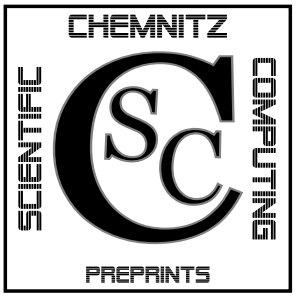
TECHNISCHE UNIVERSITÄT CHEMNITZ

Peter Benner Heike Faßbender

On the numerical solution of large-scale sparse discrete-time Riccati equations

CSC/09-11



Chemnitz Scientific Computing Preprints

Impressum:

Chemnitz Scientific Computing Preprints — ISSN 1864-0087
(1995–2005: Preprintreihe des Chemnitzer SFB393)

Herausgeber:
Professuren für
Numerische und Angewandte Mathematik
an der Fakultät für Mathematik
der Technischen Universität Chemnitz

Postanschrift:
TU Chemnitz, Fakultät für Mathematik
09107 Chemnitz
Sitz:
Reichenhainer Str. 41, 09126 Chemnitz

`http://www.tu-chemnitz.de/mathematik/csc/`

TECHNISCHE UNIVERSITÄT CHEMNITZ

Chemnitz Scientific Computing Preprints

Peter Benner Heike Faßbender

On the numerical solution of large-scale sparse discrete-time Riccati equations

CSC/09-11

CSC/09-11 ISSN 1864-0087 December 2009

Contents

Author's addresses:

Peter Benner
TU Chemnitz
Fakultät für Mathematik
D-09107 Chemnitz

benner@mathematik.tu-chemnitz.de

Heike Faßbender
TU Braunschweig
Carl-Friedrich-Gauß-Fakultät
Institut Computational Mathematics
AG Numerik
38092 Braunschweig

h.fassbender@tu-bs.de

Abstract

The numerical solution of Stein (aka discrete Lyapunov) equations is the primary step in Newton's method for the solution of discrete-time algebraic Riccati equations (DARE). Here we present a low-rank Smith method as well as a low-rank alternating-direction-implicit-iteration to compute low-rank approximations to solutions of Stein equations arising in this context. Numerical results are given to verify the efficiency and accuracy of the proposed algorithms.

Keywords. Riccati equation, discrete-time control, sparse matrices, Newton's method, Smith iteration, ADI iteration, low rank factor.

AMS subject classifications (MSC2010). 65F30, 15A24, 93C20, 93C05.

1 Introduction

After full (time and spatial) discretization, the solution of optimal control problems for parabolic partial differential equations (PDEs) leads to generalized large-scale, sparse discrete–time algebraic Riccati equation (DARE)

$$\begin{aligned} 0 \;=\; \mathcal{R}(X) \;&=\; CQC^T + AXA^T - EXE^T \\ &\quad -(AXB^T + CS^T)(R + BXB^T)^{-1}(BXA^T + SC^T), \end{aligned} \tag{1}$$

where A and $E \in \mathbb{R}^{n\times n}$ are large and sparse, $B \in \mathbb{R}^{m\times n}, C \in \mathbb{R}^{n\times p}, Q \in \mathbb{R}^{p\times p}, R \in \mathbb{R}^{m\times m}, S \in \mathbb{R}^{m\times p}$, and Q and R are symmetric. We also assume E to be invertible throughout.

In the last decades, much research has addressed the construction of numerically robust algorithms for the solution of (1). However, these methods generally have at least a memory complexity $\mathcal{O}(n^2)$ and a computational complexity $\mathcal{O}(n^3)$, regardless whether or not the system matrix A is sparse or otherwise structured. Therefore, the majority of numerical algorithms is restricted to systems of moderate order. Of course, the upper limit for this order depends on the problem to be solved as well as the particular computing environment and may vary between a few hundreds and a few thousands. However, a significant number of applications lead to systems of larger order. Large systems arise from the semi-discretization of (possibly linearized) PDEs by means of finite differences or finite elements, see e.g. [4, 36, 40], and many more.

