Electronic Transactions on Numerical Analysis.
Volume 62, pp. 138–162, 2024.

ISSN 1068–9613.
DOI: 10.1553/etna_vol62s138

ETNA
Kent State University and
Johann Radon Institute (RICAM)

A CLASS OF PETROV–GALERKIN KRYLOV METHODS FOR ALGEBRAIC RICCATI EQUATIONS*

CHRISTIAN BERTRAM† AND HEIKE FAßBENDER†

Abstract. A class of (block) rational Krylov-subspace-based projection methods for solving the large-scale continuous-time algebraic Riccati equation (CARE) $0 = \mathcal{R}(X) := A^H X + XA + C^H C - XBB^H X$ with a large, sparse A, and B and C of full low rank is proposed. The CARE is projected onto a block rational Krylov subspace $\mathcal{K}_j$ spanned by blocks of the form $(A^H - s_k I)^{-1} C^H$ for some shifts s_k, $k = 1, \ldots, j$. The considered projections do not need to be orthogonal and are built from the matrices appearing in the block rational Arnoldi decomposition associated to $\mathcal{K}_j$. The resulting projected Riccati equation is solved for the small square Hermitian Y_j. Then the Hermitian low-rank approximation $X_j = Z_j Y_j Z_j^H$ to X is set up where the columns of Z_j span $\mathcal{K}_j$. The residual norm $\|R(X_j)\|_F$ can be computed efficiently via the norm of a readily available $2p \times 2p$ matrix. We suggest reducing the rank of the approximate solution X_j even further by truncating small eigenvalues from X_j. This truncated approximate solution can be interpreted as the solution of the Riccati residual projected to a subspace of $\mathcal{K}_j$. This gives us a way to efficiently evaluate the norm of the resulting residual. Numerical examples are presented.

Key words. algebraic Riccati equation, large-scale matrix equation, (block) rational Krylov subspace, projection method

AMS subject classifications. 15A24, 65F15

1. Introduction.

The numerical solution of large-scale algebraic Riccati equations

$$0 = \mathcal{R}(X) := A^H X + XA + C^H C - XBB^H X \tag{1.1}$$

with a large, sparse matrix $A \in \mathbb{C}^{n\times n}$, and matrices $B \in \mathbb{C}^{n\times m}$ and $C \in \mathbb{C}^{p\times n}$, is of interest in a number of applications, as noted in [8, 22, 31] and references therein. As usual, we are interested in finding the stabilizing solution X; that is, the Hermitian positive semidefinite solution $X = X^H$ such that the spectrum of the closed loop matrix $(A - BB^H X)$ lies in the open left half-plane $\mathbb{C}_-$. This solution exists and is unique [12, 21] when (A, B) is stabilizable (that is, $\operatorname{rank}[A - zI, B] = n$ for each value of z in the closed right half-plane) and (A, C) is detectable (that is, (A^H, C^H) is stabilizable). These conditions are generically fulfilled and are assumed to hold in the remainder of this paper. For our discussion, it is further assumed that B and C have full column and row rank, respectively, with $m, p \ll n$. Then it is well known that the sought-after solution X will often have a low numerical rank (that is, its numerical rank q is $q \ll n$) [3]. Thus, it can be expressed in the form $X = ZYZ^H$ for some full-rank matrices $Z \in \mathbb{C}^{n\times q}$ and $Y = Y^T \in \mathbb{C}^{q\times q}$ with $q < n$. This allows for the construction of iterative methods that approximate X with a series of low-rank matrices stored in low-rank factored form. There are several methods, e.g., rational Krylov subspace methods, low-rank Newton–Kleinman methods, and Newton-ADI (alternating direction implicit) type methods, that produce such a low-rank approximation; see, e.g., [1, 4, 6, 18, 22, 24, 30, 31, 35, 36] and see also [5] for an overview.).

Here we are concerned with projection type methods which project the original Riccati equation (1.1) onto an appropriate subspace, impose a Galerkin condition on the projected problem, and expand its solution back to the whole space. This idea can be summarized as follows; see, e.g., [5, 30]. Assume that a sequence of nested subspaces $\mathcal{N}_k \subseteq \mathcal{N}_{k+1}$, $k \geq 1$, is generated with $\dim \mathcal{N}_k = kp$. Let $Q_k \in \mathbb{C}^{n\times kp}$ denote the matrix whose columns vectors

*Received January 10, 2024. Accepted September 18, 2024. Published online on October 11, 2024. Recommended by Valeria Simoncini.

†Institut für Numerische Mathematik, Technische Universität Braunschweig, Universitätsplatz 2, 38106 Braunschweig, Germany (h.fassbender@tu-braunschweig.de).

ETNA
Kent State University and
Johann Radon Institute (RICAM)

are an orthonormal basis of $\mathcal{N}_k$. Then the Galerkin condition reads

$$Q_k^H \mathcal{R}(X_k) Q_k = 0.$$

This gives a small-scale Riccati equation for $Y_k = Y_k^H \in \mathbb{C}^{kp \times kp}$, i.e.,

$$A_k^H Y_k + Y_k A_k - Y_k B_k B_k^H Y_k + C_k^H C_k = 0, \tag{1.2}$$

with $A_k = Q_k^H A Q_k \in \mathbb{C}^{kp \times kp}$, $B_k = Q_k^H B \in \mathbb{C}^{kp \times m}$, and $C_k^H = Q_k^H C^H \in \mathbb{C}^{kp \times p}$.

If (A_k, B_k) and (A_k^H, C_k^H) are stabilizable and detectable, respectively, then the existence of a stabilizing solution of (1.2) is ensured. In [30, Proposition 3.3], a general condition on A and B is given which ensures stabilizability of (A_k, B_k). Unfortunately, in practice, it is difficult to check whether this condition holds. But as the set of matrices $(\tilde{A}, \tilde{B}, \tilde{C})$ with $(\tilde{A}, \tilde{B})$ stabilizable and $(\tilde{A}, \tilde{C})$ detectable is dense in $\mathbb{C}^{\tilde{n} \times \tilde{n}} \times \mathbb{C}^{\tilde{n} \times m} \times \mathbb{C}^{p \times \tilde{n}}$, most likely (1.2) will have a unique stabilizing Hermitian positive semidefinite solution. Thus, in general, a unique Hermitian stabilizing solution Y_k of this small-scale Riccati equation exists, such that the approximate solution

$$X_k = Q_k Y_k Q_k^H$$

can be constructed. When $k = n$, then $\mathcal{R}(X_k) = 0$ must hold and the exact solution has been found. The effectiveness of this approach depends on the choice of $\mathcal{N}_k$. The approximation spaces explored in the literature are all based on block (rational) Krylov subspaces; see [15, 18, 22, 24, 30, 31]. For a short historic review of these ideas (stemming usually from algorithms to solve Lyapunov equations) and a list of references, see [31, Section 2] or [5, Section 2]. In [18] the matrix Q_k is constructed as a basis of the extended block Krylov subspace $\kappa_k(A^H, C^H) + \kappa_k(A^{-H}, A^{-H} C^H)$ [14, 20] for the standard block Krylov subspace

$$\kappa_k(A^H, C^H) = \operatorname{range}([C^H, A^H C^H, (A^H)^2 C^H, \ldots, (A^H)^{k-1} C^H]). \tag{1.3}$$

In [15, 31], in addition, Q_k is also generated from the block rational Krylov subspace [26]

$$\begin{aligned} \mathfrak{K}_k &\coloneqq \kappa_k(A^H, C^H, \mathcal{S}_{k-1}) \\ &= \operatorname{range}([C^H, (A^H - s_1 I)^{-1} C^H, \ldots, (A^H - s_{k-1} I)^{-1} C^H]) \end{aligned} \tag{1.4}$$

for a set of shifts $\mathcal{S}_{k-1} = \{s_1, \ldots, s_{k-1}\} \subset \mathbb{C}$ disjoint from the spectrum $\Lambda(A^H)$. The use of the rational Krylov subspace $\mathfrak{K}_k$ is explored further in [30]. The resulting projection algorithm is usually termed the RKSM (for "rational Krylov subspace method") algorithm. In [5], it was observed that projecting to rational Krylov subspaces often fared better than projecting to extended Krylov subspaces.

In both [22] and [4] methods are proposed which can be interpreted as projecting the large-scale Riccati equation (1.1) onto the block rational Krylov subspace $\mathcal{K}_k$,

$$\mathcal{K}_k \coloneqq \mathcal{K}_k(A^H, C^H, \mathcal{S}_k) = \operatorname{range}([(A^H - s_1 I)^{-1} C^H, \ldots, (A^H - s_k I)^{-1} C^H]), \tag{1.5}$$

for a set of shifts $\mathcal{S}_k = \{s_1, \ldots, s_k\} \subset \mathbb{C}$ disjoint from the spectrum $\Lambda(A^H)$. In [22] approximate solutions to the continuous-time algebraic Riccati equation (CARE) (1.1) are constructed by running subspace iterations on the Cayley transforms of the Hamiltonian matrix $\left[\begin{smallmatrix} A & BB^H \\ C^H C & -A^H \end{smallmatrix}\right] \in \mathbb{C}^{2n \times 2n}$ associated to (1.1). It is observed that the columns of the orthonormal matrix Q_k in the resulting approximate solution $X_k^{\text{cay}} = Q_k Y_k^{\text{cay}} Q_k^H$ span $\mathcal{K}_k$. Moreover, it

ETNA
Kent State University and
Johann Radon Institute (RICAM)

is proven in [22, Theorem 4.4] that the subspace iteration yields (under certain conditions) a matrix Q_k such that, besides (1.5), also $Q_k^H \mathcal{R}(X_k^{\text{cay}}) Q_k = 0$ holds. In the case when the reduced problem (1.2) admits a unique stabilizing solution, this gives equivalence between the subspace iteration and the Galerkin projection onto the rational Krylov subspace $\mathcal{K}_k$. The RADI (low-rank alternating direction implicit) algorithm suggested in [4] is a generalization of the Cholesky-factored variant of the Lyapunov ADI method. It generates approximations $X_k^{\text{radi}} = Z_k Y_k^{\text{radi}} Z_k^H$, where the columns of Z_k can be interpreted as a nonorthogonal basis of $\mathcal{K}_k$. As noted in [11, Section 4.2], X_j^{radi} can be interpreted as the solution of a projection of the large-scale Riccati equation (1.1) onto the Krylov subspace $\mathcal{K}_k$ employing an oblique projection. For both methods, the rank of the Riccati residual $\mathcal{R}(X_k)$ is always equal to p. The norm of $\mathcal{R}(X_k)$ can be evaluated efficiently, giving a cheap stopping criterion.

We take up the idea of employing a Petrov–Galerkin condition for the Riccati residual equation (1.1). In our discussion we concentrate solely on projecting onto a block rational Krylov subspace $\mathcal{K}_k$ (1.5), which so far has not been considered in the literature as an idea on its own. We make use of the (generalized) block rational Arnoldi decomposition (BRAD) $A^H V_{k+1} \underline{K}_k = V_{k+1} \underline{H}_k$ [16, Definition 2.2], where $\operatorname{range}(V_{k+1}) = \mathfrak{K}_{k+1}$. As $\operatorname{range}(V_{k+1}\underline{K}_k) = \mathcal{K}_k$, choosing $Z_k = V_{k+1}\underline{K}_k$ and a suitable matrix W_k yields a (not necessarily orthogonal) projection $\Pi_k = Z_k (W_k^H Z_k)^{-1} W_k^H$ onto $\mathcal{K}_k$. The projected CARE $\Pi_k \mathcal{R}(X_k) \Pi_k^H = 0$ gives a small-scale Riccati equation whose coefficient matrices are given essentially by the matrices of the BRAD. The resulting projected Riccati equation is solved for the small square Hermitian Y_k. Then the Hermitian low-rank approximation $X_k = Z_k Y_k Z_k^H$ to X is set up.

Details of this approach are explained in Section 3. As explained in Section 4, the residual norm $\|\mathcal{R}(X_k)\|_F$ can be computed efficiently via the norm of a readily available $2p \times 2p$ matrix, avoiding the explicit computation of the large-scale matrices X_k and $\mathcal{R}(X_k)$. A secondary result of that discussion is that $\operatorname{rank}(\mathcal{R}(X_k)) = 2p$, unlike for the RADI algorithm, where the residual is of rank p. In Section 5, we suggest reducing the rank of the approximate solution X_k even further by truncating small singular values from X_k. This truncated approximate solution can be interpreted as the solution of the Riccati residual projected to a subspace of $\mathcal{K}_k$. As a result, we obtain a way to efficiently evaluate the norm of the resulting residual. The transfer of the results to the generalized Riccati equation is explained in Section 6. Numerical examples are presented in Section 7. The results discussed here can be found in slightly different form in the PhD thesis [10].

2. Preliminaries. Rational Krylov spaces were initially proposed by Ruhe in the 1980s for the purpose of solving large sparse eigenvalue problems [26, 27, 28]; more recent work on block Krylov subspaces by Elsworth and Güttel can be found in [16]. We briefly recall some definitions and results on block rational Krylov subspaces and block rational Arnoldi decompositions from [16, Sections 1 and 2].

We are given a set $\mathcal{S}_k = \{s_1, \ldots, s_k\} \subset \mathbb{C} \backslash \{\infty\}$ disjoint from the spectrum $\Lambda(A^H)$. In the case in which A, B, and C in (1.1) are real matrices, any complex s_k should be accompanied by its complex-conjugate partner $\bar{s}_j$, so that the set $\mathcal{S}_k$ is closed under conjugation. In that case, all the algorithms suggested herein can be implemented in real arithmetic.

In the following, we will consider the block Krylov subspaces (1.4) and (1.5), that is,

$$\mathfrak{K}_{k+1} = \kappa_{k+1}(A^H, C^H, \mathcal{S}_k) = \operatorname{range}([C^H, (A^H - s_1 I)^{-1} C^H, \ldots, (A^H - s_k I)^{-1} C^H]),$$
$$\mathcal{K}_k = \mathcal{K}_k(A^H, C^H, \mathcal{S}_k) = \operatorname{range}([(A^H - s_1 I)^{-1} C^H, \ldots, (A^H - s_k I)^{-1} C^H]).$$

The columns of the starting block C^H lie in $\mathfrak{K}_{k+1}$, but not in $\mathcal{K}_k$. It is assumed that the $(k+1)p$ columns of $\mathfrak{K}_{k+1}$ are linearly independent.

