Using LDL^{T} factorizations in Newton's method for solving general large-scale algebraic Riccati equations

Jens Saak* Steffen W. R. Werner†

*Max Planck Institute for Dynamics of Complex Technical Systems, Sandtorstraße 1, 39106 Magdeburg, Germany.
Email: saak@mpi-magdeburg.mpg.de, ORCID: 0000-0001-5567-9637
†Department of Mathematics and Division of Computational Modeling and Data Analytics, Academy of Data Science, Virginia Tech, Blacksburg, VA 24061, USA.
Email: steffen.werner@vt.edu, ORCID: 0000-0003-1667-4862

Abstract: Continuous-time algebraic Riccati equations can be found in many disciplines in different forms. In the case of small-scale dense coefficient matrices, stabilizing solutions can be computed to all possible formulations of the Riccati equation. This is not the case when it comes to large-scale sparse coefficient matrices. In this paper, we provide a reformulation of the Newton-Kleinman iteration scheme for continuous-time algebraic Riccati equations using indefinite symmetric low-rank factorizations. This allows the application of the method to the case of general large-scale sparse coefficient matrices. We provide convergence results for several prominent realizations of the equation and show in numerical examples the effectiveness of the approach.

Keywords: Riccati equation, Newton's method, large-scale sparse matrices, low-rank factorization, indefinite terms

Mathematics subject classification: 15A24, 49M15, 65F45, 65H10, 93A15

Novelty statement: In this work, we use indefinite symmetric low-rank factorizations to extend the Newton-Kleinman iteration to general Riccati equations with large-scale sparse coefficient matrices. We provide a convergence theory for several prominent realizations of the equation and investigate different scenarios numerically.

1 Introduction

The solutions to continuous-time algebraic Riccati equations (CAREs) play essential roles for many concepts in systems and control theory. They occur, for example, in the design of optimal and robust regulators for dynamical processes [2, 40, 47, 61], model order reduction methods for dynamical systems [26, 31, 36, 52], network analysis [3] or applications with differential games [7, 30]. In general, CAREs are quadratic matrix equations of the form

$$A^{\mathsf{T}}XE + E^{\mathsf{T}}XA + C^{\mathsf{T}}QC - \left(B^{\mathsf{T}}XE + S^{\mathsf{T}}\right)^{\mathsf{T}} R^{-1} \left(B^{\mathsf{T}}XE + S^{\mathsf{T}}\right) = 0, \tag{1}$$

with $A, E \in \mathbb{R}^{n\times n}$, $B, S \in \mathbb{R}^{n\times m}$, $C \in \mathbb{R}^{p\times n}$, $Q = Q^\mathsf{T} \in \mathbb{R}^{p\times p}$ and $R = R^\mathsf{T} \in \mathbb{R}^{m\times m}$ invertible. For simplicity of illustration, we present the proposed algorithm and results for the case that E is invertible; however, we outline modifications for the case of non-invertible E matrices in Section 3.5. Among all the solutions to (1), the one of particular interest in most cases is the stabilizing solution, here denoted as $X_* \in \mathbb{R}^{n\times n}$, for which it holds that the eigenvalues of the generalized matrix pencil

$$\lambda E - (A - BR^{-1}(B^\mathsf{T}X_*E + S^\mathsf{T}))$$

lie in the open left complex half-plane.

In the case of dense coefficient matrices of small dimension n, a variety of different approaches has been established for the numerical solution of (1). Direct methods can be used to construct the solution via an eigenvalue decomposition of the underlying Hamiltonian or even matrix pencils [1, 5, 43]. On the other hand, iterative approaches such as the matrix sign function iteration and structure-preserving doubling avoid the eigendecomposition and aim directly for the computation of the eigenspaces of interest [14, 29, 37, 53]. Other iterative approaches construct sequences of matrices that converge to the stabilizing solution [38, 42, 59].

With the problem dimension n increasing, the task of solving (1) becomes more complicated. Even if in those cases A and E typically become sparse, the stabilizing solution X_* of (1) must be expected to be densely populated. Thus, the demands on computational resources such as time and memory become infeasible when computing X_* via classical approaches for $n \in \mathcal{O}(10^5)$ and larger. Under the assumption that the dimensions of the factored coefficients in (1) are significantly smaller than the solution dimension, i.e., $p, m \ll n$, new iterative approaches for the solution of (1) have been developed for some particular realizations. The key ingredient in all instances is the use of low-rank factorized approximations of the solution X_*, typically given as $Z_*Z_*^\mathsf{T} \approx X_*$, where $Z_* \in \mathbb{R}^{n\times \ell}$ and $\ell \ll n$. This is justified by a fast singular value decay of the solutions [10, 62].

For the special case that $S = 0$, Q is symmetric positive semi-definite and R is symmetric positive definite, a variety of new approaches has been developed in recent years. Methods like the Newton and Newton-Kleinman iterations have been extended [17, 21] employing yet another low-rank solver such as the low-rank alternating direction implicit (LR-ADI) method [19–21, 45] for the Lyapunov equations occurring in every Newton step. Projection-based methods construct approximating subspaces to project the coefficients of (1) onto smaller dimension and then solve small-scale Riccati equations with classical dense approaches [35, 60]. The Riccati alternating direction implicit (RADI) method [11, 28] and the incremental low-rank subspace iteration (ILRSI) [46] are among the most successful low-rank solvers for this variant of the Riccati equation. We refer the reader to [12, 23, 39] for general overviews and numerical comparisons of these methods.

In other instances of (1), the amount of established methods decreases significantly. In the case of Q symmetric positive semi-definite and R symmetric negative definite, only extensions of the Newton and Newton-Kleinman iteration have been proposed for large-scale sparse systems [25]. Recently, a new low-rank method has been developed in [16] that allows to compute the solution to (1) with indefinite R and Q symmetric positive semi-definite matrices. Under the assumption that the stabilizing solution X_* is symmetric positive semi-definite, this new low-rank method iteratively approximates X_* via accumulating solutions to classical Riccati equations with positive definite R terms.

In this work, we are lifting all restrictions and investigate the numerical approximation of the stabilizing solution to the general CARE (1). Therefore, we focus on the Newton-Kleinman method [38] and extend this approach to the case of large-scale sparse

coefficient matrices by utilizing an indefinite symmetric low-rank factorization of the stabilizing solution. We show that this new approach generalizes existing methods and provide a theoretical background for several of the practically occurring scenarios. The theoretical analysis is supported by multiple numerical experiments.

Throughout this paper, A^T denotes the transposed of the matrix A. Also, we denote symmetric positive (semi-)definite matrices A by $A > 0$ ($A \geq 0$) and we write $A > B$ ($A \geq B$) if $A - B$ is symmetric positive (semi-)definite. Similarly, we use $A < 0$ ($A \leq 0$) to denote symmetric negative (semi-)definite matrices and write $A < B$ ($A \leq B$) if $A - B$ is symmetric negative (semi-)definite. Moreover, $\langle ., . \rangle$ denotes the Frobenius inner product, i.e., $\langle A, B \rangle = \operatorname{tr}\left(A^\mathsf{T} B\right)$ for real matrices A and B of compatible dimensions. By I_n we denote the n-dimensional identity matrix.

The remainder of this paper is organized as follows: In Section 2, we provide an overview about different realizations of the continuous-time algebraic Riccati equation from the literature with their motivational background and how they fit into the presented general formulation (1). In Section 3, we derive the Newton-Kleinman formulation for (1) based on which we extend the approach to the large-scale sparse setting. Afterwards, we provide a theoretical analysis of the convergence behavior, formulas for an exact line search procedure in the Newton iteration and an extension of the method to non-invertible E matrices. Numerical experiments to support the theoretical discussions of this paper are conducted in Section 4. The paper is concluded in Section 5.

2 Example equations from the literature

Several realizations of CAREs are displayed throughout the literature. The form (1) we consider in this work appears to be the most general formulation of the CARE with factorized terms that allow for low-rank approximations in the large-scale sparse setting. Some of the most prominent realizations are outlined in the following. These will also serve as examples in the numerical experiments in Section 4.

2.1 Linear-quadratic regulator problems

First, we may consider the CARE formulation given in (1). With the additional assumptions that $Q \geq 0$ and $R > 0$, this realization can be found in optimal control for the construction of optimal state-feedback regulators [2, 47, 61]. The corresponding optimization problem is given by

$$\min_{u \text{ stab.}} \int_0^\infty y(t)^\mathsf{T} Q y(t) + x(t)^\mathsf{T} S u(t) + u(t)^\mathsf{T} R u(t) \,\mathrm{d}\, t \tag{2a}$$

subject to

$$\begin{aligned} E\dot{x}(t) &= Ax(t) + Bu(t), \\ y(t) &= Cx(t). \end{aligned} \tag{2b}$$

The task is to find a controller u that solves the optimization problem Eq. (2a) while stabilizing the corresponding dynamical system Eq. (2b). Assume that a stabilizing solution X_* to (1) exists, then the solution to Eq. (2) is given by $u(t) = K_* x(t)$, where the feedback matrix is given by $K_* = R^{-1}(B^\mathsf{T} X_* E + S^\mathsf{T})$. If the matrix pencil $\lambda E - A$ is stabilizable with respect to B and observable with respect to C, then a sufficient condition for the

existence of the stabilizing solution X_* is that

$$\begin{bmatrix} C^\mathsf{T}QC & S \\ S^\mathsf{T} & R \end{bmatrix} \geq 0$$

holds; see [40]. Note that under the assumptions above, the stabilizing solution X_* is known to be positive semi-definite.

2.2 Linear-quadratic-Gaussian control and unstable model order reduction

A different realization of (1) relates to the construction of optimal controllers and model order reduction of unstable dynamical systems. Consider the modified optimal regulator problem

$$\min_{u \text{ stab.}} \int_0^\infty y(t)^\mathsf{T}\widetilde{Q}y(t) + u(t)^\mathsf{T}\widetilde{R}u(t)\,\mathrm{d}\,t$$

subject to

$$\begin{aligned} E\dot{x}(t) &= Ax(t) + Bu(t), \\ y(t) &= Cx(t) + Du(t), \end{aligned}$$

with the feed-through matrix $D \in \mathbb{R}^{p\times m}$, $\widetilde{Q} \geq 0$ and $\widetilde{R} > 0$. The corresponding CARE that provides the optimal stabilizing control is given by

$$A^\mathsf{T}XE + E^\mathsf{T}XA + C^\mathsf{T}\widetilde{Q}C - \left(B^\mathsf{T}XE + D^\mathsf{T}C\right)^\mathsf{T}\left(\widetilde{R} + D^\mathsf{T}D\right)^{-1}\left(B^\mathsf{T}XE + D^\mathsf{T}C\right) = 0. \quad (3)$$

The equation (3) can be rewritten as (1) by setting

$$Q = \widetilde{Q}, \quad R = \widetilde{R} + D^\mathsf{T}D, \quad S = D^\mathsf{T}C.$$

The very same equation (3) can also be found in the design of optimal linear-quadratic-Gaussian (LQG) controllers and in the LQG balanced truncation method that is used for the computation of reduced-order dynamical systems with unstable dynamics [26, 36].

