hermitian approximation $X^{(\bar{k})}$, let $X_+^{(\bar{k})}$ be defined as in (14) 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,*

$$\left\|X - X_+^{(\bar{k})}\right\| \leq 2\left\|X - X^{(\bar{k})}\right\|,$$
$$\|R_+\| \leq \|R\| + 2\|A\|_2 \left\|X - X^{(\bar{k})}\right\|,$$

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

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

$$X_+^{(\bar{k})} = \operatorname{argmin}_{G \text{ is SPSD}} \|X^{(\bar{k})} - G\|, \tag{15}$$

both for the Frobenius and the spectral norm.[42,43] Therefore,

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

where the last inequality follows from (15) by taking into account that X is SPSD.

For the second inequality, applying again (15) yields

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

■

3 | RESIDUAL AND ERROR ANALYSIS OF ALGORITHM 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., References 22,44), we study how these compression steps affect the residual and error norms associated with the approximation returned by Algorithm 2. The relations retrieved in this section hold for both the Frobenius norm and the spectral norm.

Let us suppose that Algorithm 2 terminates after $\bar{k} < k_{\max}$ restarts, so that $\left\|R^{(\bar{k})}\right\| < \varepsilon$. 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 $\|\Delta Z^{(j)}\| \leq \delta$ for $j = 0, \ldots, \bar{k}$.

Similarly, the sequence of residuals verifies

$$AZ^{(j)} + Z^{(j)}B + R^{(j-1)} - \Delta R^{(j-1)} = R^{(j)}, \quad j = 0, \ldots, \bar{k}, \tag{16}$$

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

Summing up (16) 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

$$\|AX^{(\bar{k})} + X^{(\bar{k})}B + \boldsymbol{C}\boldsymbol{D}^*\| \leq \varepsilon + (\bar{k}+1)(\|A\| + \|B\| + 1)\delta. \tag{17}$$

Equation (17) shows how the truncation tolerance δ is connected with the final attainable accuracy from our restarted routine. In turn, it is reasonable to choose δ in Algorithm 2 such that $(k_{\max}+1)(\|A\| + \|B\| + 1)\delta \leq \varepsilon$.

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

$$A\left(X^{(\bar{k})} + \sum_{j=0}^{\bar{k}} \Delta Z^{(j)} - X\right) + \left(X^{(\bar{k})} + \sum_{j=0}^{\bar{k}} \Delta Z^{(j)} - X\right)B = R^{(\bar{k})} + \sum_{j=0}^{\bar{k}} \Delta R^{(j)}. \tag{18}$$

The impact of perturbing the constant term 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 2. *Let $A, B \in \mathbb{R}^{n\times n}$ be normal matrices with eigenvalues $\lambda_{A,j}$ and $\lambda_{B,j}$, such that*

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

If Algorithm 2 terminates after $\bar{k} < k_{max}$ iterations then the returned solution $X^{(\bar{k})}$ verifies

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

Proof. By applying Theorem 1.1 in Reference 45 to Equation (18) 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 References 25,26 but implemented like a rational KSM with poles at zero and infinity) and the standard Krylov subspace method (SKSM) of Reference 46, 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, for example, 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 ICHOL preconditioned) block conjugate gradients (BCG) method from Reference 47 when A is Hermitian positive definite, and (possibly ILU preconditioned) block GMRES otherwise[2]. 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

[2]Both the ICHOL and ILU preconditioners have been computed with zero fill-in. Moreover, the computational time spent on their construction is not reported in the tables that follow. In all experiments, this has been negligible compared to the total solution time.

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 Reference 22 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[48]), 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 computational intensity, that is, many FLOPs to memory operations, are preferable for high-performance architectures; see, for example, Reference 49. 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, for example, the thesis by Birk[32]; see also the PAPI library, which allows users to profile a routine's performance in real-time.[50]

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[51] 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 at 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 $\Omega=(0,1)^2$. 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 $\boldsymbol{C}\in\mathbb{R}^{n^2\times 3}$ is drawn from a normal random distribution, and it is such that $\|\boldsymbol{C}\boldsymbol{C}^*\|_F = 1$. In this example we call RESTARTED_LYAP (Algorithm 4) 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, that is, $\texttt{mem}_{\max}$, equal to 96. We report the results in Table 1.