ETNA
Kent State University and
Johann Radon Institute (RICAM)

There is a one-to-one correspondence between block rational Krylov spaces and so-called block rational Arnoldi decompositions. In [30] the decomposition

$$A^H Q_{j+1} = Q_{j+2} \underline{T}_{j+1}^H$$

is used, where the columns of $Q_{j+1} \in \mathbb{C}^{n\times(j+1)p}$ are orthonormal ($Q_{j+1}^H Q_{j+1} = I_{(j+1)p}$) and span either the extended block Krylov subspace or the block rational Krylov subspace $\mathfrak{K}_{j+1}$. We will consider the more general relation of the form

$$A^H V_{j+1} \underline{K}_j = V_{j+1} \underline{H}_j, \tag{2.1}$$

called a (generalized) orthonormal block rational Arnoldi decomposition (BRAD) as introduced in [16]. The following four conditions hold:

- the columns of $V_{j+1} \in \mathbb{C}^{n\times(j+1)p}$ are orthonormal, such that $\operatorname{range}(V_{j+1}) = \mathfrak{K}_{j+1}$;
- the matrices $\underline{H}_j$ and $\underline{K}_j$ are block upper Hessenberg matrices of size $(j+1)p \times jp$, where at least one of the matrices $H_{i+1,i}$ and $K_{i+1,i}$ is nonsingular,

$$\underline{H}_j = \begin{bmatrix} H_{11} & H_{12} & \cdots & H_{1,j-1} & H_{1j} \\ H_{21} & H_{22} & \cdots & H_{2,j-1} & H_{2j} \\ & \ddots & \ddots & \vdots & \vdots \\ & & H_{j-1,j-2} & H_{j-1,j-1} & H_{j-1,j} \\ & & & H_{j,j-1} & H_{jj} \\ & & & & H_{j+1,j} \end{bmatrix},$$

$$\underline{K}_j = \begin{bmatrix} K_{11} & K_{12} & \cdots & K_{1,j-1} & K_{1j} \\ K_{21} & K_{22} & \cdots & K_{2,j-1} & K_{2j} \\ & \ddots & \ddots & \vdots & \vdots \\ & & K_{j-1,p-2} & K_{j-1,j-1} & K_{j-1,j} \\ & & & K_{j,j-1} & K_{jj} \\ & & & & K_{j+1,j} \end{bmatrix}; \tag{2.2}$$

- $\beta_i K_{i+1,i} = \gamma_i H_{i+1,i}$, with scalars $\beta_i, \gamma_i \in \mathbb{C}$ such that $|\beta_i| + |\gamma_i| \neq 0$ for all $i = 1, \ldots, j$; and
- the quotients β_i/γ_i, $i = 1, \ldots, j$, called poles of the BRAD, correspond to the shifts $s_1, \ldots, s_j \in \mathcal{S}_j$.

An algorithm that constructs an orthonormal BRAD can be found in [16, Algorithm 2.1]; see also [9].

As a result of [16, Lemma 3.2(iii)], both matrices $\underline{K}_j$ and $\underline{H}_j$ are of full rank jp. This does not imply that all subdiagonal blocks $H_{i+1,i}$ and $K_{i+1,i}$ are nonsingular. As noted in [16, Lemma 3.2(ii)], if one of the subdiagonal blocks $H_{i+1,i}$ and $K_{i+1,i}$ is singular, it is the zero matrix. Thus, in the case $H_{i+1,i} = 0$ in (2.1), the matrix $K_{i+1,i}$ must be nonsingular. As $\beta_i K_{i+1,i} = \gamma_i H_{i+1,i}$ has to hold with $|\beta_i| + |\gamma_i| \neq 0$, this implies that $\beta_i = 0$. Hence, this can only happen for a zero shift in $\mathcal{S}_j$. In the case $K_{i+1,i} = 0$ in (2.1), the matrix $H_{i+1,i}$ must be nonsingular. As $\beta_i K_{i+1,i} = \gamma_i H_{i+1,i}$ has to hold with $|\beta_i| + |\gamma_i| \neq 0$, this implies that $\gamma_i = 0$ and a shift at infinity. Because of our assumption that the roots are finite, this case is excluded here. Thus, all blocks $K_{i+1,i}$ are nonsingular, while some blocks $H_{i+1,i}$ may be singular.

Any BRAD can be transformed into an equivalent one of the form

$$A^H \check{V}_{j+1} \underline{\check{K}}_j = \check{V}_{j+1} \underline{\check{H}}_j,$$

ETNA
Kent State University and
Johann Radon Institute (RICAM)

with $\underline{\check{K}}_j = \begin{bmatrix} 0_{p\times jp} \\ I_{jp} \end{bmatrix}$, $\operatorname{range}(\check{V}_{j+1}) = \mathfrak{K}_{j+1}$, $\check{V}_{j+1}\begin{bmatrix} I_p \\ 0_{jp\times p} \end{bmatrix} = C^H\Phi$, $\Phi \in \mathbb{C}^{p\times p}$, and an upper Hessenberg matrix $\underline{\check{H}}_j$, see [11, Section 2]. This implies that

$$\operatorname{range}(V_{j+1}\underline{K}_j) = \operatorname{range}(\check{V}_{j+1}\underline{\check{K}}_j) = \mathcal{K}_j, \tag{2.3}$$

$$\operatorname{range}(V_{j+1}\underline{H}_j) = \operatorname{range}(\check{V}_{j+1}\underline{\check{H}}_j) = A^H\mathcal{K}_j. \tag{2.4}$$

REMARK 2.1. Consider the BRAD $A^H V_{j+1}\underline{K}_j = V_{j+1}\underline{H}_j$. Let the thin QR decomposition $\underline{K}_j = QR$ be given with an orthonormal upper Hessenberg matrix $Q \in \mathbb{C}^{(j+1)p\times jp}$ and an upper triangular $R \in \mathbb{C}^{jp\times jp}$. As $\underline{K}_j$ is of full rank, R is nonsingular. With $\underline{\hat{K}}_j = Q$ and $\underline{\hat{H}}_j = \underline{H}_j R^{-1}$ we obtain the equivalent BRAD

$$A^H V_{j+1}\underline{\hat{K}}_j = V_{j+1}\underline{\hat{H}}_j,$$

with an orthonormal $\underline{\hat{K}}_j$. In the case in which V_{j+1} is orthonormal as well, the matrix $Z_j = V_{j+1}\underline{\hat{K}}_j$ is orthonormal.

3. General projection method. In this section we present in detail our approach to reduce the Riccati residual (1.1) to a low-dimensional one using a Petrov–Galerkin projection. In order to do so, we let the columns of $Z_j \in \mathbb{C}^{n\times jp}$ span some approximation space $\mathcal{K}_j$. We choose a matrix $W_j \in \mathbb{C}^{n\times jp}$ such that its columns span the test space $\mathcal{L}_j$ and $W_j^H Z_j \in \mathbb{C}^{jp\times jp}$ is nonsingular. Thus,

$$\Pi_j := Z_j(W_j^H Z_j)^{-1}W_j^H \in \mathbb{C}^{n\times n}$$

is a (not necessarily orthogonal) projection onto $\mathcal{K}_j$ along $\mathcal{L}_j^{\perp}$. Assume that the solution X of (1.1) can be approximated by

$$X_j = Z_j Y_j Z_j^H \tag{3.1}$$

for some Hermitian matrix $Y_j \in \mathbb{C}^{jp\times jp}$. Then, imposing a Petrov–Galerkin condition on $\mathcal{R}(X_j)$,

$$\begin{aligned} 0 &= \Pi_j\mathcal{R}(X_j)\Pi_j^H \\ &= Z_j\{((W_j^H Z_j)^{-1}W_j^H A^H Z_j)Y_j + Y_j(Z_j^H A W_j(W_j^H Z_j)^{-H}) \\ &\quad + ((W_j^H Z_j)^{-1}W_j^H C^H)(CW_j(W_j^H Z_j)^{-H}) + Y_j(Z_j^H B)(B^H Z_j)Y_j\}Z_j^H, \end{aligned}$$

one obtains a small-scale Riccati equation for Y_j,

$$0 = A_j^H Y_j + Y_j A_j + C_j^H C_j - Y_j B_j B_j^H Y_j, \tag{3.2}$$

with $A_j = Z_j^H A W_j(W_j^H Z_j)^{-H} \in \mathbb{C}^{jp\times jp}$, $C_j = CW_j(W_j^H Z_j)^{-H} \in \mathbb{C}^{p\times jp}$, and $B_j = Z_j^H B \in \mathbb{C}^{jp\times m}$.

In particular, we suggest projecting onto the block rational Krylov subspace $\mathcal{K}_j$ as in (1.4); that is, $\mathcal{K}_j = \mathcal{K}_j(A^H, C^H, \mathcal{S}_j)$. A suitable basis for this space is constructed via the generalized orthonormal RAD

$$A^H V_{j+1}\underline{K}_j = V_{j+1}\underline{H}_j, \tag{3.3}$$

where the columns of $V_{j+1}\underline{K}_j$ span the space $\mathcal{K}_j = \mathcal{K}_j(A^H, C^H, \mathcal{S}_j)$; see (2.3). Thus, we choose

$$Z_j := V_{j+1}\underline{K}_j \in \mathbb{C}^{n\times jp}. \tag{3.4}$$

ETNA
Kent State University and
Johann Radon Institute (RICAM)

Note that the columns of Z_j are in general not orthonormal. As $\underline{K}_j$ is of full rank, it is possible to choose a matrix $\underline{L}_j \in \mathbb{C}^{(j+1)p\times jp}$ such that $\underline{L}_j^H \underline{K}_j \in \mathbb{C}^{jp\times jp}$ is nonsingular. Set $W_j \coloneqq V_{j+1}\underline{L}_j \in \mathbb{C}^{n\times jp}$. Denote (as before) the space spanned by the columns of the matrix W_j by $\mathcal{L}_j$. Hence, $\Pi_j Z_j (W_j^H Z_j)^{-1} W_j^H \in \mathbb{C}^{n\times n}$ is a projection onto $\mathcal{K}_j$ along $\mathcal{L}_j^\perp$. Noting that $W_j^H Z_j = \underline{L}_j^H V_{j+1}^H V_{j+1}\underline{K}_j = \underline{L}_j^H \underline{K}_j$ holds and making use of (3.3), the coefficient matrices in (3.2) can be expressed as

$$\begin{aligned} A_j &\coloneqq \underline{H}_j^H \underline{L}_j (\underline{K}_j^H \underline{L}_j)^{-1} \in \mathbb{C}^{jp\times jp}, \\ B_j &\coloneqq \underline{K}_j^H V_{j+1}^H B \in \mathbb{C}^{jp\times m}, \\ C_j &\coloneqq C V_{j+1}\underline{L}_j (\underline{K}_j^H \underline{L}_j)^{-1} \in \mathbb{C}^{p\times jp}. \end{aligned} \tag{3.5}$$

The solution Y_j of (3.2) fully determines the approximate solution $X_j = V_{j+1}\underline{K}_j Y_j \underline{K}_j^H V_{j+1}$ (3.1) of (1.1).

The resulting algorithm is summarized in Algorithm 1. In the first step V_1 is chosen from the thin QR decomposition of $C^H = V_1 R$. This yields $CV_{j+1} = [R^H \;\; 0_{p\times jp}]$. Once the iteration is stopped, the approximate solution $X_j = V_{j+1}\underline{K}_j Y_j \underline{K}_j^H V_{j+1}^H$ can be set up (theoretically). In a large-scale setting, one may consider only its low-rank factor $V_{j+1}\underline{K}_j G_j$ with a Cholesky(-like) decomposition $Y_j = G_j G_j^H$.

Algorithm 1 General projection method for solving (1.1).

Require: System matrices A, B, C and set of shifts $\mathcal{S}_j = \{s_1, \ldots, s_j\} \subset \mathbb{C}\backslash\{\infty\}$ with $\mathcal{S}_j \cap \Lambda(A^H) = \emptyset$.

Ensure: Approximate solution $X_j = V_{j+1}\underline{K}_j Y_j \underline{K}_j^H V_{j+1}^H$.

1: Compute thin QR decomposition $C^H = V_1 R$ with $R \in \mathbb{C}^{p\times p}$.
2: Set $j = 1$.
3: **while** not converged **do**
4: Obtain next shift μ from $\mathcal{S}_j$.
5: Expand orthonormal BRAD to obtain $V_{j+1} = [V_j \;\; \hat{V}_{j+1}]$ orthonormal, $\underline{K}_j$, and $\underline{H}_j$.
6: Choose $\underline{L}_j$.
7: Compute $A_j = \underline{H}_j^H \underline{L}_j (\underline{K}_j^H \underline{L}_j)^{-1}$.
8: Compute $B_j = \underline{K}_j^H V_{j+1}^H B$.
9: Compute $C_j = [R^H \;\; 0_{p\times jp}]\underline{L}_j (\underline{K}_j^H \underline{L}_j)^{-1}$.
10: Solve $A_j^H Y_j + Y_j A_j + C_j^H C_j - Y_j B_j B_j^H Y_j = 0$.
11: Compute $\|\mathcal{R}(X_j)\|$ (see Algorithm 2).
12: Set $j = j + 1$.
13: **end while**

From the rank conditions on the matrices involved in (3.5) it follows that A_j is nonsingular, and the matrices C_j and B_j are of full rank, as $\operatorname{rank} C_j = p$ and $\operatorname{rank} B_j = \min\{pj, m\}$. Whether (3.2) has a unique stabilizing solution cannot be read off in general. But as the set of matrices (A_j, B_j, C_j) with (A_j, B_j) stabilizable and (A_j, C_j) detectable is dense in $\mathbb{C}^{jp\times jp} \times \mathbb{C}^{jp\times m} \times \mathbb{C}^{p\times jp}$, most likely (3.2) will have a unique stabilizing Hermitian positive semidefinite solution. Only in very rare cases of our numerical experiments could the small-scale equation (3.2) not be solved for a stabilizing solution.