2.3 $\mathcal{H}_\infty$ control and robust model order reduction

Another realization related to controller design and model order reduction comes in the form of the $\mathcal{H}_\infty$-Riccati equation

$$A^\mathsf{T}XE + E^\mathsf{T}XA + C^\mathsf{T}\widetilde{Q}C - E^\mathsf{T}X\left(B_2\widetilde{R}^{-1}B_2^\mathsf{T} - \frac{1}{\gamma^2}B_1B_1^\mathsf{T}\right)XE = 0; \quad (4)$$

see [15, 16, 50]. This equation is typically associated with dynamical systems of the form

$$\begin{aligned} E\dot{x}(t) &= Ax(t) + B_1w(t) + B_2u(t), \\ y(t) &= Cx(t), \end{aligned}$$

where $B_1 \in \mathbb{R}^{n\times n_1}$ models the influence of external disturbances on the control problem and $B_2 \in \mathbb{R}^{n\times m_2}$ are the actual control inputs. The matrices $\widetilde{R} > 0$ and $\widetilde{Q} \geq 0$ are weighting matrices from the associated optimal control problem similar to Eq. (2a), and $\gamma > 0$ is the robustness margin that is achieved by the constructed regulator/controller. Equations of the form (4) can be rewritten into (1) via

$$B = \begin{bmatrix} B_1 & B_2 \end{bmatrix}, \quad Q = \widetilde{Q}, \quad R = \begin{bmatrix} -\gamma^2 I_{m_1} & 0 \\ 0 & \widetilde{R} \end{bmatrix}, \quad S = 0.$$

In this case, the quadratic weighting term R in (1) is indefinite.

2.4 Passivity, contractivity and spectral factorizations

As last examples, we would like to mention two equations that are related to dynamical system properties such as contractivity and passivity as well as spectral factorizations of rational functions [26, 31, 48, 52]. The so-called bounded-real Riccati equation is given as

$$A^\mathsf{T}XE+E^\mathsf{T}XA+C^\mathsf{T}C+\left(B^\mathsf{T}XE+D^\mathsf{T}C\right)^\mathsf{T}\left(\gamma^2 I_m-D^\mathsf{T}D\right)^{-1}\left(B^\mathsf{T}XE+D^\mathsf{T}C\right)=0, \quad (5)$$

with $D\in\mathbb{R}^{p\times m}$ and $\gamma>\|H\|_{\mathcal{H}_\infty}$, where $\|.\|_{\mathcal{H}_\infty}$ denotes the $\mathcal{H}_\infty$ Hardy norm and $H(s)=C(sE-A)^{-1}B+D$ is a rational function in the complex variable $s\in\mathbb{C}$. On the other hand, the positive-real Riccati equation reads

$$A^\mathsf{T}XE+E^\mathsf{T}XA+\left(B^\mathsf{T}XE-C\right)^\mathsf{T}\left(D^\mathsf{T}+D\right)^{-1}\left(B^\mathsf{T}XE-C\right)=0, \quad (6)$$

where the dimensions satisfy $m=p$. Equation (5) can be rewritten as (1) by choosing

$$Q=I_p,\quad R=-\left(\gamma^2-D^\mathsf{T}D\right),\quad S=C^\mathsf{T}D,$$

and equation (6) can be reformulated using

$$Q=0,\quad R=-\left(D^\mathsf{T}+D\right),\quad S=C^\mathsf{T}D.$$

With the assumptions above, the R matrix is symmetric negative definite in both cases, while Q is either symmetric positive definite or 0.

3 Low-rank inexact Newton-Kleinman iteration with line search

In this section, we derive the low-rank Newton-Kleinman iteration and provide the formulas for inexact steps, a line search approach and projected Riccati equations.

3.1 Derivation of the low-rank Newton-Kleinman scheme

Solving the CARE (1) is a root-finding problem with a nonlinear matrix-valued equation and solution. Therefore, Newton's method is a valid approach to compute a solution to the problem [59], and it has been shown in many cases that the computed solution is the desired stabilizing one. The basic method can be derived by considering the Fréchet derivative of the Riccati operator

$$\mathcal{R}(X)=A^\mathsf{T}XE+E^\mathsf{T}XA+C^\mathsf{T}QC-\left(B^\mathsf{T}XE+S^\mathsf{T}\right)^\mathsf{T}R^{-1}\left(B^\mathsf{T}XE+S^\mathsf{T}\right), \quad (7)$$

with respect to the unknown X. The first Fréchet derivative of (7) with respect to X and evaluated in N is given by

$$\mathcal{R}'(X)(N)=\left(A-BR^{-1}(B^\mathsf{T}XE+S^\mathsf{T})\right)^\mathsf{T}NE+E^\mathsf{T}N\left(A-BR^{-1}(B^\mathsf{T}XE+S^\mathsf{T})\right)^\mathsf{T},$$

and the second Fréchet derivative with respect to X evaluated in N_1 and N_2 is independent of X and can be written as

$$\mathcal{R}''(X)(N_1,N_2)=-E^\mathsf{T}N_1BR^{-1}B^\mathsf{T}N_2E-E^\mathsf{T}N_2BR^{-1}B^\mathsf{T}N_1E.$$

As outlined in [17], the classical Newton approach is usually undesired in the case of large-scale sparse systems when compared to the reformulation given by the Newton-Kleinman scheme [38]. Either method is based on the solution of a Lyapunov equation in every iteration step. However, while the classical Newton method computes an update to the current iterate of the form $X_{k+1} = X_k + N_k$, where N_k is given as the solution to a Lyapunov equation, the Newton-Kleinman method computes X_{k+1} directly as the solution of the Lyapunov equation that is given by

$$\mathcal{R}'(X_k)(X_{k+1}) = \mathcal{R}'(X_k)(X_k) - \mathcal{R}(X_k);$$

see [17]. Following this idea, the authors of [4,5] developed a Newton-Kleinman approach for (1) solving the following Lyapunov equation

$$A_k^\mathsf{T} X_{k+1} E + E^\mathsf{T} X_{k+1} A_k + C^\mathsf{T} Q C + K_k^\mathsf{T} R K_k - S K_k - (S K_k)^\mathsf{T} = 0 \tag{8}$$

in every iteration step, with $A_k = A - BK_k$ and $K_k = R^{-1}(B^\mathsf{T} X_k E + S^\mathsf{T})$, and starting with some initial stabilizing feedback K_0. This K_0 is chosen such that all eigenvalues of $\lambda E - (A - BK_0)$ lie in the open left half-plane.

To extend the scheme (8) to the large-scale sparse setting, we must first observe that the part of the equation that does not contain the current unknown X_{k+1}, in other words its right-hand side, is an indefinite symmetric matrix. To utilize this form of the right-hand side, similar to the argumentation in [41], we propose to approximate the solution matrix to (1) by a symmetric indefinite low-rank factorization of the form

$$LDL^\mathsf{T} \approx X, \tag{9}$$

where $L \in \mathbb{R}^{n\times\ell}$ and $D \in \mathbb{R}^{\ell\times\ell}$ is symmetric. By the low-rank structure of the right-hand side coefficient matrices of (1), as well as its quadratic terms (since $m, p \ll n$), we expect the solution to have numerically a low rank such that $\ell \ll n$ holds [10,62]. Rewriting the right-hand side of (8) into the same shape as (9) yields

$$\begin{aligned} &C^\mathsf{T} Q C + K_k^\mathsf{T} R K_k - S K_k - (S K_k)^\mathsf{T} \\ &= \begin{bmatrix} C^\mathsf{T} & K_k^\mathsf{T} & S \end{bmatrix} \begin{bmatrix} Q & 0 & 0 \\ 0 & R & -I_m \\ 0 & -I_m & 0 \end{bmatrix} \begin{bmatrix} C \\ K_k \\ S^\mathsf{T} \end{bmatrix}, \end{aligned} \tag{10}$$

where the center matrix has the dimension $2m + p$. It is possible to avoid the switching term (the two negative identities) in the lower right corner of the center matrix in (10) by making the following reformulations:

$$\begin{aligned} &C^\mathsf{T} Q C + K_k^\mathsf{T} R K_k - S K_k - (S K_k)^\mathsf{T} \\ &= C^\mathsf{T} Q C + K_k^\mathsf{T} R K_k - S K_k - (S K_k)^\mathsf{T} - \underbrace{S R^{-1} S^\mathsf{T} + S R^{-1} S^\mathsf{T}}_{=0} \\ &= C^\mathsf{T} Q C - S R^{-1} S^\mathsf{T} + (K_k - R^{-1} S^\mathsf{T})^\mathsf{T} R (K_k - R^{-1} S^\mathsf{T}) \\ &= \begin{bmatrix} C^\mathsf{T} & S & (K_k - R^{-1} S^\mathsf{T})^\mathsf{T} \end{bmatrix} \begin{bmatrix} Q & 0 & 0 \\ 0 & -R & 0 \\ 0 & 0 & R \end{bmatrix} \begin{bmatrix} C \\ S^\mathsf{T} \\ K_k - R^{-1} S^\mathsf{T} \end{bmatrix}. \end{aligned} \tag{11}$$

The reformulation in (11) also has $2m + p$ as inner dimension of the factors and features a block diagonal center matrix, which we believe to be advantageous in the implementation. Plugging (11) into (8) yields the final LDL^T-factorized Lyapunov equation that we employ in our new Newton-Kleinman iteration. The resulting method is summarized in Algorithm 1. Lyapunov equations such as in Line 4 of Algorithm 1 can be efficiently solved, for example, via the LDL^T-factorized low-rank ADI method in [41].

Algorithm 1: LDL^T-factorized low-rank Newton-Kleinman iteration.

Input: Matrices A, B, S, C, Q, R, E from (1), stabilizing feedback K_0 such that $sE - (A - BK_0)$ is Hurwitz.

Output: Approximation $L_k D_k L_k^\mathsf{T} \approx X_*$ to the stabilizing solution of (1).

1 Initialize $k = 0$,

$$V^\mathsf{T} = R^{-1}S^\mathsf{T} \quad \text{and} \quad T = \begin{bmatrix} Q & 0 & 0 \\ 0 & -R & 0 \\ 0 & 0 & R \end{bmatrix}.$$

2 **while** *not converged* **do**

3 Construct the residual right-hand side

$$W_k = \begin{bmatrix} C \\ V^\mathsf{T} \\ K_k - V^\mathsf{T} \end{bmatrix}.$$

4 Solve the Lyapunov equation

$$A_k^\mathsf{T} X_{k+1} E + E^\mathsf{T} X_{k+1} A_k + W_k^\mathsf{T} T W_k = 0,$$

for $L_{k+1} D_{k+1} L_{k+1}^\mathsf{T} \approx X_{k+1}$ and where $A_k = A - BK_k$.