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 Reference 46 for more details. Despite the very low storage demands of SKSM, the number of A-calls exceeds that of the compress-and-restart scheme.

[3]Note that, strictly speaking, a `matvec` is usually defined as the application of A to a single column vector $\boldsymbol{v}$.
[4]See https://www.mpi-magdeburg.mpg.de/cluster/mechthild for further details.

TABLE 1 2D Laplacian from Section 4.1

	Its (restarts)	Rank($X^{(k)}$)	A-calls	matvecs	Efficiency	Time (s)
RESTARTED_LYAP	158 (20)	53	158	1845	11	3.88
EKSM (BCG)	15 (—)	56	3328	9984	3	10.52
EKSM (BCG+ICHOL)	15 (—)	56	1109	3327	3	5.14
EKSM (exact)	15 (—)	56	30	90	3	0.37
SKSM (two-pass Lanczos)	148 (—)	65	295	885	3	4.23

Note: Performance measures for RESTARTED_LYAP with $\texttt{mem}_{\max} = 96$ compared with EKSM and SKSM.

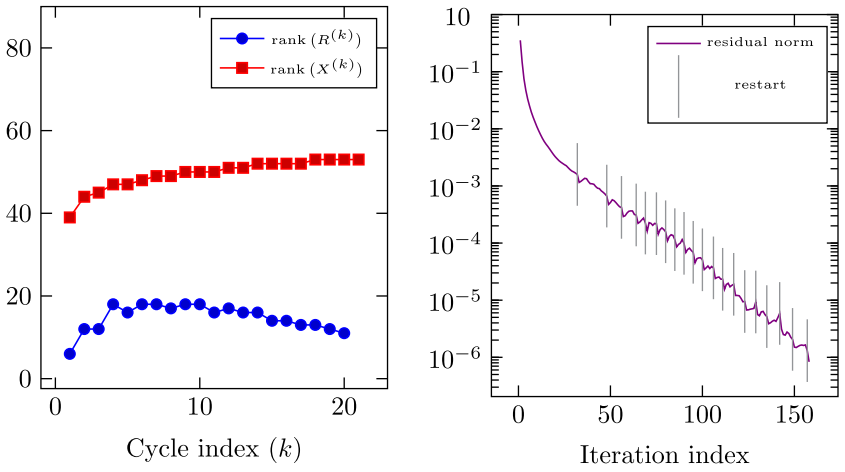

FIGURE 1 2D Laplacian from Section 4.1. Residual and solution ranks (left) and residual norms (right) for RESTARTED_LYAP. Vertical tick marks indicate the start of a new cycle

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 + ICHOL). 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 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 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 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 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, for example, References 52,53. 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 setting. In particular, the computation of the matrix Y is much more involved and computationally expensive. Moreover, its semidefiniteness is not guaranteed even in case of a stable matrix $\mathcal{H}_m$ and a semidefinite constant term; see, for example, References 19,54,55.

We now illustrate some observations about the possible computation of an indefinite approximate solution. See also the end of Section 2.3. In Figure 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 the random matrix $\boldsymbol{C}\boldsymbol{C}^*$. 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 2 (right) we report the

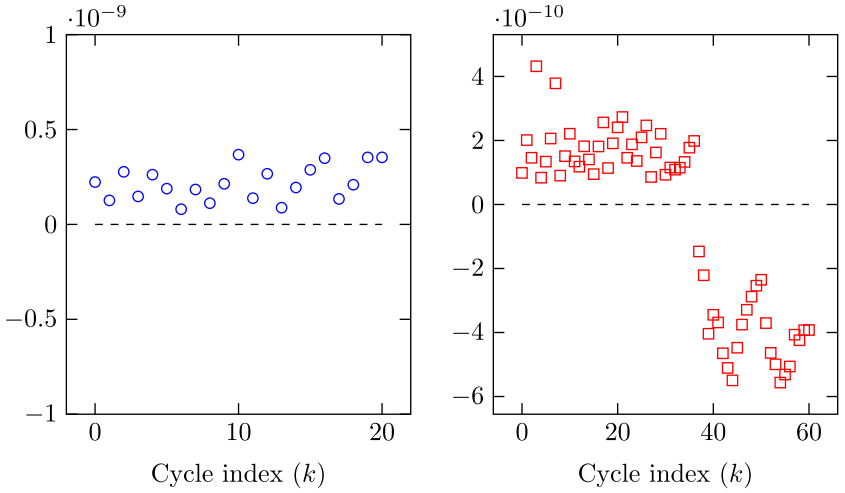

FIGURE 2 2D Laplacian from Section 4.1. Smallest nonzero eigenvalue of approximation $X^{(k)}$ computed by RESTARTED_LYAP in the case of a normalized constant term (left) and nonnormalized constant term (right)