The effectiveness of the proposed projection method depends on the choice of the set $\mathcal{S}_j$ of shifts and of the space $\mathcal{L}_j$ (the matrix $\underline{L}_j$). Here we briefly mention three different choices of $\underline{L}_j$:

ETNA
Kent State University and
Johann Radon Institute (RICAM)

- A natural choice is $\underline{L}_j := \underline{K}_j$, which yields a Galerkin projection as $\mathcal{L}_j = \mathcal{K}_j$.
- Another choice is $\underline{L}_j := \underline{H}_j$, which yields a Petrov–Galerkin projection as $\mathcal{L}_j = A^H \mathcal{K}_j$ due to (2.4).
- A more general choice is $\underline{L}_j := \alpha \underline{H}_j - \beta \underline{K}_j$, with $\alpha, \beta \in \mathbb{C}$ and $|\alpha| + |\beta| \neq 0$, which yields a Petrov–Galerkin projection in the case $\alpha \neq 0$.

REMARK 3.1. Assume that an orthonormal BRAD with orthonormal $\underline{K}_j$ (see Remark 2.1) is used. Then the matrix $Q_j := V_{j+1}\underline{K}_j$ has orthonormal columns. Its columns are an orthonormal basis of the Krylov subspace $\mathcal{K}_j$. Set $\underline{L}_j = \underline{K}_j$, so the projection is orthogonal. Then it follows from (3.5) that

$$\begin{aligned} A_j &= \underline{H}_j^H \underline{K}_j (\underline{K}_j^H \underline{K}_j)^{-1} = \underline{H}_j^H V_{j+1}^H V_{j+1} \underline{K}_j = \underline{K}_j^H V_{j+1}^H A\, V_{j+1} \underline{K}_j = Q_j^H A\, Q_j, \\ B_j &= \underline{K}_j^H V_{j+1}^H B = Q_j^H B, \\ C_j &= C Q_j \end{aligned}$$

due to $\underline{K}_j^H \underline{K}_j = I_j$, $V_{j+1}^H V_{j+1} = I_{j+1}$, and the BRAD (3.3). These reduced matrices look like those in (1.2). Recall that here we project onto the Krylov subspace $\mathcal{K}_j$, while in most of the literature [18, 22, 24, 30, 31] a projection on the Krylov subspace $\mathfrak{K}_j$ is used. In numerical experiments, no advantage concerning accuracy when using $Q_j^H A\, Q_j$, $Q_j^H B$ and CQ_j has been observed.

REMARK 3.2. Assume that V_{j+1} is some matrix with orthonormal columns. If $Z_j = V_{j+1}\underline{K}_j$ is chosen with some arbitrary $\underline{K}_j$ not satisfying (3.3), then A_j in (3.5) is given by $A_j = \underline{K}_j^H V_{j+1}^H A\, V_{j+1} \underline{L}_j (\underline{K}_j^H \underline{L}_j)^{-1}$. The derivations in Sections 3.1, 3.2, 4, and 5 do not hold in this case, as they depend on (3.3).

3.1. Efficient implementation. Algorithm 1 should not be implemented as formulated above. The block upper Hessenberg form of the matrices $\underline{K}_j$ and $\underline{H}_j$ should be utilized so that the matrices A_j, B_j, and C_j are not explicitly set up in each step, but rather obtained by a (simple) update from A_{j-1}, B_{j-1}, and C_{j-1}, respectively. Moreover, in steps 7 and 9 of Algorithm 1, a linear system of equations with jp (respectively p) right-hand sides has to be solved for the same $jp \times jp$ system matrix. This would require roughly $(jp)^3$ flops if the structure of the system matrix cannot be utilized.

Let us first consider the computation of B_j. As $\underline{K}_j$ is a block upper Hessenberg matrix which grows by one block row/column in each iteration step, we have from (2.2) that

$$\underline{K}_j = \begin{bmatrix} \underline{K}_{j-1} & k_j \\ 0_{p\times(j-1)p} & K_{j+1,j} \end{bmatrix} \quad \text{with} \quad k_j = \begin{bmatrix} K_{1j}^H & K_{2j}^H & \cdots & K_{jj}^H \end{bmatrix}^H \in \mathbb{C}^{jp\times p}. \tag{3.6}$$

Thus, as $V_{j+1} = \begin{bmatrix} V_j & \hat{V}_{j+1} \end{bmatrix}$, it holds that

$$B_j = \underline{K}_j^H V_{j+1}^H B = \begin{bmatrix} \underline{K}_{j-1}^H & 0_{(j-1)p\times p} \\ k_j^H & K_{j+1,j}^H \end{bmatrix} \begin{bmatrix} V_j^H \\ \hat{V}_{j+1}^H \end{bmatrix} B = \begin{bmatrix} B_{j-1} \\ k_j^H V_j^H B + K_{j+1,j}^H \hat{V}_{j+1}^H B \end{bmatrix}.$$

Thus, in each iteration step, B_j can be updated cheaply from $B_{j-1} = \underline{K}_{j-1}^H V_j^H B$, as only the last block row of B_j needs to be determined, since the rest of B_j is already known. Moreover, the recomputation of $V_j^H B$ for the last block row can be avoided by storing $\tilde{B}_j = V_{j+1}^H B = [\tilde{B}_{j-1} \;\; \hat{V}_{j+1}^H B]$ in each iteration step. Then $B_j = [k_j^H \;\; K_{j+1,j}^H]\tilde{B}_j$.

Next, we consider the computation of the matrix products $\underline{H}_j^H \underline{L}_j$ and $\underline{K}_j^H \underline{L}_j$. If in each iteration step a new $\underline{L}_j$ is selected without taking the previous $\underline{L}_{j-1}$ into account, these products have to be computed from scratch in each iteration step. This is different if $\underline{L}_j$ is

ETNA
Kent State University and
Johann Radon Institute (RICAM)

chosen as a block upper Hessenberg matrix whose leading part is given by $\underline{L}_{j-1}$ like $\underline{K}_j$ or $\underline{H}_j$ in (2.2) (see also (3.6)),

$$\underline{L}_j = \begin{bmatrix} \underline{L}_{j-1} & \ell_j \\ 0_{p\times(j-1)p} & L_{j+1,j} \end{bmatrix}. \tag{3.7}$$

Then only the last block row and column of $\underline{H}_j^H \underline{L}_j$ and $\underline{K}_j^H \underline{L}_j$ need to be computed when $\underline{H}_{j-1}^H \underline{L}_{j-1}$ and $\underline{K}_{j-1}^H \underline{L}_{j-1}$ are already known, e.g.,

$$\begin{aligned} \underline{K}_j^H \underline{L}_j &= \begin{bmatrix} \underline{K}_{j-1}^H & 0_{(j-1)p\times p} \\ k_j^H & K_{j+1,j}^H \end{bmatrix} \begin{bmatrix} \underline{L}_{j-1} & \ell_j \\ 0_{p\times(j-1)p} & L_{j+1,j} \end{bmatrix} \\ &= \begin{bmatrix} \underline{K}_{j-1}^H \underline{L}_{j-1} & \underline{K}_{j-1}^H \ell_j \\ k_j^H \underline{L}_{j-1} & k_j^H \ell_j + K_{j+1,j}^H L_{j+1,j} \end{bmatrix}. \end{aligned}$$

We need to be able to solve linear systems with the coefficient matrix $\underline{K}_j^H \underline{L}_j$ efficiently in order to set up A_j and C_j. One idea is to make use of the block LDU decomposition

$$\underline{K}_j^H \underline{L}_j = \begin{bmatrix} I_{(j-1)p} & 0_{(j-1)p\times p} \\ \Omega_j & I_p \end{bmatrix} \begin{bmatrix} \Psi_j & 0_{(j-1)p\times p} \\ 0_{p\times(j-1)p} & \Gamma_j \end{bmatrix} \begin{bmatrix} I_{(j-1)p} & \Delta_j \\ 0_{p\times(j-1)p} & I_p \end{bmatrix}, \tag{3.8}$$

with

$$\begin{aligned} \Omega_j &= k_j^H \underline{L}_{j-1} (K_{j-1}^H \underline{L}_{j-1})^{-1} \in \mathbb{C}^{p\times(j-1)p}, \\ \Delta_j &= (\underline{K}_{j-1}^H \underline{L}_{j-1})^{-1} \underline{K}_{j-1}^H \ell_j \in \mathbb{C}^{(j-1)p\times p}, \\ \Psi_j &= \underline{K}_{j-1}^H \underline{L}_{j-1} \in \mathbb{C}^{(j-1)p\times(j-1)p}, \\ \Gamma_j &= k_j^H \ell_j + K_{j+1,j}^H L_{j+1,j} - k_j^H \underline{L}_{j-1} (\underline{K}_{j-1}^H \underline{L}_{j-1})^{-1} \underline{K}_{j-1}^H \ell_j \\ &= k_j^H \ell_j + K_{j+1,j}^H L_{j+1,j} - k_j^H \underline{L}_{j-1} \Delta_j \in \mathbb{C}^{p\times p}. \end{aligned}$$

Setting up this decomposition, two linear systems with the system matrix $\Psi_j = \underline{K}_{j-1}^H \underline{L}_{j-1}$ need to be solved in order to determine Ω_j and Δ_j, both with p right-hand sides. For $\underline{K}_{j-1}^H \underline{L}_{j-1}$, there is a block LDU decomposition just like (3.8) which may be used to solve those systems recursively. In particular, for

$$\Delta_j = \begin{bmatrix} \Delta_{j,1} \\ \Delta_{j,2} \end{bmatrix}, \quad \Omega_j = [\Omega_{j,1} \ \ \Omega_{j,2}], \quad \underline{K}_{j-1}^H \ell_j = \begin{bmatrix} \mathfrak{h}_1 \\ \mathfrak{h}_2 \end{bmatrix}, \quad \text{and} \quad k_j^H \underline{L}_{j-1} = [\mathfrak{g}_1 \ \ \mathfrak{g}_2],$$

with $\Delta_{j,1}, \mathfrak{h}_1, \mathfrak{g}_1^T \in \mathbb{C}^{(j-2)p\times p}$ and $\Delta_{j,2}, \mathfrak{h}_2, \mathfrak{g}_2^T \in \mathbb{C}^{p\times p}$, we obtain

$$\begin{aligned} \Delta_{j,2} &= \Gamma_{j-1}^{-1} (\mathfrak{h}_2 - \Omega_{j-1} \mathfrak{h}_1), \\ \Delta_{j,1} &= \Psi_{j-1}^{-1} \mathfrak{h}_1 - \Delta_{j-1} \Delta_{j,2}, \\ \Omega_{j,2} &= (\mathfrak{g}_2 - \mathfrak{g}_1 \Delta_{j-1}) \Gamma_{j-1}^{-1} \\ \Omega_{j,1} &= \mathfrak{g}_1 \Psi_{j-1}^{-1} - \Omega_{j,2} \Omega_{j-1}. \end{aligned}$$

Thus, continuing in this fashion, in order to determine Δ_j and Ω_j, some $2j$ linear systems with the $p \times p$ coefficient matrices Γ_ℓ, $\ell = 1, \ldots, j$, need to be solved for p right-hand sides each. The computational cost for each of these steps is dominated by computing the LU decomposition of Γ_ℓ, which costs $\mathcal{O}(p^3)$ flops. Hence, setting up the LDU decomposition of $\underline{K}_j^H \underline{L}_j$ amounts to $\mathcal{O}(jp^3)$ flops.

ETNA
Kent State University and
Johann Radon Institute (RICAM)

Now we will concentrate on computing $A_j = \underline{H}_j^H \underline{L}_j (\underline{K}_j^H \underline{L}_j)^{-1}$ and assume that we have already determined

$$\tilde{A}_j = \underline{H}_j^H \underline{L}_j = \begin{bmatrix} \underline{H}_{j-1}^H \underline{L}_{j-1} & \underline{H}_{j-1}^H \ell_j \\ h_j^H \underline{L}_{j-1} & h_j^H \ell_j + H_{j+1,j}^H L_{j+1,j} \end{bmatrix} = \begin{bmatrix} (\tilde{A}_j)_{11} & (\tilde{A}_j)_{12} \\ (\tilde{A}_j)_{21} & (\tilde{A}_j)_{22} \end{bmatrix}.$$

The matrix

$$A_j = \underline{H}_j^H \underline{L}_j (\underline{K}_j^H \underline{L}_j)^{-1} = \tilde{A}_j (\underline{K}_j^H \underline{L}_j)^{-1} = \begin{bmatrix} (A_j)_{11} & (A_j)_{12} \\ (A_j)_{21} & (A_j)_{22} \end{bmatrix}$$

is given by

$$\begin{aligned} (A_j)_{12} &= ((\tilde{A}_j)_{12} - (\tilde{A}_j)_{11}\Delta_j)\Gamma_j^{-1}, \quad (A_j)_{11} = (\tilde{A}_j)_{11}\Psi_j^{-1} - (A_j)_{12}\Omega_j, \\ (A_j)_{22} &= ((\tilde{A}_j)_{22} - (\tilde{A}_j)_{21}\Delta_j)\Gamma_j^{-1}, \quad (A_j)_{21} = (\tilde{A}_j)_{21}\Psi_j^{-1} - (A_j)_{22}\Omega_j, \end{aligned}$$

due to (3.8), where $(\tilde{A}_j)_{11}\Psi_j^{-1} = \underline{H}_{j-1}^H \underline{L}_{j-1} (\underline{K}_{j-1}^H \underline{L}_{j-1})^{-1}$ is already known from the previous iteration. Thus, in order to determine A_j, two linear systems with the $p \times p$ coefficient matrix Γ_j need to be solved for $(j-1)p$ right-hand sides each. Computing the LU decomposition of Γ_j costs $\mathcal{O}(p^3)$ flops; the $(2j-2)p$ forward and backward solves amount to $\mathcal{O}(jp^3)$ flops. Computing $(\tilde{A}_j)_{21}\Psi_j^{-1}$ can be done in the same way as the calculation of Ω_j discussed above. The LU decompositions of the Γ_ℓ, $\ell = 1, \ldots, j$, should be reused here such that only additional forward and backward solves are needed here.