5 Update the feedback matrix $K_{k+1} = R^{-1}\left(B^\mathsf{T}(L_{k+1}D_{k+1}L_{k+1}^\mathsf{T})E + S^\mathsf{T}\right)$.

6 Increment $k \leftarrow k + 1$.

7 **end**

3.2 An equivalent reformulation via low-rank updates

The efficient handling of the sparse plus low-rank operator $A_k = A - BK_k$ in the Lyapunov equation in Line 4 of Algorithm 1 is essential for computing its solution in the large-scale sparse case. Typically, linear systems of equations with A_k need to be solved, which can be effectively implemented using the Sherman-Morrison-Woodbury matrix inversion formula or the augmented matrix approach; see, for example, [34]. Since the handling of such operators is already implemented in most software for the solution of matrix equations such as the M-M.E.S.S. library [18], we may use a reformulation of (1) to hide the S inside the other matrices. First, we observe that by multiplying out the terms in (1), we obtain the equivalent CARE

$$\begin{aligned}\left(A - BR^{-1}S^\mathsf{T}\right)^\mathsf{T} XE + E^\mathsf{T} X \left(A - BR^{-1}S^\mathsf{T}\right) + \left(C^\mathsf{T}QC - SR^{-1}S^\mathsf{T}\right) \\ - E^\mathsf{T} X B R^{-1} B^\mathsf{T} X E = 0.\end{aligned} \tag{12}$$

After some renaming of the terms in (12), we obtain

$$\widehat{A}^\mathsf{T} X E + E^\mathsf{T} X \widehat{A} + \widehat{C}^\mathsf{T} \widehat{Q} \widehat{C} - E^\mathsf{T} X B R^{-1} B^\mathsf{T} X E = 0, \tag{13}$$

where

$$\widehat{A} = A + UV^\mathsf{T}, \quad U = -B, \quad V = SR^{-\mathsf{T}}, \quad \widehat{C} = \begin{bmatrix} C \\ S^\mathsf{T} \end{bmatrix}, \quad \widehat{Q} = \begin{bmatrix} Q & 0 \\ 0 & -R^{-1} \end{bmatrix}.$$

Running Algorithm 1 for the renamed matrices $\widehat{A}$, B, $\widehat{S} = 0$, $\widehat{C}$, $\widehat{Q}$, R and E will yield exactly the same iterates computed in every step, while hiding the original S term in $\widehat{A}$

and $\widehat{C}$. Note here that all the corresponding stabilizing feedbacks $\widehat{K}_k$ are changed such that $\lambda E - (\widehat{A} - B\widehat{K}_k)$ is stabilized rather than $\lambda E - (A - BK_k)$. In particular, the initial stabilizing feedback must be chosen correctly. On the other hand, the final stabilizing feedback $\widehat{K}_{k_{\max}}$, corresponding to the final iterate $\widehat{X}_{k_{\max}}$, can easily be modified to stabilize the true associated matrix pencil $\lambda E - A$ via

$$K_{k_{\max}} = \widehat{K}_{k_{\max}} + V^\mathsf{T}.$$

While the two formulations (1) and (13) are equivalent, employing the Newton-Kleinman method for (13) rather than (1) is expected to be mildly more expensive in the general case because the dimension of the right-hand side stays unchanged while the dimension of the low-rank updates in A_k are increased by m columns. On the other hand, considering the example equations from Section 2, we can also see a reduction in computational costs using (13). In Equations (3), (5), and (6),the S term is a multiplication of the constant term C with some appropriately sized matrix D. This fact allows us some additional dimension reduction of the constant term in (13). As example, consider the case of the LQG CARE in (3): The constant term in (13) can be written as

$$\widehat{C}^\mathsf{T}\widehat{Q}\widehat{C} = C^\mathsf{T}\widetilde{Q}C - C^\mathsf{T}D(\widetilde{R} + D^\mathsf{T}D)^{-1}D^\mathsf{T}C = C^\mathsf{T}\underbrace{\left(\widetilde{Q} - D(\widetilde{R} + D^\mathsf{T}D)^{-1}D^\mathsf{T}\right)}_{=\check{Q}}C = C^\mathsf{T}\check{Q}C.$$

In such cases, the size of the constant term in (13) can be reduced from $2m+p$ to $m+p$, which improves the performance of large-scale sparse solvers that build the solution using the constant right-hand side. We have implemented this version of Algorithm 1 for the reformulated CARE (13) in the M-M.E.S.S. library [18] for our numerical experiments due to the easy integration into the existing framework of CAREs of the form (13).

3.3 Convergence results

In the following theorem, we collect the convergence results for two distinct cases of (1), depending on the definiteness of the quadratic term. The results are formulated for the exact iterates X_k of the Newton-Kleinman iteration in Algorithm 1 rather than the low-rank approximations $L_kD_kL_k^\mathsf{T}$ since the introduced disturbances may render the results wrong. However, for accurate enough approximations, the results remain true in practice.

Theorem 1. *Assume (1) has a unique stabilizing solution X_*, let K_0 be a feedback matrix such that the eigenvalues of $\lambda E - (A - BK_0)$ lie in the open left complex half-plane and let either $R > 0$ or $R < 0$ be true. Then, for the exact iterates $X_k = L_kD_kL_k^\mathsf{T}$ in Algorithm 1, it holds that*

(i) the closed-loop pencils $\lambda E - A_k$ with $A_k = A - BK_k$ are stable for all $k \geq 0$,

(ii) $\lim_{k\to\infty} X_k = X_*$ and $\lim_{k\to\infty} \mathcal{R}(X_k) = 0$,

(iii) the iterates X_k converge globally and quadratic to X_*,

(iv) if $R > 0$, then

$$X_1 \geq \cdots \geq X_k \geq X_{k+1} \geq \cdots \geq X_*,$$

and if $R < 0$, then

$$X_1 \leq \cdots \leq X_k \leq X_{k+1} \leq \cdots \leq X_*.$$

Proof. The results have been proven for the case $R > 0$ in [4]. In the case of $R < 0$, we may use the convergence results from [25, Thm. 3.2], which are based on earlier results from [9,65]. Therefore, we consider the equivalent reformulation of the CARE into classical form (13). Since $R < 0$, it holds that $-R > 0$ and therefore, we may write (13) as

$$\widehat{A}^\mathsf{T} X E + E^\mathsf{T} X \widehat{A} + \widehat{C}^\mathsf{T} \widehat{Q} \widehat{C} + E^\mathsf{T} X B \widehat{R}^{-1} B^\mathsf{T} X E = 0,$$

where $\widehat{R} > 0$. With $\widehat{K}_0 = K_0 - R^{-1} S^\mathsf{T}$, the matrix pencil $\lambda E - (\widehat{A} - B\widehat{K}_0)$ is stable and since $\widehat{R} > 0$, we have that $B\widehat{R}^{-1}B^\mathsf{T} \geq 0$. Thus, the assumptions of [25, Thm. 3.2] are satisfied, which proofs the results of this theorem. □

Theorem 1 shows that the general convergence behavior of Algorithm 1 is only determined by the definiteness of the quadratic term. The other terms C, Q and S only affect the definiteness of the stabilizing solution X_*, to which the method converges. Beyond the convergence theory, the reformulations made in Section 3.1 and in the proof of Theorem 1 show that in exact arithmetic, the proposed Algorithm 1 provides the exact same iterates as the Newton-Kleinman methods developed in [4, 25].

The techniques used to show all the convergence results in the proofs in [4, 25] are based on the main observation that the difference of two consecutive steps in the Newton-Kleinman scheme is given as the unique solution of the Lyapunov equation

$$X_k - X_{k+1} = \int_0^\infty \left(e^{(AE^{-1}-BK_k)t}\right)^\mathsf{T} (K_k - K_{k+1})^\mathsf{T} R (K_k - K_k) e^{(AE^{-1}-BK_{k+1})t} \,\mathrm{d}\, t.$$

The definiteness of the difference $X_k - X_{k+1}$ depends thereby on the definiteness of the R matrix resulting in the monotonic convergence behavior described in Theorem 1. In the case of indefinite R, this monotonic behavior is likely to be lost as we will demonstrate later in Section 4. However, we were not able to construct a case for which a stabilizing solution X_* exists while the Newton-Kleinman method (Algorithm 1) diverges or converges to the wrong solution. In fact, Algorithm 1 only diverged in experiments in which there was no stabilizing X_*. This indicates that Algorithm 1 may always converge to the correct solution; however, most of the convergence results in Theorem 1 will not be true anymore for the case of indefinite R.

A different approach that provides a convergence theory for the case of indefinite R is the Riccati iteration [16, 42]. This method iterates on Riccati equations with positive semi-definite quadratic terms for which the convergence theory in Theorem 1 holds. Under the additional assumption that $Q \geq 0$ and $X_* \geq 0$ hold, the Riccati iteration constructs iterates that monotonically converge towards X_* as

$$X_0^{\mathrm{RI}} \leq \cdots \leq X_k^{\mathrm{RI}} \leq X_{k+1}^{\mathrm{RI}} \leq X_*.$$

Since each of these iterates is computed via a CARE solver for $R > 0$, this overall iteration scheme can be interpreted as splitting the two opposing convergence behaviors in Theorem 1 into an inner and an outer iteration. Similar to the Newton methods, the Riccati iteration provides global quadratic convergence. The main difference to the results in Theorem 1 is that the closed-loop matrix pencils constructed in the outer loop of the iteration are not guaranteed to be stable such that additional stabilization might be needed to employ an inner CARE solver in the large-scale sparse case.

3.4 Inexact Newton with exact line search

Newton's method with exact line search has first been discussed for dense generalized algebraic Riccati equations in [13]. Based on this work, Weichelt et al. [17,66] formulated an inexact low-rank Newton-ADI method with exact line search, focusing on the representation of solutions in the form $X \approx ZZ^\mathsf{T}$. Since, in this work, we are pointing out advantages of the $X \approx LDL^\mathsf{T}$ representation, we provide the required formulas in this context and show that they can also be evaluated at low cost.