TABLE 2 2D Laplacian from Section 4.1

	Its (restarts)	Rel. res.
RESTARTED_LYAP	165 (33)	7.06×10^{-7}
EKSM (BCG)	5 (—)	1.51×10^{-4}
EKSM (BCG+ICHOL)	5 (—)	1.60×10^{-4}
EKSM (exact)	5 (—)	1.43×10^{-4}
SKSM (two-pass Lanczos)	69 (—)	5.76×10^{-4}

Note: Number of total iterations, restarts, as well as relative residual for RESTARTED_LYAP with $\texttt{mem}_{\max} = 250$ compared with EKSM and SKSM.

minimum nonzero eigenvalue of $X^{(k)}$ for $k = 0, \ldots, 60$. 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 $\|AX_+^{(k)} + X_+^{(k)}A^* + \boldsymbol{C}\boldsymbol{C}^*\|_F / \|\boldsymbol{C}\boldsymbol{C}^*\|_F = 8.84 \times 10^{-7}$.

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 2 demonstrates the superiority of RESTARTED_LYAP in precisely such a scenario, where $\texttt{mem}_{\max} = 250$ and the constant term $\boldsymbol{C}\boldsymbol{C}^*$, $\boldsymbol{C} \in \mathbb{R}^{n^2 \times s}$, $s = 25$, 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, reorthogonalization of the basis 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 constant term has high rank.

4.2 | Convection–diffusion equation

We turn our attention to the main problem, a Sylvester equation of the form (1), where the coefficient matrices A and B stem from the second-order, centered, finite-difference discretization of the 3D convection–diffusion operators

$$\mathcal{L}_A(u) = -\varepsilon \Delta u + \vec{w}_A \cdot \nabla u, \quad \mathcal{L}_B(u) = -\varepsilon \Delta u + \vec{w}_B \cdot \nabla u, \tag{19}$$

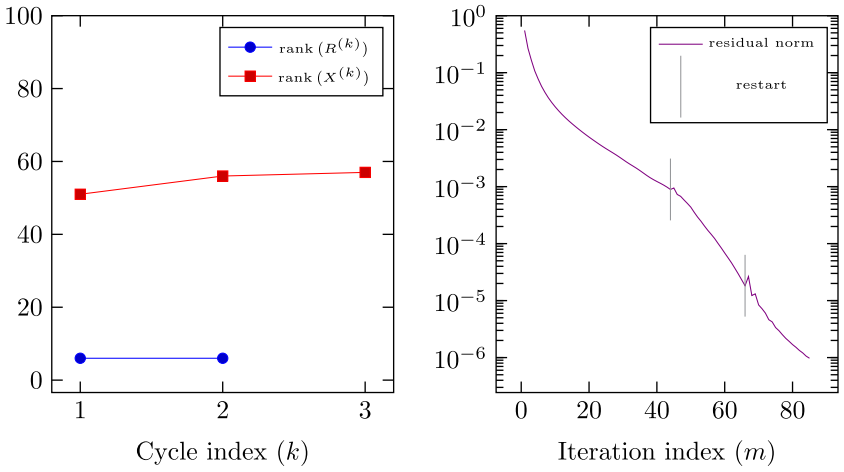

FIGURE 3 Convection–diffusion equation from Section 4.2 with $n = 25$. Residual and solution ranks (left) and residual norms (right) for RESTARTED_SYLV. Vertical tick marks indicate the start of a new cycle

on the unit cube $\Omega = (0,1)^3$, respectively. The viscosity parameter is $\epsilon = 0.01$ 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 (19) 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), $n = 25$ and $n = 80$. The resulting problems are of dimension 15,625 and 512,000, respectively. The low-rank matrices $\boldsymbol{C}, \boldsymbol{D} \in \mathbb{R}^{n^3 \times 3}$ 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, that is, we set $\texttt{mem}_{\max}$ in Algorithm 2 equal to 264. With this setting, RESTARTED_SYLV needs two 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 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 3 (right), the entire relative residual norm history is reported. We again see that the low-rank compression procedure (Algorithm 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 the matrices A and B have dissimilar spectral properties. This can be appreciated by looking at the results in Table 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, 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 512,000), leads to 26 iterations for 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