Finally, we consider the computation of $C_j = CV_{j+1}\underline{L}_j (\underline{K}_j^H \underline{L}_j)^{-1}$. With $R^H = CV_1$, we obtain $CV_{j+1} = [R^H \quad 0_{p \times jp}]$. Assuming that $\underline{L}_j$ is chosen as a block upper Hessenberg matrix as in (3.7), it follows that

$$\begin{aligned} \tilde{C}_j = CV_{j+1}\underline{L}_j &= \begin{bmatrix} R^H & 0_{p \times jp} \end{bmatrix} \begin{bmatrix} L_{11} & L_{12} & \cdots & L_{1,j-1} & L_{1j} \\ L_{21} & L_{22} & \cdots & L_{2,j-1} & L_{2j} \\ & \ddots & \ddots & \vdots & \vdots \\ & & L_{j-1,j-2} & L_{j-1,j-1} & L_{j-1,j} \\ & & & L_{j,j-1} & L_{jj} \\ & & & & L_{j+1,j} \end{bmatrix} \\ &= \begin{bmatrix} R^H L_{11} & R^H L_{12} & \cdots & R^H L_{1,j-1} & R^H L_{1j} \end{bmatrix} \\ &= \begin{bmatrix} CV_j \underline{L}_{j-1} & R^H L_{1j} \end{bmatrix} = \begin{bmatrix} \tilde{C}_{j-1} & (\tilde{C}_j)_{12} \end{bmatrix}. \end{aligned}$$

Thus, $CV_{j+1}\underline{L}_j$ can be updated cheaply from $CV_j\underline{L}_{j-1}$ as only the block $R^H L_{1j}$ needs to be determined; the rest of $CV_{j+1}\underline{L}_j$ is already known. We obtain $C_j = \tilde{C}_j (\underline{K}_j^H \underline{L}_j)^{-1} = [(C_j)_{11} \;\; (C_j)_{12}]$ with $(C_j)_{12} \in \mathbb{C}^{p \times p}$ and (3.8) via

$$(C_j)_{12} = ((\tilde{C}_j)_{12} - \tilde{C}_{j-1}\Delta_j)\Gamma_j^{-1}, \quad (C_j)_{11} = \tilde{C}_{j-1} (\underline{K}_{j-1}^H \underline{L}_{j-1})^{-1} - (C_j)_{12}\Omega_j.$$

The matrix $\tilde{C}_{j-1}(\underline{K}_{j-1}^H \underline{L}_{j-1})^{-1}$ is already known from the previous step. Thus, in order to determine C_j from C_{j-1}, there is just one linear system with the $p \times p$ system matrix Γ_j and p right-hand sides to be solved. Thus, the cost for setting up C_j is dominated by the cost for the LU decomposition of Γ_j, that is, $\mathcal{O}(p^3)$ flops.

In summary, each iteration step can be implemented such that just $\mathcal{O}(jp^3)$ flops are needed in the case where $\underline{L}_j$ is chosen as a block upper Hessenberg matrix whose leading part is given by $\underline{L}_{j-1}$ like $\underline{K}_j$ or $\underline{H}_j$ in (2.2); see also (3.6). This approach implies that all matrices $\Omega_\ell, \Delta_\ell^T \in \mathbb{C}^{p \times (\ell-1)p}$ and $\Gamma_\ell \in \mathbb{C}^{p \times p}$, $\ell = 1, \ldots, j$, have to be stored. Otherwise, that is, without taking the previous $\underline{L}_{j-1}$ into account, each iteration step will cost $\mathcal{O}((jp)^3)$ flops.

ETNA
Kent State University and
Johann Radon Institute (RICAM)

3.2. Shift selection. For fast convergence, the choice of the poles of the block rational Krylov subspace used in the approximate solution X_j is crucial. It would be desirable to determine a set of shifts $\mathcal{S}_j$ such that the approximation X_j^* satisfies $\|X - X_j^*\| = \min_{\mathcal{S}_j} \|X - X_j\|$ in some norm, where X_j has been computed by Algorithm 1 using $\mathcal{S}_j$. This question has been considered for symmetric Sylvester equations [2]. Related work on Lyapunov equations can be found in [33, 34]. Up to now, it is an open question on how to compute optimal shifts in the sense described above for Riccati equations.

Many shift strategies based on other ideas exist, but their description is beyond the scope of this work. In [15, Section 5] and in [30] the shift selection for methods projecting (1.1) onto the rational Krylov subspace $\mathfrak{K}_j$ (1.4) is discussed. The subspace iteration method proposed in [22] can (under certain assumptions) be interpreted as projecting (1.1) onto the rational Krylov subspace $\mathcal{K}_j$ (1.5). Hence, the comments on the shift selection given in [22] hold for the approach proposed in this paper. Moreover, the subspace iteration method is equivalent to the RADI algorithm [4] (when using $X_0 = 0$ and the same set of shifts). A detailed discussion of the shift selection in the context of the RADI algorithm and related, equivalent methods is given in [4, Section 4.5]. In particular, the idea of choosing s_{j+1} in order to minimize $\|\mathcal{R}(X_{j+1})\|$ once X_j is fixed is pursued [4, Section 4.5.2]. A recent numerical comparison of different solvers of (1.1) including the choice of shifts can be found in [5].

4. Efficient residual norm evaluation. Typically, the residual norm $\|\mathcal{R}(X_j)\|$ is used as an indicator for whether X_j is a good approximation to the desired solution X of (1.1). In the previous section, we replaced solving the large-scale $n \times n$ problem (1.1) by solving a much smaller one, (3.2). In order to be able to use $\|\mathcal{R}(X_j)\|$ as a convergence indicator, an equivalent small-scale expression has to be found, as $\mathcal{R}(X_j) \in \mathbb{C}^{n\times n}$ cannot be computed explicitly due to storage constraints. In the following, we will derive such an expression for $\|\mathcal{R}(X_j)\|$ that allows its efficient evaluation even if n is large.

As a first step, we need to derive an alternative expression for the Riccati residual $\mathcal{R}(X_j) = A^H X_j + X_j A + C^H C - X_j B B^H X_j$. Recall that $X_j = Z_j Y_j Z_j^H$ holds with $Z_j = V_{j+1}\underline{K}_j$ as in (3.4). Thus, making use of (3.3) we obtain

$$X_j A = V_{j+1}\underline{K}_j Y_j \underline{K}_j^H V_{j+1}^H A = V_{j+1}\underline{K}_j Y_j \underline{H}_j^H V_{j+1}^H.$$

Moreover, we have $X_j B B H_j^H = V_{j+1}\underline{K}_j Y_j S_j Y_j \underline{K}_j^H V_{j+1}^H$ with

$$S_j = \underline{K}_j^H V_{j+1}^H B B^H V_{j+1}\underline{K}_j = Z_j^H B B^H Z_j. \tag{4.1}$$

With this, the Riccati residual can be expressed as

$$\begin{aligned}\mathcal{R}(X_j) &= V_{j+1}\underline{H}_j Y_j \underline{K}_j^H V_{j+1}^H + V_{j+1}\underline{K}_j Y_j \underline{H}_j^H V_{j+1}^H + C^H C - V_{j+1}\underline{K}_j Y_j S_j Y_j \underline{K}_j^H V_{j+1}^H \\ &= V_{j+1}(\underline{H}_j Y_j \underline{K}_j^H + \underline{K}_j Y_j \underline{H}_j^H + \tilde{C}\tilde{C}^H - \underline{K}_j Y_j S_j Y_j \underline{K}_j^H) V_{j+1}^H, \end{aligned} \tag{4.2}$$

with $C^H = V_{j+1}\tilde{C}$.

Next, rewrite Π_j as

$$\Pi_j = Z_j (W_j^H Z_j)^{-1} W_j^H = V_{j+1}\underline{K}_j (\underline{L}_j^H \underline{K}_j)^{-1} \underline{L}_j^H V_{j+1}^H = V_{j+1}\pi_j V_{j+1}^H$$

with

$$\pi_j := \underline{K}_j (\underline{L}_j^H \underline{K}_j)^{-1} \underline{L}_j^H \in \mathbb{C}^{(j+1)p\times(j+1)p}. \tag{4.3}$$

Note that π_j is a projection onto the space spanned by the columns of $\underline{K}_j$ along the orthogonal complement of the space spanned by the columns of $\underline{L}_j$. This implies that $\underline{K}_j^H \pi_j^H = \underline{K}_k^H \underline{L}_j (\underline{K}_k^H \underline{L}_j)^{-1} \underline{K}_j^H = \underline{K}_j^H$.

ETNA
Kent State University and
Johann Radon Institute (RICAM)

With this, the projected Riccati residual can be expressed as

$$\begin{aligned} 0 &= \Pi_j \mathcal{R}(X_j) \Pi_j^H \\ &= V_{j+1} \pi_j (\underline{H}_j Y_j \underline{K}_j^H + \underline{K}_j Y_j \underline{H}_j^H + \tilde{C}\tilde{C}^H - \underline{K}_j Y_j S_j Y_j \underline{K}_j^H) \pi_j^H V_{j+1}^H. \end{aligned} \tag{4.4}$$

As $\Pi_j \mathcal{R}(X_j) \Pi_j^H = 0$, (4.2) and (4.4) yield the desired alternative expression for the Riccati residual:

$$\begin{aligned} \mathcal{R}(X_j) &= \mathcal{R}(X_j) - \Pi \mathcal{R}(X_j) \Pi^H \\ &= V_{j+1} (\underline{H}_j Y_j \underline{K}_j^H + \underline{K}_j Y_j \underline{H}_j^H + \tilde{C}\tilde{C}^H - \underline{K}_j Y_j S_j Y_j \underline{K}_j^H) V_{j+1}^H \\ &\quad - V_{j+1} \pi_j (\underline{H}_j Y_j \underline{K}_j^H + \underline{K}_j Y_j \underline{H}_j^H + \tilde{C}\tilde{C}^H - \underline{K}_j Y_j S_j Y_j \underline{K}_j^H) \pi_j^H V_{j+1}^H \\ &= V_{j+1} (\tilde{\pi}_j \underline{H}_j Y_j \underline{K}_j^H + \underline{K}_j Y_j \underline{H}_j^H \tilde{\pi}_j^H + \tilde{C}\tilde{C}^H - \pi_j \tilde{C}\tilde{C}^H \pi_j^H) V_{j+1}^H, \end{aligned} \tag{4.5}$$

with the projection

$$\tilde{\pi}_j \coloneqq I - \pi_j.$$

By construction, $\tilde{\pi}_j$ is a projection onto the orthogonal complement of the space spanned by the columns of $\underline{L}_j$ along the space spanned by the columns of $\underline{K}_j$. That is,

$$\tilde{\pi}_j = U (W^H U)^{-1} W^H, \tag{4.6}$$

with $W \in \mathbb{C}^{(j+1)p \times p}$ such that $\operatorname{range}(W) = \operatorname{range}(\underline{K}_j)^\perp$ and $U \in \mathbb{C}^{(j+1)p \times p}$ such that $\operatorname{range}(U) = \operatorname{range}(\underline{L}_j)^\perp$.

This allows for a formulation of the residual that reduces the storage requirements for evaluation of the residual from an $n \times n$ matrix to a $2p \times 2p$ matrix, independent of the reduced order $(j+1)p$ of (3.2).

PROPOSITION 4.1. *Let* $\Upsilon \coloneqq \underline{K}_j Y_j \underline{H}_j^H + (I - 0.5\tilde{\pi}_j)\tilde{C}\tilde{C}^H \in \mathbb{C}^{(j+1)p \times (j+1)p}$ *and* $T \coloneqq \Upsilon W (U^H W)^{-1} \in \mathbb{C}^{(j+1)p \times p}$. *Let* $[U \;\; T] = QR$, *with* $Q \in \mathbb{C}^{(j+1)p \times 2p}$ *and* $R \in \mathbb{C}^{2p \times 2p}$ *be an economy-size QR decomposition. Then, for any unitarily invariant norm, it holds that*

$$\|\mathcal{R}(X_j)\| = \left\| R \begin{bmatrix} 0 & I_p \\ I_p & 0 \end{bmatrix} R^H \right\|. \tag{4.7}$$

Proof. The constant term in (4.5) can be written as

$$\begin{aligned} \tilde{C}\tilde{C}^H - \pi_j \tilde{C}\tilde{C}^H \pi_j^H &= 0.5(\tilde{C}\tilde{C}^H - \pi_j \tilde{C}\tilde{C}^H \pi_j^H - \pi_j \tilde{C}\tilde{C}^H + \tilde{C}\tilde{C}^H \pi_j^H) \\ &\quad + 0.5(\tilde{C}\tilde{C}^H - \pi_j \tilde{C}\tilde{C}^H \pi_j^H + \pi_j \tilde{C}\tilde{C}^H - \tilde{C}\tilde{C}^H \pi_j^H) \\ &= \tilde{\pi}_j \tilde{C}\tilde{C}^H (I - 0.5\tilde{\pi}_j)^H + (I - 0.5\tilde{\pi}_j)\tilde{C}\tilde{C}^H \tilde{\pi}_j^H \end{aligned}$$

because of

$$\begin{aligned} 0.5(\tilde{C}\tilde{C}^H - \pi_j \tilde{C}\tilde{C}^H \pi_j^H - \pi_j \tilde{C}\tilde{C}^H + \tilde{C}\tilde{C}^H \pi_j^H) &= 0.5(I - \pi_j)\tilde{C}\tilde{C}^H (I + \pi_j)^H \\ &= 0.5\tilde{\pi}_j \tilde{C}\tilde{C}^H (I + I - \tilde{\pi}_j)^H \\ &= \tilde{\pi}_j \tilde{C}\tilde{C}^H (I - 0.5\tilde{\pi}_j)^H, \end{aligned}$$

and, by a similar argument,

$$0.5(\tilde{C}\tilde{C}^H - \pi_j \tilde{C}\tilde{C}^H \pi_j^H + \pi_j \tilde{C}\tilde{C}^H - \tilde{C}\tilde{C}^H \pi_j^H) = (I - 0.5\tilde{\pi}_j)\tilde{C}\tilde{C}^H \tilde{\pi}_j^H.$$

ETNA
Kent State University and
Johann Radon Institute (RICAM)

Herewith, (4.5) can be further manipulated to obtain

$$\begin{aligned}\mathcal{R}(X_j) &= V_{j+1}\{\tilde{\pi}_j(\underline{H}_j Y_j \underline{K}_j^H + \tilde{C}\tilde{C}^H(I - 0.5\tilde{\pi}_j)^H) \\ &\quad + (\underline{K}_j Y_j \underline{H}_j^H + (I - 0.5\tilde{\pi}_j)\tilde{C}\tilde{C}^H)\tilde{\pi}_j^H\} V_{j+1}^H \\ &= V_{j+1}(\tilde{\pi}\Upsilon^H + \Upsilon\tilde{\pi}^H)V_{j+1}^H,\end{aligned}$$

with $\Upsilon := \underline{K}_j Y_j \underline{H}_j^H + (I - 0.5\tilde{\pi}_j)\tilde{C}\tilde{C}^H$. Hence, for any unitarily invariant norm, we have[1]

$$\|\mathcal{R}(X_j)\| = \|\tilde{\pi}\Upsilon^H + \Upsilon\tilde{\pi}^H\|. \tag{4.8}$$

Making use of (4.6) we can further reduce the size of the matrix whose norm has to be computed. It holds that

$$\tilde{\pi}\Upsilon^H + \Upsilon\tilde{\pi}^H = U(W^H U)^{-1}W^H\Upsilon^H + \Upsilon W(U^H W)^{-1}U^H = UT^H + TU^H$$

with $T = \Upsilon W(U^H W)^{-1}$. As

$$UT^H + TU^H = [U \;\; T]\begin{bmatrix} 0 & I_p \\ I_p & 0 \end{bmatrix}[U \;\; T]^H,$$

we have with the economy-size QR decomposition

$$[U \;\; T] = QR, \quad Q \in \mathbb{C}^{(j+1)p\times 2p},\ R \in \mathbb{C}^{2p\times 2p}$$

for any unitarily invariant norm

$$\|\mathcal{R}(X_j)\| = \left\| R\begin{bmatrix} 0 & I_p \\ I_p & 0 \end{bmatrix} R^H \right\|.$$

This proves our claim. □

The calculation of the residual norm as described above is summarized in Algorithm 2. There is no incremental update formula for how to compute $\|\mathcal{R}(X_j)\|$ from $\|\mathcal{R}(X_{j-1})\|$, but only computations with small-scale matrices of size $(j+1)p \times jp$ and $(j+1)p \times p$ are involved.