To this end, we may call the k-th and $(k+1)$-st classic Newton-Kleinman iterates X_k and X_{k+1} and note that they are connected via the step matrix N_k, since $X_{k+1} = X_k + N_k$. Further, we denote the $(k+1)$-st iterate after line search with the resulting step size ξ_k as $X_{k+1,\xi_k} = X_k + \xi_k N_k$. In [66, Chap. 6], using earlier results from [8,13] for the dense case, the author shows that the dependence on the step size ξ_k of the squared Riccati residual norm, in the k-th Newton step, forms a quartic polynomial

$$\begin{aligned} f_{\mathcal{R},k}(\xi) &= \|\mathcal{R}(X_{k+1,\xi_k})\|_F^2 \\ &= (1-\xi)^2 v_1^{(k)} + \xi^2 v_2^{(k)} + \xi^4 v_3^{(k)} + 2\xi(1-\xi) v_4^{(k)} - 2\xi^2(1-\xi) v_5^{(k)} - 2\xi^3 v_6^{(k)}. \end{aligned} \tag{14}$$

The coefficients are expressed in terms of the norms of the Riccati residual and its derivatives evaluated in the above quantities, and expressed in low-rank form. In the context of the equations investigated here, these terms become

$$\begin{aligned} v_1^{(k)} &= \|\mathcal{R}(X_k)\|_F^2 = \operatorname{tr}\Big((U_k P_k U_k^\mathsf{T})^2\Big) = \operatorname{tr}\Big((U_k^\mathsf{T} U_k P_k)^2\Big), \\ v_{2^{(k)}} &= \|\mathcal{L}(X_{k+1})\|_F^2 = \operatorname{tr}\Big((F_{k+1} G_{k+1} F_{k+1}^\mathsf{T})^2\Big) = \operatorname{tr}\Big((F_{k+1}^\mathsf{T} F_{k+1} G_{k+1})^2\Big), \\ v_3^{(k)} &= \left\|\frac{1}{2}\mathcal{R}''(X_{k+1})(N_k, N_k)\right\|_F^2 = \left\|E^\mathsf{T} N_k B R^{-1} B^\mathsf{T} N_k E\right\|_F^2 \\ &= \operatorname{tr}\Big((\Delta K_{k+1} R \Delta K_{k+1}^\mathsf{T})^2\Big) = \operatorname{tr}\Big((\Delta K_{k+1}^\mathsf{T} \Delta K_{k+1} R)^2\Big), \\ v_4^{(k)} &= \langle \mathcal{R}(X_k), \mathcal{L}(X_{k+1})\rangle = \operatorname{tr}\Big(U_k P_k U_k^\mathsf{T} F_{k+1} G_{k+1} F_{k+1}^\mathsf{T}\Big) \\ &= \operatorname{tr}\Big(F_{k+1}^\mathsf{T} U_k P_k U_k^\mathsf{T} F_{k+1} G_{k+1}\Big), \\ v_5^{(k)} &= \langle \mathcal{R}(X_k), \mathcal{R}''(X_{k+1})(N_k, N_k)\rangle = \operatorname{tr}\Big(U_k P_k U_k^\mathsf{T} \Delta K_{k+1} R \Delta K_{k+1}^\mathsf{T}\Big) \\ &= \operatorname{tr}\Big(\Delta K_{k+1}^\mathsf{T} U_k P_k U_k^\mathsf{T} \Delta K_{k+1} R\Big), \\ v_6^{(k)} &= \langle \mathcal{L}(X_{k+1}), \mathcal{R}''(X_{k+1})(N_k, N_k)\rangle = \operatorname{tr}\Big(F_{k+1} G_{k+1} F_{k+1}^\mathsf{T} \Delta K_{k+1} R \Delta K_{k+1}^\mathsf{T}\Big) \\ &= \operatorname{tr}\Big(\Delta K_{k+1}^\mathsf{T} F_{k+1} G_{k+1} F_{k+1}^\mathsf{T} \Delta K_{k+1} R\Big). \end{aligned}$$

This is employing the Fréchet derivatives from Section 3.1, and we use that $N_k = X_{k+1} - X_k$. Consequently, $R^{-1}B^\mathsf{T} N_k E = K_{k+1} - K_k = \Delta K_{k+1}$ holds. Further, we have defined $U_k = \begin{bmatrix} F_k & \Delta K_k \end{bmatrix}$ and

$$P_k = \begin{bmatrix} G_k & 0 \\ 0 & -R \end{bmatrix},$$

to express the Riccati residual in the k-th Newton step as $\mathcal{R}(X_k) = U_k P_k U_k^\mathsf{T}$, extending [66, Eqn. (6.33b)] to non-trivial center matrices. Here, $\mathcal{L}(X_k) = F_k G_k F_k^\mathsf{T}$ denotes the final Lyapunov residual of the k-th Newton step equations. Observe how the cyclic permutation property of the trace allows turning all arguments into the final small dense matrices.

Sorting terms by the powers of ξ in (14) leads to five coefficients of the fourth order polynomial in standard form. The minimizing argument ξ_k is computed from the zeros of $\frac{d}{d\xi} f_{\mathcal{R},k}$. Then, the actual step size is

$$\xi_k = \operatorname{argmin}_{\xi \in \Lambda(\widetilde{A},\widetilde{E}) \cap (0,2]} f_{\mathcal{R},k}(\xi)$$

for the 3×3 generalized eigenvalue problem for the matrix pencil

$$(\widetilde{A}, \widetilde{E}) = \left(\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ a_1 & a_2 & a_3 \end{bmatrix}, \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & a_4 \end{bmatrix} \right),$$

where $a = \frac{1}{\|\widehat{a}\|}\widehat{a}$ and $\widehat{a} \in \mathbb{R}^4$ the with components

$$\begin{aligned} \widehat{a}_1 &= 2(v_4^{(k)} - v_1^{(k)}), \\ \widehat{a}_2 &= 2(v_1^{(k)} + v_2^{(k)} - 2(v_4^{(k)} + v_5^{(k)})), \\ \widehat{a}_3 &= 6(v_5^{(k)} - v_6^{(k)}), \\ \widehat{a}_4 &= 4v_3^{(k)}. \end{aligned}$$

These last steps are exactly identical to the presentation in [17,66]. Note that additional care is necessary when multiple consecutive iteration steps use line search since the Riccati residual factors grow with the number of consecutive line searches and also ΔK_{k+1} appends a new block of columns, equal to its own size, with each additional line search iteration. See the discussion in [17] after Equation (5.4) for details. In our context, the corresponding center matrix P_k then block diagonally accumulates the corresponding center matrices rather than simple signed identities. Note further, that alternatively an Armijo line search can be used, but then the step size is limited to the interval $(0, 1]$.

While the line search can help reduce the total number of Newton steps required, the cost of the single steps can be reduced by an inexact Newton approach. The above considerations are ensuring the *sufficient decrease condition*

$$\|\mathcal{R}(X_{k+1,\xi_k})\|_F < (1 - \xi_k\beta) \|\mathcal{R}(X_k)\|_F ,$$

for a certain positive safety parameter β. The inexact Newton acceleration, on the other hand, is controlled by

$$\|\mathcal{L}(X_{k+1})\|_F < \tau_k \|\mathcal{R}(X_k)\|_F ,$$

for an appropriate forcing sequence $(\tau_k)_{k\in\mathbb{N}}$. In [66], the author suggests $\tau_k = \frac{1}{k^3+1}$ to achieve super-linear convergence and $\tau_k = \min\{0.1, 0.9 \|\mathcal{R}(X_k)\|_F\}$ to preserve quadratic convergence; see [66, Table 6.1]. In general any $\tau_k \to 0$ for $k \to \infty$ would guarantee super-linear convergence, while $\tau_k \in \mathcal{O}(\mathcal{R}(X_k))$ ensures quadratic convergence. However, note that while the general low-rank inexact Newton framework builds on the theory in [32], certain definiteness conditions required in their central theorem can not be guaranteed in general in the low-rank case such that the low-rank inexact Newton-Kleinman method may break down. Implementations need to check this and potentially restart the method without inexactness.

3.5 Non-invertible E matrices and projected Riccati equations

The examples for CAREs we have considered in Section 2 are all based on or associated with linear dynamical systems. A regularly occurring situation is that these dynamical

systems are described by differential-algebraic rather than ordinary differential equations, in which case the E matrix in (1) becomes non-invertible. Assume that the matrix pencil $\lambda E - A$ is regular, i.e., there exists a $\lambda \in \mathbb{C}$ such that $\det(\lambda E - A) \neq 0$. Then, one typically considers the solution of (1) over the subspace of finite eigenvalues of $\lambda E - A$ via the projected Riccati equation

$$\begin{aligned} A^\mathsf{T} X E + E^\mathsf{T} X A + \mathcal{P}_r^\mathsf{T} C^\mathsf{T} Q C \mathcal{P}_r - \left(B^\mathsf{T} X E + S^\mathsf{T} \mathcal{P}_r\right)^\mathsf{T} R^{-1} \left(B^\mathsf{T} X E + S^\mathsf{T} \mathcal{P}_r\right) = 0, \\ \mathcal{P}_\ell^\mathsf{T} X \mathcal{P}_\ell = X, \end{aligned} \tag{15}$$

where $\mathcal{P}_r$ and $\mathcal{P}_\ell$ are the right and left projectors onto the subspace of finite eigenvalues of $\lambda E - A$. In general, these are given as spectral projectors via the Weierstrass canonical form of $\lambda E - A$; see, for example, [25]. While the necessary computations to obtain these spectral projectors are typically undesired in the large-scale sparse case, for several practically occurring matrix structures, the projectors have been formulated explicitly in terms of parts the coefficient matrices [25, 63].

In practice, a more efficient approach than explicitly forming $\mathcal{P}_r$ and $\mathcal{P}_\ell$ is the implicit application of equivalent structural projectors. In this case, the stabilizing solution of (15) is directly computed on the correct lower dimensional subspace. Similar to the use of the spectral projectors, the implicit projection can, in practice, only be realized for certain matrix structures, for which the projectors onto the correct subspaces and truncation of the coefficient matrices are known by construction; see, for example, [6, 33, 57]. In all cases, it needs to be noted that the steps in Algorithm 1 do not change for (15). The case of non-invertible E matrices can typically be implemented by simply modifying the matrix-matrix and matrix-vector operations needed in Algorithm 1 to work on the correct subspaces.

4 Numerical experiments

The experiments reported here have been executed on a machine with an AMD Ryzen Threadripper PRO 5975WX 32-Cores processor running at 4.02 GHz and equipped with 252 GB total main memory. The computer is running on Ubuntu 22.04.3 LTS and uses MATLAB 23.2.0.2365128 (R2023b). The proposed low-rank Newton-Kleinman method in Algorithm 1 has been implemented for dense equations using MORLAB version 6.0 [24, 27] and for large-scale sparse equations using M-M.E.S.S. version 3.0 [18, 56]. The resulting modified versions of these two toolboxes as well as the source code, data and results of the numerical experiments can be found at [58]. The implementations of Algorithm 1 will be incorporated into the upcoming releases of M-M.E.S.S. and MORLAB.