TABLE 3 Convection–diffusion equation from Section 4.2 with $n = 25$

	***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

Note: Performance measures for RESTARTED_SYLV with $\texttt{mem}_{\max} = 264$ compared with EKSM.

TABLE 4 Convection-diffusion equation from Section 4.2 with $n = 25$ and $n = 80$

	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

Note: Computational times for RESTARTED_SYLV compared with EKSM.

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. 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 with 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 setup 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.

ACKNOWLEDGEMENTS

Stefano Massei and Davide Palitta are members of the Italian INdAM Research group GNCS. Kathryn Lund is 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. Open access funding enabled and organized by Projekt DEAL.

CONFLICT OF INTEREST

This work does not have any conflicts of interest.

ORCID

Daniel Kressner https://orcid.org/0000-0003-3369-2958
Kathryn Lund https://orcid.org/0000-0001-9851-6061
Stefano Massei https://orcid.org/0000-0003-1813-4181
Davide Palitta https://orcid.org/0000-0002-6987-4430

REFERENCES

1. Baker J, Embree M, Sabino J. Fast singular value decay for Lyapunov solutions with nonnormal coefficients. SIAM J Matrix Anal Appl. 2015;36:656–668.
2. Grasedyck L. Existence of a low rank or $\mathscr{H}$-matrix approximant to the solution of a Sylvester equation. Numer Linear Algebra Appl. 2004;11:371–389.
3. Penzl T. Eigenvalue decay bounds for solutions of Lyapunov equations: The symmetric case. Syst Control Lett. 2000;40:139–144. https://doi.org/10.1016/S0167-6911(00)00010-4.
4. Antoulas AC. Approximation of large-scale dynamical systems. Philadelphia, PA: SIAM Publications, 2005.
5. Zhou K, Doyle JC, Glover K. Robust and optimal control. Upper Saddle River, NJ: Prentice-Hall, 1996.
6. Palitta D, Simoncini V. Matrix-equation-based strategies for convection-diffusion equations. BIT. 2016;56:751–776. https://doi.org/10.1007/s10543-015-0575-8.
7. Bini DA, Iannazzo B, Meini B. Numerical solution of algebraic Riccati equations, vol. 9 of Fundamentals of Algorithms. Philadelphia, PA: SIAM, 2012.
8. Binning A. Solving second and third-order approximations to DSGE models: A recursive Sylvester equation solution, Norges bank working paper. Vol 18, 2013. https://papers.ssrn.com/sol3/papers.cfm?abstract_id=2353010.
9. Benner P, Saak J. Numerical solution of large and sparse continuous time algebraic matrix Riccati and Lyapunov equations: A state of the art survey. GAMM-Mitt. 2013;36:32–52. https://doi.org/10.1002/gamm.201310003.
10. Simoncini V. Computational methods for linear matrix equations. SIAM Rev. 2016;58:377–441. https://doi.org/10.1137/130912839.
11. Davis TA, Rajamanickam S, Sid-Lakhdar WM. A survey of direct methods for sparse linear systems. Acta Numer. 2016;25:383–566. https://doi.org/10.1017/S0962492916000076.
12. S. A. Sauter and C. Schwab, Boundary element methods, vol. 39 of Springer series in computational mathematics, Springer-Verlag, Berlin, Germany: 2011, https://doi.org/10.1007/978-3-540-68093-2.
13. Saad Y. Numerical solution of large Lyapunov equations. Signal processing, scattering and operator theory, and numerical methods (Amsterdam, 1989), vol. 5 of programming systems control theory. Boston, MA: Birkhäuser, 1990; p. 503–511.
14. Jaimoukha IM, Kasenally EM. Krylov subspace methods for solving large Lyapunov equations. SIAM J Numer Anal. 1994;31:227–251. https://doi.org/10.1137/0731012.
15. Beckermann B. An error analysis for rational Galerkin projection applied to the Sylvester equation. SIAM J Numer Anal. 2011;49:2430–2450. https://doi.org/10.1137/110824590.
16. Simoncini V, Druskin V. Convergence analysis of projection methods for the numerical solution of large Lyapunov equations. SIAM J Numer Anal. 2009;47:828–843. https://doi.org/10.1137/070699378.
17. Simoncini V. Restarted full orthogonalization method for shifted linear systems. BIT. 2003;43:459–466. https://doi.org/10.1023/A:1026000105893.
18. Frommer A, Glässner U. Restarted GMRES for shifted linear systems. SIAM J Sci Comput. 1998;19:15–26. https://doi.org/10.1137/S1064827596304563.
19. Palitta D, Simoncini V. Optimality properties of Galerkin and Petrov-Galerkin methods for linear matrix equations. Vietnam J. Math. 2020. https://doi.org/10.1007/s10013-020-00390-7.
20. Ellner NS, Wachspress EL. New ADI model problem applications. Proceedings of the 1986 ACM Fall Joint Computer Conference, New York, NY, IEEE; 1986. p. 528–534.
21. Benner P, Li R-C, Truhar N. On the ADI method for Sylvester equations. J Comput Appl Math. 2009;233:1035–1045. https://doi.org/10.1016/j.cam.2009.08.108.
22. Kürschner P, Freitag M. Inexact methods for the low rank solution to large scale Lyapunov equations. BIT. 2020. https://doi.org/10.1007/s10543-020-00813-4.
23. Benner P, Kürschner P, Saak J. Self-generating and efficient shift parameters in ADI methods for large Lyapunov and Sylvester equations. Electron Trans Numer Anal. 2014;43:142–162.
24. Druskin V, Simoncini V. Adaptive rational Krylov subspaces for large-scale dynamical systems. Syst Control Lett. 2011;60:546–560. https://doi.org/10.1016/j.sysconle.2011.04.013.
25. Breiten T, Simoncini V, Stoll M. Low-rank solvers for fractional differential equations. Electr Trans Numer Anal. 2016;45:107–132.
26. Simoncini V. A new iterative method for solving large-scale Lyapunov matrix equations. SIAM J Sci Comput. 2007;29:1268–1288. https://doi.org/10.1137/06066120X.