Algorithm 2 Residual norm calculation.

Require: $\underline{K}_j, \underline{H}_j, \underline{L}_j \in \mathbb{C}^{(j+1)p\times jp}$, solution Y_j of (3.2), and $\tilde{C} \in \mathbb{C}^{(j+1)p\times p}$ such that $C^H = V_{j+1}\tilde{C}$.
Ensure: Residual norm $\mathcal{R}(X_j)$.
1: Compute basis $U \in \mathbb{C}^{(j+1)p\times p}$ of $\operatorname{range}(\underline{L}_j)^\perp$.
2: Compute basis $W \in \mathbb{C}^{(j+1)p\times p}$ of $\operatorname{range}(\underline{K}_j)^\perp$.
3: Compute $\Gamma = W(U^H W)^{-1} \in \mathbb{C}^{(j+1)p\times p}$.
4: Compute $\Psi = \tilde{C}^H\Gamma \in \mathbb{C}^{(p\times p}$.
5: Compute $T = \underline{K}_j Y_j \underline{H}_j^H\Gamma + (\tilde{C} - 0.5U\Psi^H)\Psi \in \mathbb{C}^{(j+1)p\times p}$.
6: Compute the economy-size QR decomposition $[U \;\; T] = QR$ with $R \in \mathbb{C}^{2p\times 2p}$.
7: Return $\left\| R\begin{bmatrix} 0 & I_p \\ I_p & 0 \end{bmatrix} R^H \right\|$.

[1] The residual formulation (4.8) is a generalization of the expression in [30, Proposition 5.3].

ETNA
Kent State University and
Johann Radon Institute (RICAM)

REMARK 4.2. From

$$\mathcal{R}(X_j) = V_{j+1}(\tilde{\pi}\Upsilon^H + \Upsilon\tilde{\pi}^H)V_{j+1}^H = V_{j+1}(UT^H + TU^H)V_{j+1}^H$$

it follows that the rank of the residual is at most $2p$, as V_{j+1} is of full rank. There are essentially only two scenarios in which the residual rank is smaller. On the one hand, the residual can be of rank p, which happens for the RADI approximate solution as observed in [4]. On the other hand, a rank-0 residual is obtained for an exact solution. The rank-$2p$ property has also been investigated in [34, Section 5.2] for Lyapunov equations. It was called the dilemma of the rational Krylov subspace approach, as in general the norm-minimizing approximate solution yields a residual with larger rank.

5. Truncation of the approximate solution. The general projection method generates an approximate solution of the form

$$X_j = V_{j+1}\underline{K}_j Y_j \underline{K}_j^H V_{j+1}^H.$$

This matrix should be (by construction) Hermitian positive semidefinite with $n - jp$ zero eigenvalues. But due to rounding errors, it may even be indefinite. We propose to use an eigendecomposition of the Hermitian matrix $\underline{K}_j Y_j \underline{K}_j^H$ in order to truncate nonpositive (and possibly some small positive) eigenvalues. This yields a truncated approximate solution $\hat{X}_j$, which is positive semidefinite with at least $n - jp$ zero eigenvalues. Thus, the approximate $\hat{X}_j$ is of lower rank than X_j. This truncated approximate solution can be interpreted as the solution of the Riccati equation projected to a subspace $\hat{\mathcal{K}}_j \subset \mathcal{K}_j$ of dimension $r \leq jp$, which is determined by the decomposition of X_j. Hence, unlike for similar truncated solutions in other projection-based approaches, making use of the derivations in Section 4, the residual norm $\|\mathcal{R}(\hat{X}_j)\|$ can be evaluated cheaply.

Let

$$A^H V_{j+1}\underline{K}_j = V_{j+1}\underline{H}_j \tag{5.1}$$

be an orthonormal BRAD as in (3.3) and let Y_j be the solution of (3.2), so that $X_j = V_{j+1}\underline{K}_j Y_j \underline{K}_j^H V_{j+1}^H$ is the current approximate solution to (1.1). Assume that the Hermitian matrix $\underline{K}_j Y_j \underline{K}_j^H \in \mathbb{C}^{(j+1)p\times(j+1)p}$ has been block-diagonalized by a unitary matrix such that

$$\underline{K}_j Y_j \underline{K}_j^H = \begin{bmatrix}\hat{\underline{Q}}_j & \check{\underline{Q}}_j\end{bmatrix}\begin{bmatrix}\hat{Y}_j & 0\\ 0 & \check{Y}_j\end{bmatrix}\begin{bmatrix}\hat{\underline{Q}}_j^H\\ \check{\underline{Q}}_j^H\end{bmatrix}, \tag{5.2}$$

with $\hat{Y}_j = \hat{Y}_j^H \in \mathbb{C}^{r\times r}$, $\hat{\underline{Q}}_j \in \mathbb{C}^{(j+1)p\times r}$ and $\check{\underline{Q}}_j \in \mathbb{C}^{(j+1)p\times((j+1)p-r)}$, where $\hat{\underline{Q}}_j^H\hat{\underline{Q}}_j = I_r$, $\check{\underline{Q}}_j^H\check{\underline{Q}}_j = I_{(j+1)p-r}$, and $\check{\underline{Q}}_j^H\hat{\underline{Q}}_j = 0_{(j+1)p-r\times r}$. All unwanted (e.g., all nonpositive) eigenvalues of $\underline{K}_j Y_j \underline{K}_j^H$ are assumed to be eigenvalues of $\check{Y}_j$. Hence, $r \leq jp$. As the truncated approximate solution, define

$$\hat{X}_j = V_{j+1}\hat{\underline{Q}}_j\hat{Y}_j\hat{\underline{Q}}_j^H V_{j+1}^H.$$

This can be understood as if $\hat{X}_j$ is given by its economy-size singular value decomposition.

Next we will show that $\hat{X}_j$ is the solution of the Riccati equation (1.1) projected onto the subspace $\hat{\mathcal{K}}_j = \operatorname{range}(V_{j+1}\hat{\underline{Q}}_j)$. In order to do so, we first derive a BRAD-like expression involving $V_{j+1}\hat{\underline{Q}}_j$. Let $T_1 \in \mathbb{C}^{jp\times r}$ be such that $\underline{K}_j T_1 = \hat{\underline{Q}}_j$ holds. As $\underline{K}_j$ is a matrix of full

ETNA
Kent State University and
Johann Radon Institute (RICAM)

column rank, its pseudoinverse $\underline{K}_j^+ = (\underline{K}_j^H \underline{K}_j)^{-1} \underline{K}_j^H$ exists. Thus, we have $T_1 = \underline{K}_j^+ \hat{\underline{Q}}_j$. Define

$$\hat{\underline{H}}_j := \underline{H}_j T_1 \in \mathbb{C}^{(j+1)p \times r}. \tag{5.3}$$

Then postmultiplying (5.1) by T_1 yields the BRAD-like relation

$$A^H V_{j+1} \hat{\underline{Q}}_j = A^H V_{j+1} \underline{K}_j T_1 = V_{j+1} \underline{H}_j T_1 = V_{j+1} \hat{\underline{H}}_j.$$

Let $\hat{Z}_j := V_{j+1} \hat{\underline{Q}}_j \in \mathbb{C}^{n \times r}$. Choose $\hat{\underline{L}}_j \in \mathbb{C}^{(j+1)p \times r}$ in the same way as $\underline{L}_j$ was chosen in Algorithm 1, that is, for example, choose $\hat{\underline{L}}_j = \hat{\underline{K}}_j$, $\hat{\underline{L}}_j = \hat{\underline{H}}_j$, or $\hat{\underline{L}}_j = \hat{\underline{H}}_j - \hat{\underline{K}}_j$ in the cases in which $\underline{L}_j = \underline{K}_j$, $\underline{L}_j = \underline{H}_j$, or $\underline{L}_j = \underline{H}_j - \underline{K}_j$, respectively, was chosen. This guarantees by construction that $\operatorname{rank}(\hat{\underline{L}}_j) = r$, and hence that $\hat{\underline{L}}_j^H \hat{\underline{Q}}_j$ is nonsingular.

Let $\hat{W}_j := V_{j+1} \hat{\underline{L}}_j \in \mathbb{C}^{n \times r}$ and

$$\hat{\Pi}_j := \hat{Z}_j (\hat{W}_j^H \hat{Z}_j)^{-1} \hat{W}_j^H = V_{j+1} \hat{\underline{Q}}_j (\hat{\underline{L}}_j^H \hat{\underline{Q}}_j)^{-1} \hat{\underline{L}}_j^H V_{j+1}^H = V_{j+1} \hat{\pi}_j V_{j+1}^H$$

with $\hat{\pi}_j := \hat{\underline{Q}}_j (\hat{\underline{L}}_j^H \hat{\underline{Q}}_j)^{-1} \hat{\underline{L}}_j^H$. Then $\hat{\Pi}_j$ is an (in general oblique) projection onto $\hat{\mathcal{K}}_j$, while $\hat{\pi}_j$ is a projection on the space spanned by the columns of $\hat{\underline{Q}}_j$. Thus, due to (4.3), we have $\hat{\pi}_j = \hat{\pi}_j \pi_j$.

Now we can state and prove the main statement of this section.

THEOREM 5.1. *The truncated approximate solution* $\hat{X}_j = V_{j+1} \hat{\underline{Q}}_j \hat{Y}_j \hat{\underline{Q}}_j^H V_{j+1}^H$ *satisfies the projected equation*

$$\hat{\Pi}_j \mathcal{R}(\hat{X}_j) \hat{\Pi}_j^H = 0. \tag{5.4}$$

That is, for $\hat{Y}_j \in {}^{r \times r}$, *the equation*

$$\hat{\pi}_j (\hat{\underline{H}}_j \hat{Y}_j \hat{\underline{Q}}_j^H + \hat{\underline{Q}}_j \hat{Y}_j \hat{\underline{H}}_j^H + \tilde{C}\tilde{C}^H - \hat{\underline{Q}}_j \hat{Y}_j \hat{S}_j \hat{Y}_j \hat{\underline{Q}}_j^H) \hat{\pi}_j^H = 0 \tag{5.5}$$

holds, where $\hat{S}_j = \hat{\underline{Q}}_j V_{j+1}^H B B^H V_{j+1} \hat{\underline{Q}}_j^H$.

Proof. As in (4.4), we have for (5.4)

$$0 = \hat{\Pi}_j \mathcal{R}(\hat{X}_j) \hat{\Pi}_j^H = V_{j+1} \hat{\pi}_j (\hat{\underline{H}}_j \hat{Y}_j \hat{\underline{Q}}_j^H + \hat{\underline{Q}}_j \hat{Y}_j \hat{\underline{H}}_j^H + \tilde{C}\tilde{C}^H - \hat{\underline{Q}}_j \hat{Y}_j \hat{S}_j \hat{Y}_j \hat{\underline{Q}}_j^H) \hat{\pi}_j^H V_{j+1}^H.$$

This is equivalent to the small-scale equation (5.5).

We will now prove that $\hat{Y}_j$ fulfills (5.5), which proves the statement of the theorem. In order to do so, we start from the equation

$$0 = \pi_j (\underline{H}_j Y_j \underline{K}_j^H + \underline{K}_j Y_j \underline{H}_j^H + \tilde{C}\tilde{C}^H - \underline{K}_j Y_j S_j Y_j \underline{K}_j^H) \pi_j^H,$$

which is equivalent to (4.4). Pre- and postmultiplication by $\hat{\pi}_j$ and making use of $\hat{\pi}_j \pi_j = \hat{\pi}_j$ gives

$$0 = \hat{\pi}_j (\underline{H}_j Y_j \underline{K}_j^H + \underline{K}_j Y_j \underline{H}_j^H + \tilde{C}\tilde{C}^H - \underline{K}_j Y_j S_j Y_j \underline{K}_j^H) \hat{\pi}_j^H. \tag{5.6}$$

We will see that this equation is equivalent to (5.5). In order to see this, we consider the different terms one by one.

ETNA
Kent State University and
Johann Radon Institute (RICAM)

But first note that, due to (5.2),

$$\hat{\pi}_j \underline{K}_j Y_j \underline{K}_j^H = \begin{bmatrix} \hat{\underline{Q}}_j & 0 \end{bmatrix} \begin{bmatrix} \hat{Y}_j & 0 \\ 0 & \check{Y}_j \end{bmatrix} \begin{bmatrix} \hat{\underline{Q}}_j^H \\ \check{\underline{Q}}_j^H \end{bmatrix} = \hat{\underline{Q}}_j \hat{Y}_j \hat{\underline{Q}}_j^H = \hat{\pi}_j \hat{\underline{Q}}_j \hat{Y}_j \hat{\underline{Q}}_j^H \tag{5.7}$$

holds, as $\hat{\pi}_j \hat{\underline{Q}}_j = \hat{\underline{Q}}_j$ and $\hat{\pi}_j \check{\underline{Q}}_j = 0$.