4.1 Experimental setup

An overview about the used example data with the computed equation setups and corresponding dimensions is shown in Table 1. The used example data are:

`aircraft` is the `AC10` data set from [44] modeling the linearized vertical-plane dynamics of an aircraft,

`msd` is a mass-spring-damper system with a holonomic constraint as described in [49],

`rail`$_{(1,6)}$ models a heat transfer problem for optimal cooling of steel profiles in two differently accurate discretizations [22, 51] using the re-implementation [54, 55],

Table 1: Overview about example data, matrix dimensions, considered equation setups and the stability properties of the matrix pencil $\lambda E - A$. The first three examples are treated as dense and the latter three as large-scale sparse.

example	n	m	m_1	m_2	p	LQG	HINF	BR	PR	stability
`aircraft`	55	5	2	3	5	✓	✓	—	—	unstable
`rail`$_{(1)}$	371	7	3	4	6	✓	✓	✓	✓	stable
`triplechain`$_{(1)}$	602	1	—	—	1	—	—	✓	✓	stable
`msd`	12 001	1	—	—	3	—	—	✓	—	stable
`triplechain`$_{(2)}$	12 002	1	—	—	1	—	—	✓	✓	stable
`rail`$_{(6)}$	317 377	7	3	4	6	✓	✓	✓	✓	stable

`triplechain`$_{(1,\ 2)}$ is the triplechain oscillator benchmark introduced in [64] with two different sets of parameters and numbers of masses.

The data set `msd` has a non-invertible E matrix and is handled via structured implicit projections as outlined in Section 3.5, following the theory in [57]. To test different scenarios of matrix pencil properties paired with different weighting terms, we have set up the different formulations of CAREs as motivated in Section 2. Further on, we denote examples for equation (3) as `LQG`, equation (4) as `HINF`, equation (5) as `BR` and equation (6) as `PR`. The modifications of the example data from the literature to fit into the described equation types can be found in the accompanying code package [58].

To compare the solutions of different computational approaches, we evaluate three types of scaled residual norms that have been used for similar purposes in the literature:

$$\begin{aligned}
\mathrm{res}_1(X) &= \frac{\|\mathcal{R}(X)\|_2}{\|\widehat{C}^\mathsf{T}\widehat{Q}\widehat{C}\|_2},\\
\mathrm{res}_2(X) &= \frac{\|\mathcal{R}(X)\|_2}{\|\widehat{A}\|_2\|E\|_2\|X\|_2 + \|BR^{-1}B^\mathsf{T}\|_2},\\
\mathrm{res}_3(X) &= \frac{\|\mathcal{R}(X)\|_2}{2\|\widehat{A}\|_2\|E\|_2\|X\|_2 + \|\widehat{C}^\mathsf{T}\widehat{Q}\widehat{C}\|_2 + \|E\|_2^2\|X\|_2^2\|BR^{-1}B^\mathsf{T}\|_2},
\end{aligned}$$

where $\mathcal{R}(.)$ is the Riccati operator from (7),

$$\widehat{C}^\mathsf{T}\widehat{Q}\widehat{C} = C^\mathsf{T}QC - SR^{-1}S^\mathsf{T} \quad \text{and} \quad \widehat{A} = A - BR^{-1}S^\mathsf{T}.$$

In the case that multiple algorithms have been used to compute the stabilizing solution to (1), we also compare the relative differences between these solutions via

$$\mathrm{reldiff}(X_1, X_2) := \frac{\|X_1 - X_2\|_2}{0.5(\|X_1\|_2 + \|X_2\|_2)}.$$

For compactness of presentation, we introduce the following notation for the different methods used in the numerical experiments:

`NEWTON` denotes the Newton-Kleinman method from Algorithm 1,

`ICARE` is the built-in function from MATLAB for the solution of (1) implementing the algorithm in [5],

Table 2: Convergence behavior of the Newton-Kleinman method (`NEWTON`) for the example (16): For each iteration step, the columns show the normalized residuals, the two eigenvalues of the current closed-loop matrix and the two eigenvalues of the difference of two consecutive iterates.

iter. step k	$\mathbf{res}_1(X_k)$	$\Lambda(A_k)$	$\Lambda(X_k - X_{k-1})$
1	5.3610e-01	−1.1049, −4.5676	—
2	3.5593e-02	−1.4100, −4.2395	3.7386e+00, −1.9830e-03
3	6.0872e-05	−1.4068, −4.2451	−5.0004e-02, 2.6109e-05
4	1.5903e-10	−1.4068, −4.2451	6.6211e-05, 1.1112e-09
5	2.1316e-14	−1.4068, −4.2451	−1.3313e-10, −1.8760e-15

`SIGN` denotes the sign function iteration method for Riccati equations as described in [14],

`RI` is the Riccati iteration for the solution of CAREs with indefinite quadratic terms; see [16, 42].

Independent of the employed algorithm and the resulting format of the computed results, e.g., factorized or unfactorized, we denote the final approximation to the stabilizing solution X_* by any of the algorithms as $X_{k_{\max}}$.

4.2 Convergence behavior for indefinite terms

Before we test the proposed method on higher dimensional data sets against other approaches, we want to investigate the convergence behavior of Algorithm 1 for the case of indefinite quadratic and constant terms. In particular the former case is not covered by any convergence theory for `NEWTON`. First, consider the CARE (1) with the following matrices

$$A = \begin{bmatrix} 2 & 1 \\ 1 & -3 \end{bmatrix}, \quad B = \begin{bmatrix} 1 & 1 \\ 0 & 2 \end{bmatrix}, \quad R = \begin{bmatrix} -1 & 0 \\ 0 & 1.5 \end{bmatrix}, \quad C = \begin{bmatrix} 1 & 1 \end{bmatrix}, \\ E = I_2, \qquad S = 0, \qquad Q = 1. \tag{16}$$

In this example, we have an unstable matrix pencil $\lambda E - A$ with one eigenvalue in the right open and one eigenvalue in the left open half-plane. The quadratic weighting term R is indefinite but the constant weighting term Q is symmetric positive definite. For the stabilizing solution it holds that $X_* > 0$ such that besides `NEWTON`, `RI` can be used in this example. Due to the instability, a stabilizing initial feedback K_0 is constructed for `NEWTON`; see [58] for details. The convergence behavior of `NEWTON` for (16) is shown in Table 2. We observe that despite the indefinite quadratic term, the iteration provides quadratic convergence and the intermediate closed-loop matrices $A_k = A - BK_k$ are all stable. However, the monotonic convergence behavior that is theoretically shown for definite R matrices is clearly not present in this example, since the eigenvalues of $X_k - X_{k-1}$ have different signs for two of the iteration steps. Also, the definiteness of $X_k - X_{k-1}$ fully changes from step 4 to 5.

As additional verification that Algorithm 1 computes the correct, stabilizing solution, we compare it to solutions obtained via `ICARE` and `RI`. The corresponding residuals are

Table 3: Residual norms for all test examples and comparison methods in Section 4.2. `NEWTON` provides as accurate or even more accurate solutions compared to the standard approach `ICARE`. `RI` only works for the first considered scenario and diverges for the second one.

example	method	$\mathbf{res_1}(\boldsymbol{X}_{k_{\max}})$	$\mathbf{res_2}(\boldsymbol{X}_{k_{\max}})$	$\mathbf{res_3}(\boldsymbol{X}_{k_{\max}})$
example (16)	`NEWTON`	9.5151e-15	2.2825e-16	8.7894e-18
	`ICARE`	3.5804e-15	8.5887e-17	3.3073e-18
	`RI`	3.1979e-12	7.6713e-14	2.9540e-15
example (17)	`NEWTON`	1.9453e-14	3.4368e-16	1.2723e-17
	`ICARE`	6.0957e-13	1.0770e-14	3.9870e-16
	`RI`	1.7227e+40	2.5740e+19	8.3380e-02
example (18)	`NEWTON`	3.2437e-17	3.0279e-17	9.9467e-18
	`ICARE`	1.9624e-16	1.8318e-16	6.0177e-17

given in the first block of Table 3 and the relative differences are

$$\begin{aligned}\operatorname{reldiff}(X_{k_{\max}}^{\texttt{NEWTON}}, X_{k_{\max}}^{\texttt{ICARE}}) &= 8.8968\texttt{e-}15,\\ \operatorname{reldiff}(X_{k_{\max}}^{\texttt{NEWTON}}, X_{k_{\max}}^{\texttt{RI}}) &= 1.0286\texttt{e-}13,\\ \operatorname{reldiff}(X_{k_{\max}}^{\texttt{ICARE}}, X_{k_{\max}}^{\texttt{RI}}) &= 1.1175\texttt{e-}13.\end{aligned}$$

This clearly shows that all methods approximate the same stabilizing solution.

Now, we modify the example data by increasing the positive definite part of the R matrix in (16) such that we have now

$$\begin{aligned}&A = \begin{bmatrix}2 & 1\\ 1 & -3\end{bmatrix}, \quad B = \begin{bmatrix}1 & 1\\ 0 & 2\end{bmatrix}, \quad R = \begin{bmatrix}-1 & 0\\ 0 & 2\end{bmatrix}, \quad C = \begin{bmatrix}1 & 1\end{bmatrix},\\ &E = I_2, \qquad S = 0, \qquad Q = 1.\end{aligned} \tag{17}$$

Similar to (16), we consider the case of an indefinite weighing matrix in the quadratic term of (1); however, the change in the data results in the stabilizing solution X_* being indefinite. The convergence behavior of `NEWTON` for this case is shown in Table 4. As in the previous example, the convergence is quadratic towards the stabilizing solution and the iterates do not show any monotonicity. Additionally, we do not have the stability of all closed-loop matrices during the iteration as the one computed in the first step is clearly unstable. We do not expect `RI` to work for this case due to X_* being indefinite and, in fact, we see in the second block row of Table 3 that `RI` does not converge to a solution of (1). However, `NEWTON` clearly converges to the correct solution with a relative difference to the solution computed by `ICARE` of

$$\operatorname{reldiff}(X_{k_{\max}}^{\texttt{NEWTON}}, X_{k_{\max}}^{\texttt{ICARE}}) = 5.6279\texttt{e-}15.$$

As final preliminary example, we want to investigate the effect of an indefinite constant term. Therefore, we modify the previous example as follows

$$\begin{aligned}&A = \begin{bmatrix}2 & 1\\ 1 & -3\end{bmatrix}, \quad B = \begin{bmatrix}1\\ 1\end{bmatrix}, \quad Q = \begin{bmatrix}1 & 0\\ 0 & -2\end{bmatrix}, \quad C = \begin{bmatrix}1 & 1\\ 0 & 2\end{bmatrix},\\ &E = I_2, \qquad S = 0, \qquad R = 1.\end{aligned} \tag{18}$$

Table 4: Convergence behavior of the Newton-Kleinman method (`NEWTON`) for the example (17): For each iteration step, the columns show the normalized residuals, the two eigenvalues of the current closed-loop matrix and the two eigenvalues of the difference of two consecutive iterates.

iter. step k	$\mathbf{res}_1(X_k)$	$\Lambda(A_k)$	$\Lambda(X_k - X_{k-1})$
1	1.3423e+01	2.3071, −7.8315	—
2	3.1646e-01	−4.1113, −1.4164	−1.3696e+02, −7.4641e-04
3	8.2620e-03	−4.0451, −1.4620	−7.8472e-01, 2.3569e-04
4	1.9458e-06	−4.0448, −1.4626	−8.1348e-03, 7.4808e-08
5	3.3469e-14	−4.0448, −1.4626	1.5635e-06, 1.1925e-11