27. Hodel AS, Tenison B, Poolla KR. Numerical solution of the Lyapunov equation by approximate power iteration. Linear Algebra Appl. 1996;236:205–230.
28. Sorensen DC, Antoulas AC. The Sylvester equation and approximate balanced reduction. Linear Algebra Appl. 2002;351(352):671–700. https://doi.org/10.1016/S0024-3795(02)00283-5.
29. Kressner D. Memory-efficient Krylov subspace techniques for solving large-scale Lyapunov equations. Proc IEEE Int Symp Comput Control Syst Des. 2008;613–618. https://doi.org/10.1109/CACSD.2008.4627370.
30. Gutknecht MH. Block Krylov space methods for linear systems with multiple right-hand sides: An introduction. In: Siddiqi AH, Duff IS, Christensen O, editors. Modern mathematical models: Methods and algorithms for real world systems. New Delhi: Anamaya, 2007; p. 420–447.
31. Bartels RH, Stewart GW. Algorithm 432: solution of the matrix equation AX + XB = C. Comm ACM. 1972;15:820–826.
32. Birk S. Deflated shifted block Krylov subspace methods for Hermitian positive definite matrices [PhD thesis]. Fakultät für Mathematik und Naturwissenschaften, Bergische Universität Wuppertal; 2015.
33. Soodhalter KM. Stagnation of block GMRES and its relationship to block FOM. Electr Trans Numer Anal. 2017;46:162–189.
34. Higham NJ. Accuracy and stability of numerical algorithms. 2nd ed. Philadelphia, PA: SIAM, 2002.
35. Benner P, Khoury GE, Sadkane M. On the squared Smith method for large-scale Stein equations. Numer Linear Algebra Appl. 2014;21:645–665. https://doi.org/10.1002/nla.1918.
36. Kressner D, Tobler C. Low-rank tensor Krylov subspace methods for parametrized linear systems. SIAM J Matrix Anal Appl. 2011;32:1288–1316. https://doi.org/10.1137/100799010.
37. Beckermann B, Townsend A. Bounds on the singular values of matrices with displacement structure. SIAM Rev. 2019;61:319–344. https://doi.org/10.1137/19M1244433.
38. Hammarling S. Numerical solution of the stable, nonnegative definite Lyapunov equation. IMA J Numer Anal. 1982;2:303–323.
39. Benner P, Ezzatti P, Kressner D, Quintana-Ortí ES, Remón A. A mixed-precision algorithm for the solution of Lyapunov equations on hybrid CPU-GPU platforms. Parall Comput. 2011;37:439–450. https://doi.org/10.1016/j.parco.2010.12.002.
40. Lang N, Mena H, Saak J. On the benefits of the LDL^T factorization for large-scale differential matrix equation solvers. Linear Algebra Appl. 2015;480:44–71. https://doi.org/10.1016/j.laa.2015.04.006.
41. Snyders J, Zakai M. On nonnegative solutions of the equation AD+DA'=-C. SIAM J Appl Math. 1970;18:704–714. doi:10.1137/0118063