Now consider the first term in (5.6). Making use of the transpose of (5.7) and of (5.3), it holds that

$$\hat{\pi}_j \underline{H}_j Y_j \underline{K}_j^H \hat{\pi}_j^H = \hat{\pi}_j \underline{H}_j \underline{K}_j^+ \underline{K}_j Y_j \underline{K}_j^H \hat{\pi}_j^H = \hat{\pi}_j \underline{H}_j \underline{K}_j^+ \hat{\underline{Q}}_j \hat{Y}_j \hat{\underline{Q}}_j^H \hat{\pi}_j^H = \hat{\pi}_j \hat{\underline{H}}_j \hat{Y}_j \hat{\underline{Q}}_j^H \hat{\pi}_j^H,$$

as $\underline{K}_j^+ \underline{K}_j = I$ for the pseudoinverse $\underline{K}_j^+$. Thus, the first terms in (5.6) and (5.5) are equivalent.

The second term in (5.6) is just the transpose of the first term and is thus equivalent to the second term in (5.5). The third terms in (5.6) and (5.5) are identical.

For the fourth and last term in (5.6), it holds with (4.1) and (5.7) that

$$\begin{aligned} \hat{\pi}_j \underline{K}_j Y_j S_j Y_j \underline{K}_j^H \hat{\pi}_j^H &= \hat{\pi}_j \underline{K}_j Y_j \underline{K}_j^H V_{j+1}^H B B^H V_{j+1} \underline{K}_j Y_j \underline{K}_j^H \hat{\pi}_j^H \\ &= \hat{\pi}_j \hat{\underline{Q}}_j \hat{Y}_j \hat{\underline{Q}}_j^H V_{j+1}^H B B^H V_{j+1} \hat{\underline{Q}}_j \hat{Y}_j \hat{\underline{Q}}_j^H \hat{\pi}_j^H \\ &= \hat{\pi}_j \hat{\underline{Q}}_j \hat{Y}_j \hat{S}_j \hat{Y}_j \hat{\underline{Q}}_j^H \hat{\pi}_j^H. \end{aligned}$$

Thus, the fourth terms in (5.6) and (5.5) are equivalent. In summary, (5.6) and (5.5) are equivalent. ∎

In the case $\check{Y}_j = 0$, (5.2) reduces to $\underline{K}_j Y_j \underline{K}_j^H = \hat{\underline{Q}}_j \hat{Y}_j \hat{\underline{Q}}_j^H$. Thus, in that case we have $\|\mathcal{R}(X_j)\| = \|\mathcal{R}(\hat{X}_j)\|$ in any unitarily invariant norm. In any other case, it is not clear whether the norm of $\|\mathcal{R}(\hat{X}_j)\|$ will decrease or increase compared to $\|\mathcal{R}(X_j)\|$.

Theorem 5.1 allows us to efficiently evaluate the norm of the Riccati residual for the truncated approximate solution $\hat{X}_j$ in the same way as described in the previous section by using $\hat{\underline{K}}_j = \hat{\underline{Q}}_j$, $\hat{\underline{H}}_j$, $\hat{\underline{L}}_j$, and $\hat{Y}_j$ instead of the values without $\hat{}$. The solution $\hat{Y}_j$ of (5.5) is given by the decomposition (5.2), so no additional small-scale Riccati equation has to be solved; just a block-diagonalization has to be performed. The matrices $\hat{\underline{L}}_j$ and $\hat{\underline{K}}_j$ consist of r columns; thus we have $U, W \in \mathbb{C}^{(j+1)p \times p+r}$ in Algorithm 2. Hence $r < jp$ implies more columns in U and W. The rank of the residual matrix increases from $2p$ to $2(p+r)$. Storing the low-rank factor $\hat{Z}_j = V_{j+1} \hat{\underline{K}}_j$ requires r columns of length n, while storing the low-rank factor $Z_j = V_{j+1} \underline{K}_j$ requires jp columns of length n. Thus, in the case $r < jp$, the low-rank factorization $X_j = \hat{Z}_j \hat{Y}_j \hat{Z}_j^H$ is more efficient storagewise.

In summary, given an approximate $X_j = V_{j+1} \underline{K}_j Y_j \underline{K}_j^H V_{j+1}^H$ as discussed in Section 3, one computes (5.2) and (5.3) and chooses $\hat{\underline{L}}_j$ appropriately in order to compute a new approximate $\hat{X}_j$ of lower rank than X_j. Its residual can be determined using the algorithm for the efficient residual norm evaluation (Algorithm 2). The corresponding calculations need to be inserted after line 13 in Algorithm 1. We will term the so-modified Algorithm 1 the "truncation algorithm".

6. Generalized Riccati equations. For generalized Riccati equations

$$A^H X E + E^H X A + C^H C - E^H X B B^H X E = 0 \tag{6.1}$$

with an additional nonsingular matrix $E \in \mathbb{C}^{n \times n}$, it was noted in [4, Section 4.4] that the equivalent Riccati equation

$$\mathcal{R}_{\text{gen}}(X) = E^{-H} A^H X + X A E^{-1} + E^{-H} C^H C E^{-1} - X B B^H B = 0$$

ETNA
Kent State University and
Johann Radon Institute (RICAM)

has the same structure as (1.1) in which the system matrix A and the initial residual factor C^H are replaced by AE^{-1} and $E^{-H}C^H$, respectively. In an efficient algorithm, inverting E is avoided. The orthonormal BRAD used in Algorithm 1 becomes

$$E^{-H}A^H V_{j+1}\underline{K}_j = V_{j+1}\underline{H}_h \quad \Longleftrightarrow \quad A^H V_{j+1}\underline{K}_j = E^H V_{j+1}\underline{H}_j$$

with the starting block $E^{-H}C^H$. The expression (4.2) for $\mathcal{R}(X_j)$ becomes

$$\mathcal{R}(X_j) = E^H V_{j+1}(\underline{H}_j Y_j \underline{K}_j^H + \underline{K}_j Y_j \underline{H}_j^H + \tilde{C}\tilde{C}^H - \underline{K}_j Y_j S_j Y_j \underline{K}_j^H)V_{j+1}^H E,$$

with $C^H = E^H V_{j+1}\tilde{C}$, such that the subsequent derivations in Section 3 all hold without any changes.

The expression for the residual becomes

$$E^H V_{j+1}(UT^H + TU^H)V_{j+1}^H E.$$

In order to evaluate this efficiently, let $QR = E^H V_{j+1}\,[U\ T]$ be an economy-size QR decomposition. Then

$$\|E^H V_{j+1}(UT^H + TU^H)V_{j+1}^H E\| = \left\| R \begin{bmatrix} 0 & I_p \\ I_p & 0 \end{bmatrix} R^H \right\|$$

is the residual norm of (6.1); see (4.7).

7. Numerical experiments. An extensive comparison of low-rank factored algorithms for solving (1.1) has been presented in [4, 5]. We complement those findings by comparing Algorithm 1 (employing the efficient residual computation as in Algorithm 2 and truncating the approximate solution as discussed in Section 5), the RADI algorithm from [4], and the RKSM algorithm from [15, 30] with respect to their convergence performance. Recall that RADI generates approximations $X_k^{\text{radi}} = Z_k Y_k^{\text{radi}} Z_k^H$, where the columns of Z_k can be interpreted as a nonorthogonal basis of $\mathcal{K}_k = \mathcal{K}_k(A^H, C^H, \mathcal{S}_k)$ (1.5). The approximations $X_k = Z_k Y_k Z_k^H$ generated via Algorithm 1 have the same structure, just the matrix Y_k being obtained in a different way. Moreover, X_j^{radi} can be interpreted as the solution of a projection of the large-scale Riccati equation (1.1) onto the Krylov subspace $\mathcal{K}_k$ employing an oblique projection; see [11, Section 4.2]. In contrast, RKSM computes an approximate solution $X_k^{\text{rksm}} = Q_k Y_k^{\text{rksm}} Q_k^H$, where the columns of Q_k span an orthogonal basis of $\mathfrak{K}_k = \kappa_k(A^H, C^H, \mathcal{S}_{k-1})$ (1.4) and Y_k^{rksm} is the solution of (1.2) resulting from the Galerkin projection $Q_k^H \mathcal{R}(X_k) Q_k = 0$ of the Riccati residual.

The behavior of Algorithm 1 for the choices $\underline{L}_j = \underline{K}_j$ (which yields a Galerkin projection) and $\underline{L}_j = \underline{H}_j$ or $\underline{L}_j = \underline{H}_j - \underline{K}_j$ (which yield a Petrov–Galerkin projection) and different shift strategies is investigated. That is, in order to see the influence of different shift strategies on Algorithm 1, a set of shifts is precalculated. Then all three algorithms are run with this same set of shifts so that all algorithms perform the same number of iteration steps with the same shifts. Their convergence behaviors are compared. Our purpose is not to propose Algorithm 1 in its current version as a valid competitor of RADI or RKSM. In theory, the Petrov–Galerkin approach gives more degrees of freedom than the Galerkin approach. With an optimal choice of L_k and suitable shifts, the Petrov–Galerkin version of Algorithm 1 should converge at least as fast as or faster than the Galerkin version. But, so far, we do not know how to best choose L_k and the set of shifts.

For the RADI algorithm, we use the function `mess_lrradi` from the MATLAB toolbox M.E.S.S.-2.2 [29]. The implementation of the RKSM method used is based on the function `RKSMa_care` from [5]. The function `RKSMa_care` had to be modified in order to handle

ETNA
Kent State University and
Johann Radon Institute (RICAM)

precomputed shifts correctly. The modified code as well as all experimental code used to generate the results presented can be found at [17].

We consider three different shift strategies. For two of these, `mess_lrradi` is used. The shifts are precomputed employing either the default option "`opts.shifts.method = 'gen-ham-opti';`", which implies that the shift strategy residual Hamiltonian shifts [4, Section 4.5.2] are used, or the option "`opts.shifts.method = 'heur';`", which yields an estimation of suboptimal ADI shift parameters as suggested by Penzl [19, 25]. The third set is computed using the default option "`conv`" of the function `RKSMa_care`, which chooses the shifts from the convex hull of Ritz values [15].

Algorithms 1 and 2 have been implemented in order to handle generalized Riccati equations, as discussed in Section 6. The entire orthonormal BRAD (3.3) associated to the precomputed set of shifts is generated prior to the start of the iteration in Algorithm 1 using the function `rat_krylov` from the Rational Krylov Toolbox [9] in version 2.9. During the iteration, only the relevant parts of the BRAD are used. The `rat_krylov` function supports generalized Riccati equations with an additional system matrix E, block vectors $C^H \in \mathbb{C}^{n\times p}$, and realification in case complex shifts are used in conjugate-complex pairs. All occurring small-scale Riccati equations are solved with MATLAB's `icare`. For the vast majority of the small-scale Riccati equations to be solved, MATLAB's `icare` reported back that the unique solution generated is accurate (`info.Report == 0`). Nonetheless, some of the computed solutions Y_j were indefinite. In our implementation, $\check{Y}_j$ in (5.2) was chosen to contain all eigenvalues less than $10^{-12}\rho(\underline{K}_j Y_j \underline{K}_j^H)$, where $\rho(\cdot)$ denotes the spectral radius of the matrix. In order to do so, the matrix $\underline{K}_j Y_j \underline{K}_j^H$ is diagonalized using MATLAB's `eig`. The eigenvalues (and corresponding eigenvectors) are reordered such that the eigenvalues appear in descending order.

Algorithm 1 has been implemented in two versions: an efficient version that takes into account the comments from Section 3.1, as well as a version that directly implements Algorithm 1. The efficient version of Algorithm 1 does require $\underline{K}_j$, $\underline{H}_j$, and $\underline{L}_j$ to be block upper Hessenberg matrices. The function `rat_krylov` may return $\underline{K}_j$ and $\underline{H}_j$ with additional entries in the case where deflation was performed. In those cases, we make use of the second version of the implementation of Algorithm 1. In a really efficient implementation, the calculation of the BRAD would have to be linked to the successive solution of the Riccati equation, so that the BRAD is not calculated in advance, but step by step, immediately followed by the solution of the corresponding small Riccati equation. Aspects such as deflation and reorthogonalization would then be adapted to the problem at hand. But, for our purpose, the approach taken here is sufficient.

In our test setup, RADI will, in general, be (much) faster than RKSM and Algorithm 1, as each algorithm is run for the same number of iteration steps with the same set of shifts. The most time-consuming part of the RADI algorithm is the solution of linear systems with multiple right-hand sides. The rest of the computational effort is negligible. Assuming that, as proposed in [4, Section 4.2], the Sherman–Morrison–Woodbury formula is used to reformulate the dense linear systems in the RADI algorithm as sparse linear systems of the form $(A^H - \sigma I)\hat{Z} = R$, linear systems with $p+m$ right-hand sides have to be solved. RKSM and Algorithm 1 need to solve the same linear systems of equations with just p right-hand sides in order to advance the required basis of the block rational Krylov subspace. However, this slight advantage is more than offset by the necessary orthonormalization of the basis in RKSM and Algorithm 1. Moreover, unlike in the RADI algorithm, in RKSM and Algorithm 1 the solution of the small-scale Riccati equation (3.2) of growing dimension has to be calculated, which further increases the computational time required. Nevertheless, for the first example we present time measurements to illustrate the difference between RKSM, Algorithm 1, and

ETNA
Kent State University and
Johann Radon Institute (RICAM)

RADI and especially between RKSM and Algorithm 1.

All experiments are performed in MATLAB R2024a on an Intel® Core™ i7-8565U CPU @ 1.80 GHz 1.99 GHz with 16 GB RAM.

7.1. Example 1. The first example considered is the well-known steel profile cooling model from the Oberwolfach Model Reduction Benchmark Collection [7, 23]. This example (often termed RAIL) comes in different problem sizes n, but fixed $m = 7$ and $p = 6$. We used the one with $n = 79{,}841$. The system matrices E and A are symmetric positive and negative definite, respectively.