Table 5: Convergence behavior of the Newton-Kleinman method (`NEWTON`) for the example (18): For each iteration step, the columns show the normalized residuals, the two eigenvalues of the current closed-loop matrix and the two eigenvalues of the difference of two consecutive iterates.

iter. step k	$\mathbf{res}_1(X_k)$	$\Lambda(A_k)$	$\Lambda(X_k - X_{k-1})$
1	1.9109e-01	−3.3289, −0.8111	—
2	1.4573e-02	−3.2914, −0.5227	−3.3630e-01, −1.3069e-02
3	7.0984e-04	−3.2887, −0.4383	−5.9143e-02, −9.6054e-04
4	5.7445e-06	−3.2886, −0.4301	−5.0185e-03, −7.8659e-06
5	5.0562e-10	−3.2886, −0.4300	−4.6517e-05, −4.3494e-09
6	3.0792e-17	−3.2886, −0.4300	−4.2059e-09, −2.1477e-14

Since we have already seen the effects of an indefinite quadratic term, we consider here $R > 0$ for simplicity. The stabilizing solution in this example is indefinite again. Table 5 shows the convergence behavior of `NEWTON` for this example case. We see exactly what was expected from Theorem 1: the closed-loop matrices are stable in all steps, the convergence is quadratic and monotonic. Since `RI` has not been extended to the case of indefinite constant terms and the solution is not positive semi-definite, we omit the comparing computations with this method here and only provide the results of `ICARE` instead. The residual norms can be found in the third block row of Table 3 and the relative difference between the solutions computed by `NEWTON` and `ICARE` is

$$\mathrm{reldiff}(X_{k_{\max}}^{\texttt{NEWTON}}, X_{k_{\max}}^{\texttt{ICARE}}) = 8.8968\texttt{e-}15.$$

Both methods appear to approximate the same stabilizing solution.

4.3 Numerical comparisons

In this section, we compare the proposed algorithm with established solvers in different benchmark data sets from the literature and equation scenarios. While we concentrate on examples with small-scale dense coefficient matrices in the first part to establish trust into the proposed `NEWTON` method, we present results for large-scale sparse matrices afterwards.

4.3.1 Examples with dense coefficient matrices

An overview about the experiments presented in this section is given in the first block row of Table 1. We decided to start by experimenting with small-scale dense coefficient matrices since for this case, there are well established solvers that can handle the general case (1), which we consider in this paper. Such a variety of methods is not given for large-scale sparse matrices; therefore, here, we numerically establish trust into the solutions obtained by `NEWTON` and show that they provide reasonable accuracy in comparison to other approaches. For the comparison, we have selected `SIGN` and `ICARE` as two well-established approaches for general CAREs with dense coefficient matrices. The results of the experiments are shown in Table 6 in form of the residual norms for the different methods and in Table 7, which shows the relative differences between the solutions computed by the different approaches. For further experimental metrics such as the amount of iteration steps taken by `NEWTON` and `SIGN`, computation times, and more, we refer the reader to the log files of the experiments in the accompanying code package [58].

Overall, we can evaluate that `NEWTON` performs comparably well or even best among all those methods. Note that we used 10^{-12} as convergence tolerance for the normalized residual norm internally computed by `NEWTON` such that we do not expect much smaller values for $\mathrm{res}_1(X_{k_{\max}})$ in Table 6. Despite that, `NEWTON` shows in various examples up to one order of magnitude better residuals than `ICARE` and often several orders of magnitude better residuals than `SIGN`. The relative differences in Table 7 show numerically that all three methods approximate the same stabilizing solution to the example equations and provide similar solutions with many significant digits of accuracy in common. With these results at hand, we believe that applying `NEWTON` in the large-scale sparse setting will provide correct as well as sufficiently accurate solutions.

4.3.2 Examples with large-scale sparse coefficient matrices

Now we consider the case of CARE examples with large-scale sparse coefficient matrices. An overview about these experiments is given in the second block row of Table 1. Whenever possible, we used `RI` as comparison method where we chose RADI as solver for the Riccati equations with positive semi-definite quadratic terms occurring in each step of the iteration. The residual norms of the computed results are shown in Table 8 and the relative differences for examples in which `NEWTON` and `RI` could be applied in Table 9.

The residual norms in Table 8 show `NEWTON` to provide accurate solutions to all example equations. It stands out that, in all examples, `NEWTON` provides residual norms that are at least three orders of magnitude better than those of the solutions provided by `RI`. One possible explanation for these results is that in `RI`, the overall solution is accumulated via column concatenation and truncation. This easily leads to the loss of numerical accuracy especially in the cases when the stabilizing solution is badly conditioned. For $\texttt{rail}_{(6)}$ (`LQG`), we could not use `RI` for the comparison, since the constant term in this example is indefinite by construction. The stabilizing solution however is numerically positive semi-definite.

The convergence behavior of `NEWTON` and `RI` for the example equations on the data set $\texttt{rail}_{(6)}$ is illustrated in Figure 1. These and similar plots for the other sparse examples can be found in the accompanying code package [58]. The plots show that for the cases that `RI` was applicable, it strongly outperformed `NEWTON` in terms of computation time. This is a result of the choice for the internal CARE solver in `RI`, which in our experiments was the RADI method [11]. The residuals shown are those that the methods implicitly compute during the iterations to determine convergence. Comparing these plots with

Table 6: Residual norms for all dense test examples and comparison methods in Section 4.3.1. `NEWTON` provides reasonably accurate and often the most accurate approximations compared to `SIGN` and `ICARE`.

example	method	$\mathbf{res_1}(\boldsymbol{X}_{k_{\max}})$	$\mathbf{res_2}(\boldsymbol{X}_{k_{\max}})$	$\mathbf{res_3}(\boldsymbol{X}_{k_{\max}})$
aircraft (LQG)	NEWTON	4.1073e-08	1.3682e-20	4.5781e-27
	SIGN	2.6483e-05	8.8220e-18	2.9519e-24
	ICARE	1.0713e-07	3.5686e-20	1.1941e-26
aircraft (HINF)	NEWTON	7.4736e-07	2.4035e-20	9.7826e-26
	SIGN	3.0047e-06	9.6631e-20	3.9330e-25
	ICARE	1.9949e-05	6.4155e-19	2.6112e-24
$\texttt{rail}_{(1)}$ (LQG)	NEWTON	7.5678e-14	1.0229e-14	8.8312e-16
	SIGN	3.1905e-10	4.3122e-11	3.7231e-12
	ICARE	2.0819e-13	2.8139e-14	2.4294e-15
$\texttt{rail}_{(1)}$ (HINF)	NEWTON	2.4749e-12	4.0514e-13	8.6384e-16
	SIGN	5.5336e-10	9.0584e-11	1.9314e-13
	ICARE	5.4824e-14	8.9746e-15	1.9136e-17
$\texttt{rail}_{(1)}$ (BR)	NEWTON	1.5986e-13	1.4855e-14	6.2186e-15
	SIGN	3.1304e-14	2.9089e-15	1.2178e-15
	ICARE	1.3867e-13	1.2886e-14	5.3943e-15
$\texttt{rail}_{(1)}$ (PR)	NEWTON	9.3161e-12	6.4179e-13	2.3614e-13
	SIGN	4.3343e-10	2.9859e-11	1.0986e-11
	ICARE	6.4693e-14	4.4567e-15	1.6398e-15
$\texttt{triplechain}_{(1)}$(BR)	NEWTON	3.6923e-11	3.0799e-16	1.5397e-16
	SIGN	9.2343e-11	7.7027e-16	3.8506e-16
	ICARE	1.6221e-10	1.3531e-15	6.7641e-16
$\texttt{triplechain}_{(1)}$(PR)	NEWTON	6.4378e-12	6.9411e-17	1.4807e-15
	SIGN	3.6829e-12	3.9708e-17	8.4708e-16
	ICARE	1.6286e-11	1.7559e-16	3.7457e-15

Table 7: Relative differences for the dense test examples in Section 4.3.1. All differences are reasonably low such that numerically we can rely on the results provided by NEWTON.

example	$\mathbf{reldiff}(\boldsymbol{X}_{k_{\max}}^{\texttt{NEWTON}}, \boldsymbol{X}_{k_{\max}}^{\texttt{SIGN}})$	$\mathbf{reldiff}(\boldsymbol{X}_{k_{\max}}^{\texttt{NEWTON}}, \boldsymbol{X}_{k_{\max}}^{\texttt{ICARE}})$
aircraft (LQG)	1.5687e-12	5.2915e-14
aircraft (HINF)	3.4874e-13	5.4917e-13
$\texttt{rail}_{(1)}$ (LQG)	3.3423e-10	1.0539e-11
$\texttt{rail}_{(1)}$ (HINF)	8.0362e-10	8.0170e-10
$\texttt{rail}_{(1)}$ (BR)	9.2682e-13	9.2764e-13
$\texttt{rail}_{(1)}$ (PR)	6.4985e-10	6.6590e-10
$\texttt{triplechain}_{(1)}$ (BR)	2.3524e-11	4.8920e-11
$\texttt{triplechain}_{(1)}$ (PR)	1.7659e-12	1.1858e-10

Table 8 reveals that the residuals internally computed by RI strongly diverge from the actual normalized residual norm $\mathrm{res}_1(X_{k_{\max}})$, which is several orders of magnitude larger. On the other hand, for NEWTON the results seem to coincide very well.

5 Conclusions

In this work, we presented a new formulation of the Newton-Kleinman iteration for solving general continuous-time algebraic Riccati equations with large-scale sparse coefficient matrices using low-rank indefinite symmetric LDL^T factorizations of the solution. For relevant scenarios from the literature, we could show the theoretical convergence of the algorithm. We provided the updated formulas for an exact line search procedure and inexact inner solves, and we outlined how to handle the case of projected algebraic Riccati equations occurring for matrix pencils with infinite eigenvalues. The numerical experiments show that our proposed algorithm provides reliable and accurate solutions to the considered problem and that even in the cases for which we could not provide a convergence theory, the algorithm appears to work perfectly fine.