42. Halmos PR. Positive approximants of operators. Ind Univ Math J. 1971/72;21:951–960. https://doi.org/10.1512/iumj.1972.21.21076.
43. Higham NJ. Computing a nearest symmetric positive semidefinite matrix. Linear Algebra Appl. 1988;103:103–118. https://doi.org/10.1016/0024-3795(88)90223-6.
44. Simoncini V, Szyld DB. Theory of inexact Krylov subspace methods and applications to scientific computing. SIAM J Sci Comput. 2003;25:454–477. https://doi.org/10.1137/S1064827502406415.
45. Horn RA, Kittaneh F. Two applications of a bound on the Hadamard product with a Cauchy matrix. Electr J Linear Algebra. 1998;3:4–12. https://doi.org/10.13001/1081-3810.1010.
46. Palitta D, Simoncini V. Computationally enhanced projection methods for symmetric Sylvester and Lyapunov matrix equations. J Comput Appl Math. 2018;330:648–659. https://doi.org/10.1016/j.cam.2017.08.011.
47. Dubrulle AA. Retooling the method of block conjugate gradients. Electr Trans Numer Anal. 2001;12:216–233.
48. Duff IS, Marrone M, Radicati G, Vittoli C. Level 3 basic linear algebra subprograms for sparse matrices: a user-level interface. ACM Trans Math Softw. 1997;23:379–401. https://doi.org/10.1145/275323.275327.
49. Ballard G, Carson EC, Demmel JW, Hoemmen M, Knight N, Schwartz O. Communication lower bounds and optimal algorithms for numerical linear algebra. Acta Numer. 2014;23:1–155. https://doi.org/10.1017/S0962492914000038.
50. Terpstra D, Jagode H, You H, Dongarra J. Collecting performance data with PAPI-C. Proceedings 3rd Parallel Tools Workshop Tools for High Performance Computing 2009, Springer, Berlin, Heidelberg, Germany; 2009. p. 157-173.
51. MATLAB, version 9.3.0.713579 (R2017b). Natick, MA: The MathWorks Inc, 2017.
52. Cullum JK, Greenbaum A. Relations between Galerkin and norm-minimizing iterative methods for solving linear systems. SIAM J Matrix Anal Appl. 1996;17:223–247. https://doi.org/10.1137/S0895479893246765.
53. Cullum JK. Peaks, plateaus, numerical instabilities in a Galerkin minimal residual pair of methods for solving Ax = b. Appl Numer Math. 1995;19:255–278. https://doi.org/10.1016/0168-9274(95)00086-0.
54. Hu DY, Reichel L. Krylov-subspace methods for the Sylvester equation. Linear Algebra Appl. 1992;172:283–313.
55. Lin Y, Simoncini V. Minimal residual methods for large scale Lyapunov equations. Appl Numer Math. 2013;72:52–71. https://doi.org/10.1016/j.apnum.2013.04.004.

How to cite this article: Kressner D, Lund K, Massei S, Palitta D. Compress-and-restart block Krylov subspace methods for Sylvester matrix equations. *Numer Linear Algebra Appl.* 2021;28:e2339. https://doi.org/10.1002/nla.2339