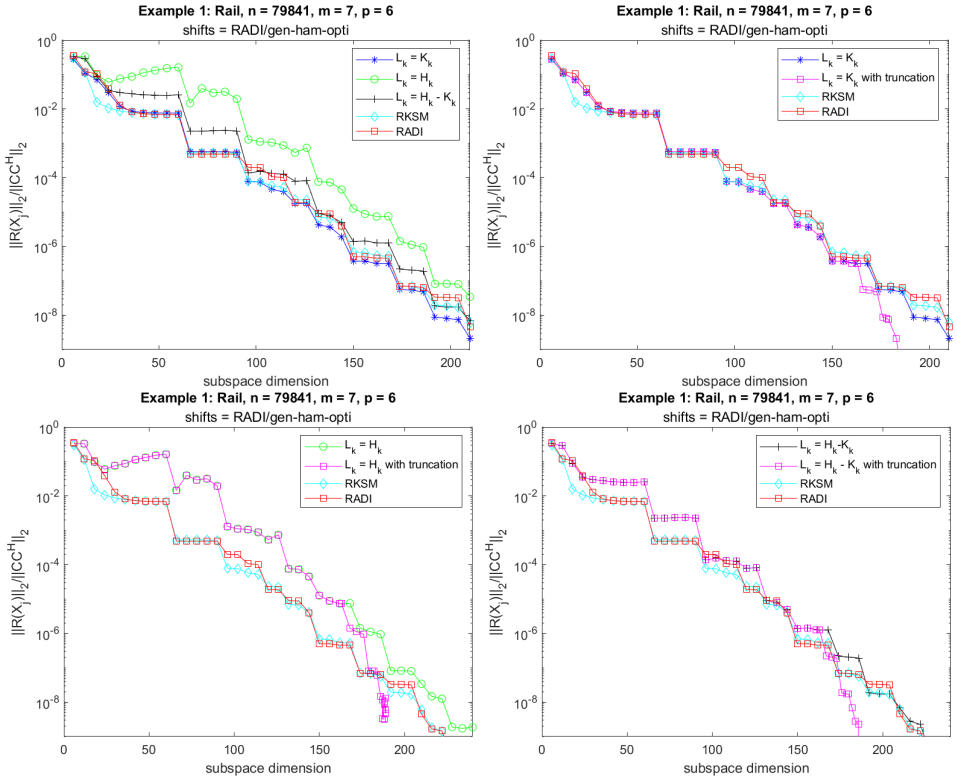

FIG. 7.1. *Example 1: Relative residual norms for shifts generated by* `mess_lrradi` *with option* `'gen-ham-opti'`.

As a first test, we precomputed one set of shifts employing `mess_lrradi` with the default option `'gen-ham-opti';`. For all other options used, we refer to [17]. This resulted in 50 real shifts. Then we ran all algorithms with this set of shifts. All plots in Figure 7.1 display the residual norm normalized by the norm of CC^H versus the dimension of the subspace projected onto for all methods considered. The plot in the upper left-hand side shows that Algorithm 1 with the choice $\underline{L}_j = \underline{H}_j$ and $\underline{L}_j = \underline{H}_j - \underline{K}_j$ converges a bit slower than RADI and RKSM. For the choice $\underline{L}_j = \underline{K}_j$ Algorithm 1 performs slightly better than RADI and (in the end) than RKSM. The other plots in Figure 7.1 show the effect of the truncation on the convergence. To make the effect clearly visible, we display the results for truncation together with three of the residual norm curves from the top-left plot in Figure 7.1, namely, the convergence of RADI, of RKSM, and of Algorithm 1 for one of the three choices of $\underline{L}_j$.

ETNA
Kent State University and
Johann Radon Institute (RICAM)

In addition, the convergence of the truncation algorithm is given for the selected L_k. The truncated approximate solution $\hat{X}_j$ is computed from X_j as explained in Section 5. For illustration purposes, this is done in each iteration step. Clearly, in the beginning no truncation is taking place, but after 26 iteration steps (that is, BRAD subspace dimension 162) truncation shows an effect. As can be seen in Figure 7.1 and Table 7.1, when employing truncation of the computed approximate solution X_j, the subspace dimensions decrease after a while such that the low-rank factorization of $\hat{X}_j$ requires less storage than that of X_j. Recall that when applying Algorithm 1, the dimension of the subspace on which the Riccati equation is projected increases by p in each iteration step, that is, in step j, it is $6j$. Moreover, in each iteration step, the rank of the Riccati residual $\mathcal{R}(X_j)$ is given by $2p = 12$. See Table 7.1 for detailed information on the dimension of the subspace used and the rank of the Riccati residual. In the end, instead of a $79{,}841 \times 300$ matrix to store $V_{j+1}\underline{K}_j$, only one of size $79{,}841 \times 187$, $79{,}841 \times 188$, or $79{,}841 \times 188$ is required, depending on the choice of $\underline{L}_j$.

TABLE 7.1
Example 1: Rank($\mathcal{R}(X_j)$) and subspace dimension for the set of `'gen-ham-opti'` *shifts as in Figure 7.1.*

j	$\underline{L}_j = \underline{K}_j$		$\underline{L}_j = \underline{H}_j$		$\underline{L}_j = \underline{H}_j - \underline{K}_j$	
	rank($\mathcal{R}(X_j)$)	subspace dimension	rank($\mathcal{R}(X_j)$)	subspace dimension	rank($\mathcal{R}(X_j)$)	subspace dimension
1–26	12	$6j$	12	$6j$	12	$6j$
27	16	160	12	162	14	161
28	22	163	22	163	22	163
29	28	166	24	168	26	167
30	34	169	28	172	32	170
31	38	173	32	176	38	173
32	44	176	38	179	44	176
33	52	178	46	181	50	179
34	62	179	56	182	60	180
35	66	183	64	184	68	182
36	76	184	72	186	76	184
37	88	184	78	189	84	186
38	98	185	92	188	98	185
39	110	185	102	189	106	187
40	120	186	114	189	118	187
41	132	186	128	188	130	187
42	144	186	140	188	142	187
43	154	187	154	187	154	187
44	166	187	166	187	166	187
45	178	187	176	188	176	188
46	190	187	188	188	188	188
47	202	187	200	188	200	188
48	214	187	212	188	212	188
49	226	187	224	188	214	188
50	238	187	236	188	238	188

ETNA
Kent State University and
Johann Radon Institute (RICAM)

Figure 7.2 displays the same information as the left upper plot in Figure 7.1, but with different precomputed shifts. The left plot shows the convergence related to the shifts generated with `my_RKSMa_care` and the option `'convR'`; the right one to the convergence related to the shifts generated with `mess_lrradi` and the option `'heur'`. All precomputed shifts are real. For both set of shifts, RKSM performs better than the other algorithms. Algorithm 1 with the choice $\underline{L}_j = \underline{K}_j$ converges slightly faster than RADI, but a bit slower than RKSM. For the set of RKSM shifts, the other versions of Algorithm 1 perform worse than the other algorithms, but for the set of `'heur'` shifts, the convergence for the choice $L_k = H_k - K_k$ is better than RADI and almost like the choice $L_k = K_k$, while for the choice $\underline{L}_j = \underline{H}_j$ the overall performance is better than that of RADI, although this does not apply to some iteration steps. Truncation has almost no effect for the set of RKSM shifts, while for the set of `'heur'` shifts the effect can be nicely seen from BRAD subspace dimension 162 onwards; see Figure 7.3.

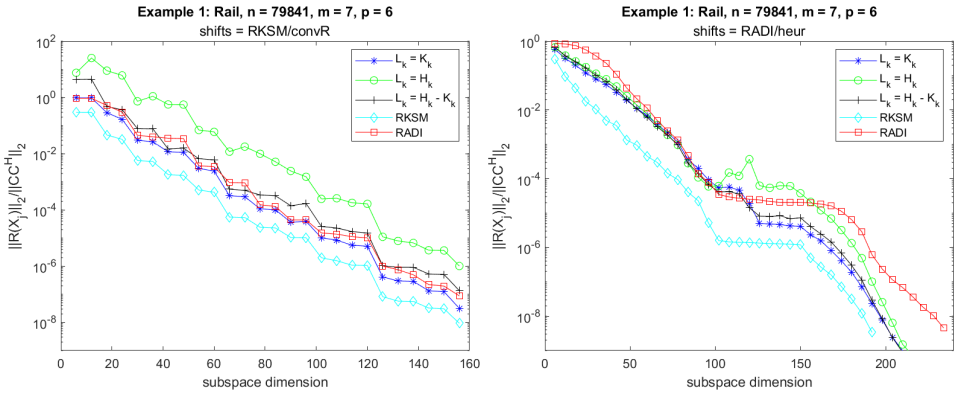

FIG. 7.2. *Example 1: Relative residual norms for different shifts.*

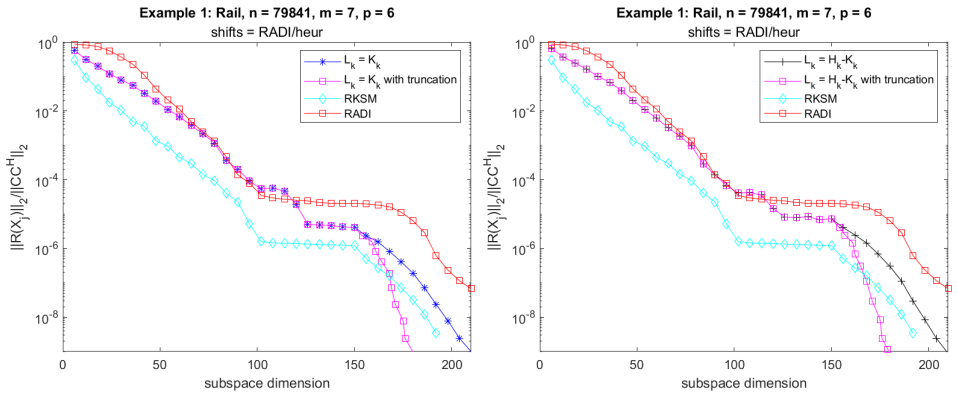

FIG. 7.3. *Example 1: Relative residual norms for shifts generated by* `mess_lrradi` *and the option* `'heur'`.

ETNA
Kent State University and
Johann Radon Institute (RICAM)

Finally, we give some timings for the different algorithms and sets of shifts; see Table 7.2. Algorithm 1 is used with the efficient implementation discussed in Section 3.1 without any truncation. The number of shifts is chosen such that the algorithms terminate with a comparable final accuracy. Due to the setup of the experiments, RADI is the fastest algorithm, as both other algorithms need to perform some orthogonalization to set up the required basis. As the timings show, Algorithm 1 requires typically less time than RKSM.

TABLE 7.2
Example 1: Timings for different sets of shifts.

Shift choice	Method	Timings	No. of shifts
RKSM	RKSM	20.1088	26 real
	RADI	12.1011	
	Alg. 1, $L_k = K_k$	15.8956	
	Alg. 1, $L_k = H_k$	15.8769	
	Alg. 1, $L_k = H_k - K_k$	15.8989	
`'gen-ham-opti'`	RKSM	31.6462	32 real
	RADI	12.4941	
	Alg. 1, $L_k = K_k$	26.7037	
	Alg. 1, $L_k = H_k$	26.8429	
	Alg. 1, $L_k = H_k - K_k$	27.6092	
`'heur'`	RKSM	49.8927	32 real
	RADI	18.8875	
	Alg. 1, $L_k = K_k$	26.5591	
	Alg. 1, $L_k = H_k$	26.2650	
	Alg. 1, $L_k = H_k - K_k$	26.4021	

7.2. Example 2. The second example considered is the convection–diffusion benchmark example from MORwiki – Model Order Reduction Wiki [25, 32]. The examples are constructed with

```
A = fdm_2d_matrix(100,'10*x','100*y','0');
B = fdm_2d_vector(100,'.1<x<=.3');
C = fdm_2d_vector(100,'.7<x<=.9')';
E = speye(size(A));
```

resulting in a SISO (single-input single-output) system of order $n = 10{,}000$. The plots in Figure 7.4 display essentially the same information as the corresponding plots in Figure 7.2. For this example, for the set of shifts generated with `my_RKSMa_care` and the option `'conv'`, all versions of Algorithm 1 perform at least as good as RKSM, while for the set of shifts generated with `mess_lrradi` and the option `'gen-ham-opti'`, the performance of all versions of Algorithm 1 lie between those of RKSM and RADI. Truncation has essentially no effect for the RKSM shifts, while for the `'gen-ham-opti'` shifts, starting from the BRAD subspace dimension 25, truncation has an effect.

For the set of shifts generated with `mess_lrradi` and the option `'heur'`, RADI performs worse than the other algorithms, which perform alike; see Figure 7.5. Truncating the solution does decrease the final subspace dimension, but also decreases the accuracy of the computed solution. This is depicted here for the choice $L_k = H_K$.

ETNA
Kent State University and
Johann Radon Institute (RICAM)

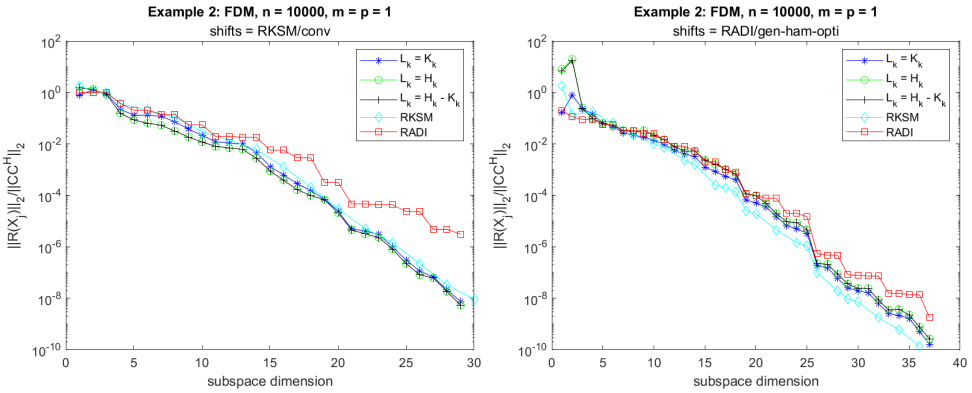

FIG. 7.4. *Example 2: Relative residual norms for different sets of shifts.*

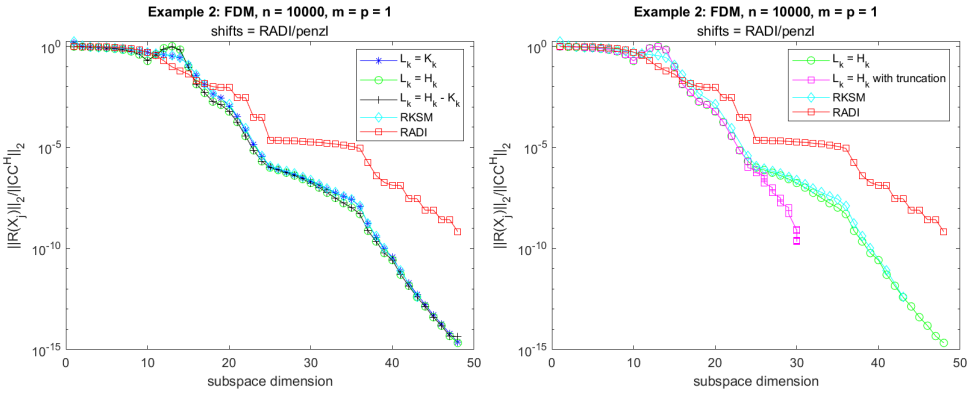

FIG. 7.5. *Example 2: Relative residual norms for* `'heur'` *shifts generated with* `mess_lrradi`.