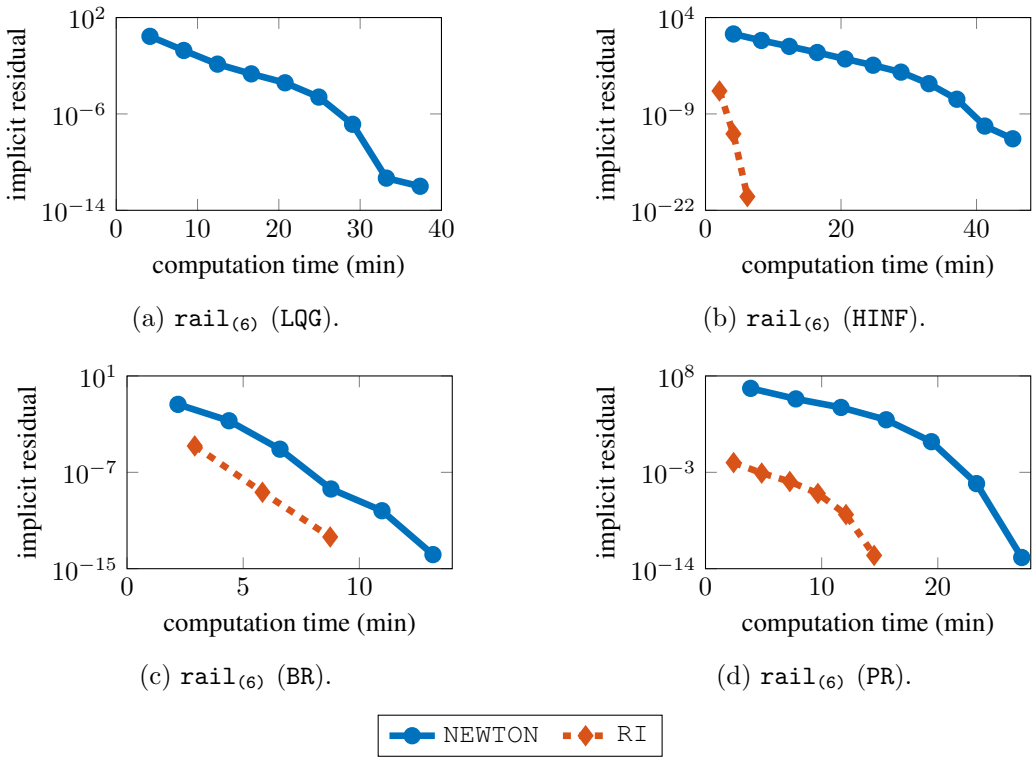
While we were able to provide convergence results for many of the practically occurring cases, the convergence behavior for the case of indefinite quadratic terms remains unsolved. The numerical results suggest that even in this situation, the proposed Newton-Kleinman method converges to the correct solution, however, the lack of monotonicity in the constructed iterates prevents the use of established strategies for proving convergence. Also, we have observed in our experiments that, while the new Newton-Kleinman iteration outperformed all comparing methods (if there were any at all) in terms of accuracy, it could not compete in the large-scale sparse case with the computational speed of the Riccati iteration that employed the RADI method as inner solver. Therefore, it is in our interest to investigate possible extensions of other, potentially faster performing methods to the case of general algebraic Riccati equations.

Table 8: Residual norms for all sparse test examples and comparison methods in Section 4.3.2. `NEWTON` provides very accurate approximations throughout all examples with residual norms up to eight orders of magnitude smaller than `RI`.

example	method	$\mathbf{res_1}(X_{k_{\max}})$	$\mathbf{res_2}(X_{k_{\max}})$	$\mathbf{res_3}(X_{k_{\max}})$
`msd` (BR)	`NEWTON`	2.1781e-13	1.4945e-17	1.8157e-19
	`RI`	1.6943e-08	1.1626e-12	1.4124e-14
$\texttt{triplechain}_{(2)}$ (BR)	`NEWTON`	7.0476e-13	7.5953e-21	3.4138e-21
	`RI`	3.6606e-05	3.9451e-13	1.7732e-13
$\texttt{triplechain}_{(2)}$ (PR)	`NEWTON`	5.0010e-13	4.6881e-21	3.5724e-24
	`RI`	3.4240e-05	3.2098e-13	2.4459e-16
$\texttt{rail}_{(6)}$ (LQG)	`NEWTON`	9.6445e-13	6.6215e-14	2.8035e-14
$\texttt{rail}_{(6)}$ (HINF)	`NEWTON`	4.5192e-13	2.8799e-14	1.4346e-15
	`RI`	1.1576e-10	7.3768e-12	3.6748e-13
$\texttt{rail}_{(6)}$ (BR)	`NEWTON`	8.8948e-14	4.6198e-15	2.2423e-15
	`RI`	1.6442e-10	8.5396e-12	4.1448e-12
$\texttt{rail}_{(6)}$ (PR)	`NEWTON`	2.8870e-13	5.3150e-16	2.6375e-16
	`RI`	5.7022e-09	1.0498e-11	5.2095e-12

Table 9: Relative differences for all sparse examples in Section 4.3.2. All differences are reasonably low such that numerically we can rely on the results provided by `NEWTON`. Due to the lack of comparison methods, relative differences could not be provided for all test scenarios.

example	$\mathbf{reldiff}(X_{k_{\max}}^{\texttt{NEWTON}}, X_{k_{\max}}^{\texttt{RI}})$
`msd` (BR)	1.8490e-12
$\texttt{triplechain}_{(2)}$ (BR)	6.9416e-11
$\texttt{triplechain}_{(2)}$ (PR)	6.9221e-11
$\texttt{rail}_{(6)}$ (HINF)	5.4646e-10
$\texttt{rail}_{(6)}$ (BR)	4.9422e-08
$\texttt{rail}_{(6)}$ (PR)	8.6206e-10



(a) `rail(6)` (`LQG`). (b) `rail(6)` (`HINF`).

(c) `rail(6)` (`BR`). (d) `rail(6)` (`PR`).

Figure 1: Convergence of `NEWTON` and `RI` for all example equations with the `rail(6)` data set: We can see that in all examples where it was applicable `RI` obtains its final approximation significantly faster than `NEWTON`. This comes from the use of RADI as underlying solver. However, the shown implicit residual computed by `RI` is not accurate as shown in Table 8.

References

[1] G. S. Ammar, P. Benner, and V. Mehrmann. A multishift algorithm for the numerical solution of algebraic Riccati equations. *Electron. Trans. Numer. Anal.*, 1:33–48, 1993. URL: https://etna.math.kent.edu/volumes/1993-2000/vol1/abstract.php?vol=1&pages=33-48.

[2] B. D. O. Anderson and J. B. Moore. *Optimal Control: Linear Quadratic Methods*. Prentice-Hall, Englewood Cliffs, NJ, 1990.

[3] B. D. O. Anderson and B. Vongpanitlerd. *Network Analysis and Synthesis: A Modern Systems Approach*. Networks Series. Prentice-Hall, Englewood Cliffs, NJ, 1972.

[4] W. F. Arnold. Numerical solution of algebraic matrix Riccati equations. Tech.-Report ADA139929, Naval Weapons Center, China Lake, CA, 1984. Public reprint of Ph.D. dissertation. URL: https://apps.dtic.mil/sti/citations/ADA139929.

[5] W. F. Arnold and A. J. Laub. Generalized eigenproblem algorithms and software for algebraic Riccati equations. *Proc. IEEE*, 72(12):1746–1754, 1984. doi:10.1109/PROC.1984.13083.

[6] E. Bänsch, P. Benner, J. Saak, and H. K. Weichelt. Riccati-based boundary feedback stabilization of incompressible Navier-Stokes flows. *SIAM J. Sci. Comput.*, 37(2):A832–A858, 2015. doi:10.1137/140980016.

[7] T. Başar and J. Moon. Riccati equations in Nash and Stackelberg differential and dynamic games. *IFAC-Pap.*, 50(1):9547–9554, 2017. 20th IFAC World Congress. doi:10.1016/j.ifacol.2017.08.1625.

[8] P. Benner. *Contributions to the Numerical Solution of Algebraic Riccati Equations and Related Eigenvalue Problems*. PhD thesis, Logos-Verlag, Berlin, 1997.

[9] P. Benner. Numerical solution of special algebraic Riccati equations via an exact line search method. In *1997 European Control Conference (ECC)*, pages 3136–3141, 1997. doi:10.23919/ECC.1997.7082591.

[10] P. Benner and Z. Bujanović. On the solution of large-scale algebraic Riccati equations by using low-dimensional invariant subspaces. *Linear Algebra Appl.*, 488:430–459, 2016. doi:10.1016/j.laa.2015.09.027.

[11] 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(2):301–330, 2018. doi:10.1007/s00211-017-0907-5.

[12] P. Benner, Z. Bujanović, P. Kürschner, and J. Saak. A numerical comparison of different solvers for large-scale, continuous-time algebraic Riccati equations and LQR problems. *SIAM J. Sci. Comput.*, 42(2):A957–A996, 2020. doi:10.1137/18M1220960.

[13] P. Benner and R. Byers. An exact line search method for solving generalized continuous-time algebraic Riccati equations. *IEEE Trans. Autom. Control*, 43(1):101–107, 1998. doi:10.1109/9.654908.

[14] P. Benner, P. Ezzatti, E. S. Quintana-Ortí, and A. Remón. A factored variant of the Newton iteration for the solution of algebraic Riccati equations via the

matrix sign function. *Numer. Algorithms*, 66(2):363–377, 2014. doi:10.1007/s11075-013-9739-2.

[15] P. Benner, J. Heiland, and S. W. R. Werner. Robust output-feedback stabilization for incompressible flows using low-dimensional $\mathcal{H}_\infty$-controllers. *Comput. Optim. Appl.*, 82(1):225–249, 2022. doi:10.1007/s10589-022-00359-x.

[16] P. Benner, J. Heiland, and S. W. R. Werner. A low-rank solution method for Riccati equations with indefinite quadratic terms. *Numer. Algorithms*, 92(2):1083–1103, 2023. doi:10.1007/s11075-022-01331-w.

[17] 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.*, 108:125–142, 2016. doi:10.1016/j.apnum.2016.05.006.

[18] P. Benner, M. Köhler, and J. Saak. Matrix equations, sparse solvers: M-M.E.S.S.-2.0.1—Philosophy, features and application for (parametric) model order reduction. In P. Benner, T. Breiten, H. Faßbender, M. Hinze, T. Stykel, and R. Zimmermann, editors, *Model Reduction of Complex Dynamical Systems*, volume 171 of *International Series of Numerical Mathematics*, pages 369–392. Birkhäuser, Cham, 2021. doi:10.1007/978-3-030-72983-7_18.

[19] P. Benner, P. Kürschner, and J. Saak. Efficient handling of complex shift parameters in the low-rank Cholesky factor ADI method. *Numer. Algorithms*, 62(2):225–251, 2013. doi:10.1007/s11075-012-9569-7.

[20] P. Benner, P. Kürschner, and J. Saak. Self-generating and efficient shift parameters in ADI methods for large Lyapunov and Sylvester equations. *Electron. Trans. Numer. Anal.*, 43:142–162, 2014. URL: https://etna.mcs.kent.edu/volumes/2011-2020/vol43/abstract.php?vol=43&pages=142-162.

[21] P. Benner, J.-R. Li, and T. Penzl. Numerical solution of large-scale Lyapunov equations, Riccati equations, and linear-quadratic optimal control problems. *Numer. Lin. Alg. Appl.*, 15(9):755–777, 2008. doi:10.1002/nla.622.