7.3. Example 3. As a third example, we consider the nonsymmetric matrix "lung2" available from The SuiteSparse Matrix Collection[2] (formerly known as the University of Florida Sparse Matrix Collection) via the matrix ID 894 [13], modeling processes in the human lung. We employ this example with the negated system matrix $-A \in \mathbb{R}^{109460\times 109460}$, $E = I$, and randomly chosen $C^H \in \mathbb{R}^{109460\times 3}$ and $B \in \mathbb{R}^{109460\times 15}$ (using `randn`). Here we report only on the numerical experiments involving the precomputed shifts using the option `'gen-ham-opti'` for `mess_lrradi`. When using the shifts precomputed with the `'heur'` option, MATLAB's `icare` had trouble solving the small-scale Riccati equations. RKSM also had problems converging. Figure 7.6 displays the same information as Figure 7.1. As can be seen, Algorithm 1 with the choice $\underline{L}_k = \underline{K}_j$ outperforms RKSM and RADI slightly, while for the choice $\underline{L}_k = \underline{H}_j - \underline{K}_j$, RKSM and RADI perform slightly better. Algorithm 1 with the choice $\underline{L}_k = \underline{H}_j$ behaves quite differently from the other two choices for $\underline{L}_j$. For this choice, only erratic and poor convergence can be observed.

[2] https://sparse.tamu.edu

ETNA
Kent State University and
Johann Radon Institute (RICAM)

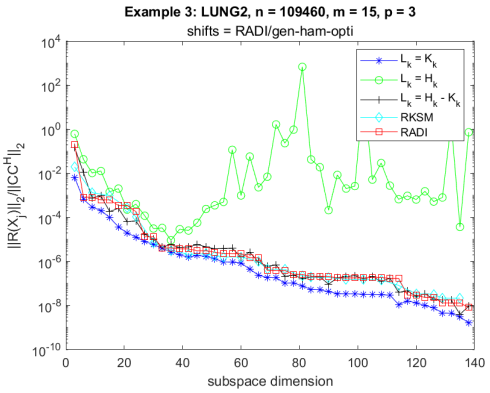

FIG. 7.6. *Example 3: Relative residual norms for shifts generated using the option* `'gen-ham-opti'` *for* `mess_lrradi`.

7.4. Summary of findings. The convergence of the general projection method typically behaves comparably to that of the RADI and the RKSM algorithms. The general projection method with orthogonal projection ($\underline{L}_j = \underline{K}_j$) shows the best convergence behavior among all the variants tested, usually slightly better than the RADI algorithm. This may be due to the fact that all shift choices tested are stemming from Galerkin projection type methods. A choice adapted to the Petrov–Galerkin scenario could possibly provide a remedy and better performance for the Petrov–Galerkin projection approach. The general projection method is computationally more demanding than the RADI algorithm, but somewhat cheaper than RKSM. Truncation of the approximate solution X_j proves to be an efficient way to further reduce the rank of the approximate solution while improving the accuracy of the approximate solution achieved for the related subspace dimension. The choice of criterion for determining $\tilde{Y}_j$ (5.2) determines the achievable accuracy of the truncated approximate solution.

8. Concluding remarks. So far, the orthogonal projection onto the block rational Krylov subspace $\mathfrak{K}_j$ (1.4) or onto the extended block Krylov subspace $\kappa_j(A^H, C^H) + \kappa_j(A^{-H}, A^{-H}C^H)$ for the standard block Krylov subspace $\kappa_j(A^H, C^H)$ (1.3) has been considered in the literature in connection with projection methods for solving large Riccati equations. Here, for the first time, we have explicitly considered the projection of the Riccati equation onto the block rational Krylov subspace $\mathcal{K}_j$ (1.5). The projections need not be orthogonal. Like the resulting projected small-scale Riccati equation, the projections are determined by the matrices in the BRAD corresponding to $\mathfrak{K}_j$. An efficient way to evaluate the norm of the residual has been derived. Instead of the norm of an $n \times n$ matrix, the norm of a readily available $2p \times 2p$ matrix has to be computed. This implies that the rank of the residual matrix is $2p$. The idea of truncating the resulting approximate solution has been proposed. By employing this idea, the rank of the approximate solution can be effectively reduced further while increasing the accuracy obtained for the corresponding subspace dimension.

It has been proven that the truncated approximate solution can be interpreted as the solution of the Riccati residual projected to a subspace of the Krylov subspace $\mathcal{K}_j$. This gives a way to efficiently evaluate the norm of the resulting residual. Numerical experiments demonstrate that the convergence of the proposed projection methods generally follows the same pattern as the convergence of the RADI and the RKSM algorithms. Among all the evaluated versions, the general projection method with orthogonal projection exhibits the best convergence behavior. The Petrov–Galerkin scenario allows for more freedom in the choice of the search space. This could possibly provide a remedy and better performance for the

ETNA
Kent State University and
Johann Radon Institute (RICAM)

Petrov–Galerkin projection approach when a suitable strategy for choosing $\underline{L}_j$ and the set of shifts has been found. However, the RADI approach requires less computational time than the projection method.

Acknowledgements. Part of this work was done while the second author visited the Oden Institute at the University of Texas at Austin in October 2023 and the Department of Mathematics and the Division of Computational Modeling and Data Analytics (CMDA) in the College of Science at Virginia Tech in Blacksburg in November 2023.

REFERENCES

[1] L. AMODEI AND J.-M. BUCHOT, *An invariant subspace method for large-scale algebraic Riccati equation*, Appl. Numer. Math., 60 (2010), pp. 1067–1082.
[2] P. BENNER AND T. BREITEN, *Rational interpolation methods for symmetric Sylvester equations*, Electron. Trans. Numer. Anal., 42 (2014), pp. 147–164. https://etna.ricam.oeaw.ac.at/vol.42.2014/pp147-164.dir/pp147-164.pdf
[3] P. BENNER AND Z. BUJANOVIĆ, *On the solution of large-scale algebraic Riccati equations by using low-dimensional invariant subspaces*, Linear Algebra Appl., 488 (2016), pp. 430–459.
[4] P. BENNER, Z. BUJANOVIĆ, P. KÜRSCHNER, AND J. SAAK, *RADI: a low-rank ADI-type algorithm for large scale algebraic Riccati equations*, Numer. Math., 138 (2018), pp. 301–330.
[5] ———, *A numerical comparison of different solvers for large-scale, continuous-time algebraic Riccati equations and LQR problems*, SIAM J. Sci. Comput., 42 (2020), pp. A957–A996.
[6] P. BENNER, M. HEINKENSCHLOSS, J. SAAK, AND H. K. WEICHELT, *An inexact low-rank Newton-ADI method for large-scale algebraic Riccati equations*, Appl. Numer. Math., 108 (2016), pp. 125–142.
[7] P. BENNER AND J. SAAK, *Linear-quadratic regulator design for optimal cooling of steel profiles*, Technical Report SFB393/05-05, Sonderforschungsbereich 393, TU Chemnitz, Chemnitz, 2005.
[8] ———, *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.
[9] M. BERLJAFA, S. ELSWORTH, AND S. GÜTTEL, *A rational Krylov toolbox for MATLAB*, MIMS EPrint 2014.56, Manchester Institute for Mathematical Sciences, University of Manchester, Manchester, 2014.
[10] C. BERTRAM, *Efficient Solution of Large-Scale Riccati Equations and an ODE Framework for Linear Matrix Equations*, Ph.D. Thesis, Technische Universität Braunschweig, Braunschweig, Germany, 2021. https://nbn-resolving.org/urn:nbn:de:gbv:084-2021110311426
[11] C. BERTRAM AND H. FASSBENDER, *On a family of low-rank algorithms for large-scale algebraic Riccati equations*, Linear Algebra Appl., 687 (2024), pp. 38–67.
[12] D. A. BINI, B. IANNAZZO, AND B. MEINI, *Numerical Solution of Algebraic Riccati Equations*, SIAM, Philadelphia, 2012.
[13] T. A. DAVIS AND Y. HU, *The University of Florida Sparse Matrix Collection*, ACM Trans. Math. Software, 38 (2011), Art. 1, 25 pages.
[14] V. DRUSKIN AND L. KNIZHNERMAN, *Extended Krylov subspaces: approximation of the matrix square root and related functions*, SIAM J. Matrix Anal. Appl., 19 (1998), pp. 755–771.
[15] V. DRUSKIN AND V. SIMONCINI, *Adaptive rational Krylov subspaces for large-scale dynamical systems*, Systems Control Lett., 60 (2011), pp. 546–560.
[16] S. ELSWORTH AND S. GÜTTEL, *The block rational Arnoldi method*, SIAM J. Matrix Anal. Appl., 41 (2020), pp. 365–388.
[17] H. FASSBENDER, *Matlab Code for "A class of Petrov-Galerkin Krylov methods for algebraic Riccati equations"*, Software, 2024. See https://doi.org/10.5281/zenodo.13332981.
[18] M. HEYOUNI AND K. JBILOU, *An extended block Arnoldi algorithm for large-scale solutions of the continuous-time algebraic Riccati equation*, Electron. Trans. Numer. Anal., 33 (2008/09), pp. 53–62. https://etna.ricam.oeaw.ac.at/vol.33.2008-2009/pp53-62.dir/pp53-62.pdf
[19] P. KÜRSCHNER, *Efficient Low-Rank Solution of Large-Scale Matrix Equations*, Ph.D. Thesis, Fakultät für Mathematik, Otto von Guericke Universität, Magdeburg, 2016. http://hdl.handle.net/11858/00-001M-0000-0029-CE18-2
[20] L. KNIZHNERMAN AND V. SIMONCINI, *A new investigation of the extended Krylov subspace method for matrix function evaluations*, Numer. Linear Algebra Appl., 17 (2010), pp. 615–638.
[21] P. LANCASTER AND L. RODMAN, *Algebraic Riccati Equations*, Clarendon Press, New York, 1995.
[22] Y. LIN AND V. SIMONCINI, *A new subspace iteration method for the algebraic Riccati equation*, Numer. Linear Algebra Appl., 22 (2015), pp. 26–47.

ETNA
Kent State University and
Johann Radon Institute (RICAM)

[23] OBERWOLFACH BENCHMARK COLLECTION, *Steel profile*, hosted at MORwiki – Model Order Reduction Wiki, 2005.
https://morwiki.mpi-magdeburg.mpg.de/morwiki/index.php/Steel_Profile

[24] D. PALITTA, *The projected Newton-Kleinman method for the algebraic Riccati equation*, Preprint on arXiv, 2019. https://arxiv.org/abs/1901.10199

[25] T. PENZL, *Lyapack – a MATLAB toolbox for large Lyapunov and Riccati equations, model reduction problems, and linear–quadratic optimal control problems*, Version 1.0., Software, NETLIB, 1999.

[26] A. RUHE, *Rational Krylov sequence methods for eigenvalue computation*, Linear Algebra Appl., 58 (1984), pp. 391–405.

[27] ———, *Rational Krylov algorithms for nonsymmetric eigenvalue problems. II. Matrix pairs*, Linear Algebra Appl., 197/198 (1994), pp. 283–295.

[28] ———, *Rational Krylov: A practical algorithm for large sparse nonsymmetric matrix pencils*, SIAM J. Sci. Comput., 19 (1998), pp. 1535–1551.

[29] J. SAAK, M. KÖHLER, AND P. BENNER, *M-M.E.S.S.-2.2—the matrix equations sparse solvers library*, Software, February 2022.
See https://doi.org/10.5281/zenodo.5938237 or
https://www.mpi-magdeburg.mpg.de/projects/mess.

[30] V. SIMONCINI, *Analysis of the rational Krylov subspace projection method for large-scale algebraic Riccati equations*, SIAM J. Matrix Anal. Appl., 37 (2016), pp. 1655–1674.

[31] V. SIMONCINI, D. B. SZYLD, AND M. MONSALVE, *On two numerical methods for the solution of large-scale algebraic Riccati equations*, IMA J. Numer. Anal., 34 (2014), pp. 904–920.

[32] THE MORWIKI COMMUNITY, *MORwiki – Model Order Reduction Wiki*, Web Resource.
http://modelreduction.org

[33] B. VANDEREYCKEN AND S. VANDEWALLE, *A Riemannian optimization approach for computing low-rank solutions of Lyapunov equations*, SIAM J. Matrix Anal. Appl., 31 (2010), pp. 2553–2579.

[34] T. WOLF, $\mathcal{H}_2$ *Pseudo-Optimal Model Order Reduction*, Ph.D. Thesis, MAS Maschinenbau, Technische Universität München, München, 2014.

[35] N. WONG AND V. BALAKRISHNAN, *Quadratic alternating direction implicit iteration for the fast solution of algebraic Riccati equations*, in 2005 International Symposium on Intelligent Signal Processing and Communication Systems, IEEE Conference Proceedings, Los Alamitos, 2005, pp. 373–376.

[36] ———, *Fast positive-real balanced truncation via quadratic alternating direction implicit iteration*, IEEE Trans. Computer-Aided Design Integr. Circ. Systems., 26 (2007), pp. 1725–1731.