[22] P. Benner and J. Saak. Linear-quadratic regulator design for optimal cooling of steel profiles. Technical Report SFB393/05-05, Sonderforschungsbereich 393 *Parallele Numerische Simulation für Physik und Kontinuumsmechanik*, TU Chemnitz, Chemnitz, Germany, 2005. URL: http://nbn-resolving.de/urn:nbn:de:swb:ch1-200601597.

[23] P. Benner and J. Saak. Numerical solution of large and sparse continuous time algebraic matrix Riccati and Lyapunov equations: a state of the art survey. *GAMM-Mitt.*, 36(1):32–52, 2013. doi:10.1002/gamm.201310003.

[24] P. Benner, J. Saak, and S. W. R. Werner. MORLAB – Model Order Reduction LABoratory (version 6.0), September 2023. See also: https://www.mpi-magdeburg.mpg.de/projects/morlab. doi:10.5281/zenodo.7072831.

[25] P. Benner and T. Stykel. Numerical solution of projected algebraic Riccati equations. *SIAM J. Numer. Anal.*, 52(2):581–600, 2014. doi:10.1137/130923993.

[26] P. Benner and T. Stykel. Model order reduction for differential-algebraic equations: A survey. In A. Ilchmann and T. Reis, editors, *Surveys in Differential-Algebraic Equations IV*, Differential-Algebraic Equations Forum, pages 107–160. Springer, Cham, 2017. doi:10.1007/978-3-319-46618-7_3.

[27] P. Benner and S. W. R. Werner. MORLAB—The Model Order Reduction LABoratory. In P. Benner, T. Breiten, H. Faßbender, M. Hinze, T. Stykel, and R. Zimmermann, editors, *Model Reduction of Complex Dynamical Systems*, volume 171 of *International Series of Numerical Mathematics*, pages 393–415. Birkhäuser, Cham, 2021. doi:10.1007/978-3-030-72983-7_19.

[28] C. Bertram and H. Faßbender. On a family of low-rank algorithms for large-scale algebraic Riccati equations. e-print 2304.01624, arXiv, 2023. Numerical Analysis (math.NA). doi:10.48550/arXiv.2304.01624.

[29] E. K.-W. Chu, H.-Y. Fan, and W.-W. Lin. A structure-preserving doubling algorithm for continuous-time algebraic Riccati equations. *Linear Algebra Appl.*, 396:55–80, 2005. doi:10.1016/j.laa.2004.10.010.

[30] M. C. Delfour. Linear quadratic differential games: Saddle point and Riccati differential equation. *SIAM J. Control Optim.*, 46(2):750–774, 2007. doi:10.1137/050639089.

[31] U. B. Desai and D. Pal. A realization approach to stochastic model reduction and balanced stochastic realizations. In *21st IEEE Conference on Decision and Control*, pages 1105–1112, 1982. doi:10.1109/CDC.1982.268322.

[32] F. Feitzinger, T. Hylla, and E. W. Sachs. Inexact Kleinman-Newton method for Riccati equations. *SIAM J. Matrix Anal. Appl.*, 31(2):272–288, 2009. doi:10.1137/070700978.

[33] F. Freitas, J. Rommes, and N. Martins. Gramian-based reduction method applied to large sparse power system descriptor models. *IEEE Trans. Power Syst.*, 23(3):1258–1270, 2008. doi:10.1109/TPWRS.2008.926693.

[34] G. H. Golub and C. F. Van Loan. *Matrix Computations*. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, Baltimore, fourth edition, 2013.

[35] 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.*, 33:53–62, 2009. URL: https://etna.math.kent.edu/volumes/2001-2010/vol33/abstract.php?vol=33&pages=53-62.

[36] E. A. Jonckheere and L. M. Silverman. A new set of invariants for linear systems–application to reduced order compensator design. *IEEE Trans. Autom. Control*, 28(10):953–964, 1983. doi:10.1109/TAC.1983.1103159.

[37] M. Kimura. Doubling algorithm for continuous-time algebraic Riccati equation. *Int. J. Syst. Sci.*, 20(2):191–202, 1989. doi:10.1080/00207728908910119.

[38] D. L. Kleinman. On an iterative technique for Riccati equation computations. *IEEE Trans. Autom. Control*, 13(1):114–115, 1968. doi:10.1109/TAC.1968.1098829.

[39] P. Kürschner. *Efficient Low-Rank Solution of Large-Scale Matrix Equations*. Dissertation, Otto-von-Guericke-Universität, Magdeburg, Germany, 2016. URL: http://hdl.handle.net/11858/00-001M-0000-0029-CE18-2.

[40] P. Lancaster and L. Rodman. *Algebraic Riccati Equations*. Oxford Science Publications. The Clarendon Press, Oxford University Press, New York, 1995.

[41] N. Lang, H. Mena, and J. Saak. An LDL^T factorization based ADI algorithm for solving large-scale differential matrix equations. *Proc. Appl. Math. Mech.*, 14(1):827–828, 2014. doi:10.1002/pamm.201410394.

[42] A. Lanzon, Y. Feng, B. D. .O. Anderson, and M. Rotkowitz. Computing the positive stabilizing solution to algebraic Riccati equations with an indefinite quadratic term via a recursive method. *IEEE Trans. Autom. Control*, 53(10):2280–2291, 2008. doi:10.1109/TAC.2008.2006108.

[43] A. J. Laub. A Schur method for solving algebraic Riccati equations. *IEEE Trans. Autom. Control*, 24(6):913–921, 1979. doi:10.1109/TAC.1979.1102178.

[44] F. Leibfritz. *COMPl$_e$ib*: *CO*nstrained *M*atrix-optimization *P*roblem *lib*rary – a collection of test examples for nonlinear semidefinite programs, control system design and related problems. Tech.-report, University of Trier, 2004. URL: http://www.friedemann-leibfritz.de/COMPlib_Data/COMPlib_Main_Paper.pdf.

[45] J.-R. Li and J. White. Low rank solution of Lyapunov equations. *SIAM J. Matrix Anal. Appl.*, 24(1):260–280, 2002. doi:10.1137/S0895479801384937.

[46] Y. Lin and V. Simoncini. A new subspace iteration method for the algebraic Riccati equation. *Numer. Linear Algebra Appl.*, 22(1):26–47, 2015. doi:10.1002/nla.1936.

[47] A. Locatelli. *Optimal Control: An Introduction*. Birkhäuser, Basel, 2001.

[48] Green M. A relative error bound for balanced stochastic truncation. *IEEE Trans. Autom. Control*, 33(10):961–965, 1988. doi:10.1109/9.7255.

[49] V. Mehrmann and T. Stykel. Balanced truncation model reduction for large-scale systems in descriptor form. In P. Benner, V. Mehrmann, and D. C. Sorensen, editors, *Dimension Reduction of Large-Scale Systems*, volume 45 of *Lect. Notes Comput. Sci. Eng.*, pages 83–115. Springer, Berlin, Heidelberg, 2005. doi:10.1007/3-540-27909-1_3.

[50] D. Mustafa and K. Glover. Controller reduction by $\mathcal{H}_\infty$-balanced truncation. *IEEE Trans. Autom. Control*, 36(6):668–682, 1991. doi:10.1109/9.86941.

[51] Oberwolfach Benchmark Collection. Steel profile. hosted at MORwiki – Model Order Reduction Wiki, 2005. URL: https://morwiki.mpi-magdeburg.mpg.de/morwiki/index.php/Steel_Profile.

[52] P. C. Opdenacker and E. A. Jonckheere. A contraction mapping preserving balanced reduction scheme and its infinity norm error bounds. *IEEE Trans. Circuits Syst.*, 35(2):184–189, 1988. doi:10.1109/31.1720.

[53] J. D. Roberts. Linear model reduction and solution of the algebraic Riccati equation by use of the sign function. *Internat. J. Control*, 32(4):677–687, 1980. Reprint of Technical Report No. TR-13, CUED/B-Control, Cambridge University, Engineering Department, 1971. doi:10.1080/00207178008922881.

[54] J. Saak and M. Behr. Reimplementation of optimal cooling process for a steel profile of a rail, 2020. URL: https://gitlab.mpi-magdeburg.mpg.de/models/fenicsrail.

[55] J. Saak and M. Behr. The Oberwolfach steel-profile benchmark revisited, July 2021. doi:10.5281/zenodo.5113560.

[56] J. Saak, M. Köhler, and P. Benner. M-M.E.S.S. – The Matrix Equations Sparse Solvers library (version 3.0), August 2023. See also: https://www.mpi-magdeburg.mpg.de/projects/mess. doi:10.5281/zenodo.7701424.

[57] J. Saak and M. Voigt. Model reduction of constrained mechanical systems in M-M.E.S.S. *IFAC-Pap.*, 51(2):661–666, 2018. 9th Vienna International Conference on Mathematical Modelling MATHMOD 2018. doi:10.1016/j.ifacol.2018.03.112.

[58] J. Saak and S. W. R. Werner. Code, data and results for numerical experiments in "Using LDL^T factorizations in Newton's method for solving general large-scale algebraic Riccati equations" (version 1.0), February 2024. doi:10.5281/zenodo.10619037.

[59] N. Sandell. On Newton's method for Riccati equation solution. *IEEE Trans. Autom. Control*, 19(3):254–255, 1974. doi:10.1109/TAC.1974.1100536.

[60] V. Simoncini. Analysis of the rational Krylov subspace projection method for large-scale algebraic Riccati equations. *SIAM J. Matrix Anal. Appl.*, 37(4):1655–1674, 2016. doi:10.1137/16M1059382.

[61] E. D. Sontag. *Mathematical Control Theory*, volume 6 of *Texts in Applied Mathematics*. Springer, New York, second edition, 1998. doi:10.1007/978-1-4612-0577-7.

[62] T. Stillfjord. Singular value decay of operator-valued differential Lyapunov and Riccati equations. *SIAM J. Control Optim.*, 56(5):3598–3618, 2018. doi:10.1137/18M1178815.

[63] T. Stykel. Low-rank iterative methods for projected generalized Lyapunov equations. *Electron. Trans. Numer. Anal.*, 30:187–202, 2008. URL: https://etna.math.kent.edu/volumes/2001-2010/vol30/abstract.php?vol=30&pages=187-202.

[64] N. Truhar and K. Veselić. An efficient method for estimating the optimal dampers' viscosity for linear vibrating systems using Lyapunov equation. *SIAM J. Matrix Anal. Appl.*, 31(1):18–39, 2009. doi:10.1137/070683052.

[65] A. Varga. On computing high accuracy solutions of a class of Riccati equations. *Control–Theory and Advanced Technology*, 10(4):2005–2016, 1995.

[66] H. K. Weichelt. *Numerical Aspects of Flow Stabilization by Riccati Feedback*. Dissertation, Otto-von-Guericke-Universität, Magdeburg, Germany, 2016. doi:10.25673/4493.