Consider, e.g., the problem

$$\text{Minimize} \quad \mathcal{J}(u) = \frac{1}{2}\int_0^\infty y(t)^T Q y(t) + u(t)^T R u(t) + 2y(t)^T S u(t) dt$$

subject to the PDE constraint

$$\begin{aligned}
\frac{\partial}{\partial t}x(\zeta,t) &= \nabla(k(\zeta)\nabla)x + c(\zeta)\nabla x + r(\zeta)x + b(\zeta)u \qquad \text{in } \Omega\times[0,T],\\
x(\zeta,t) &= 0 \qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \text{on } \partial\Omega,\\
x(\zeta,0) &= x_0(\zeta) \qquad\qquad\qquad\qquad\qquad\qquad\quad \text{on } \Omega,\\
y(t) &= \int_{\Omega_0} c(\zeta)x(\zeta,t)d\zeta, \qquad\qquad\qquad\quad \Omega_0\subset\overline{\Omega}.
\end{aligned}$$

Discretizing the PDE constraint using a finite-difference scheme or a finite element approach leads to a constraint in form of an ordinary differential equation

$$\begin{aligned}
M\dot{x}(t) &= Kx(t) + Fu,\\
y(t) &= Cx(t),
\end{aligned}$$

where M and K are large, sparse square matrices, M is positive definite and K is negative definite. Employing further a time discretization a difference equation of the form

$$Ex_{k+1} = Ax_k + Bu_k$$

is obtained. In case the semi-implicit Euler method with stepsize Δt is used, one obtains

$$(M - \Delta tK)x_{k+1} = Mx_k + \Delta tFu_k, \tag{2}$$

that is $A = M$, $E = M - \Delta tK$ and $B = \Delta tF$. Alternatively, a Crank-Nicholson-type discretization can be used. Here, we will use the average of a forward Euler and a semi-implicit Euler step in time. That is, we average the equations

$$\begin{aligned}
Mx_{k+1} &= (M + \Delta tK)x_k + \Delta tFu_k,\\
Mx_{k+1} &= Mx_k + \Delta tKx_{k+1} + \Delta tFu_k,
\end{aligned}$$

and obtain

$$\begin{aligned}
(M - \frac{\Delta t}{2}K)x_{k+1} &= (M + \frac{\Delta t}{2}K)x_k + \Delta tFu_k,\\
y_k &= Cx_k.
\end{aligned}$$

The discretized optimal control problem now reads

$$\text{Minimize } \mathcal{J}(u) = \frac{1}{2}\sum_{k=0}^{\infty} y_k^T Qy_k + u_k^T Ru_k + 2y_k^T Su_k$$

subject to

$$\begin{aligned}
Ex_{k+1} &= Ax_k + Bu_k,\\
y_k &= Cx_k,
\end{aligned}$$

where $E = M - \frac{\Delta t}{2}K, A = M + \frac{\Delta t}{2}K$, $B = \Delta t F$.

Under generic control-theoretic conditions, the optimal solution of this minimization problem is given by the feedback control

$$u_k = -(R + B^T X_d B)^{-1}(B^T X_d A + S^T C)x_k, \qquad k = 0, 1, \ldots,$$

where X_d is the stabilizing solution of the DARE (1); see, e.g., [21, 39, 44, 52] and many other textbooks on control theory. DAREs also arise in other applications such as $\mathcal{H}_\infty$-control, factorization problems for rational matrix functions, Kalman filtering. An overview of some of these applications is given in [39, Chapter 5]. A detailed discussion of the solution theory for the case $E = I$ (which is equivalent to the case E nonsingular) is given in [39], whereas the case of singular E is treated in [44]. Solutions of the optimal control problem without solving the corresponding Riccati equation are given in [7, 16, 35, 43, 44], including the singular E case. It should be noted, though, that DARE-free solutions of the linear-quadratic discrete-time optimal control problem require the full n-dimensional deflating subspace of the corresponding (generalized) symplectic pencil. For large-scale problems, this is prohibitive as even if the sparsity structure of the problem-defining matrices could be employed, the subspace itself would require memory of size $2n^2$.

Mostly, systems originating from the application mentioned above possess two interesting properties. First, their order n is large (say, $n > 1000$), but the dimensions of the input and output spaces are relatively small ($m, q \ll n$, often $m, q \ll 10$). For example, the order of a system arising from a parabolic PDE is about the number of grid points or mesh nodes used for the semi-discretization w.r.t. the spatial coordinates, which is relatively large. In contrast, m and q are often quite small and independent of the fineness of the discretization. Second, the system matrix A is structured. Often, A is a sparse matrix or it is implicitly represented as a product of sparse matrices and inverses of sparse matrices. In general, this structure allows the numerically inexpensive realization of matrix-vector products and the efficient solution of systems of linear equations with A.

In the sequel we will use the abbreviation

$$K(X) \;:=\; (AXB^T + CS^T)(R + BXB^T)^{-1}, \tag{3}$$

to simplify the notation and we will make the following assumptions:

1. E is nonsingular.
2. $R = R^T > 0$.
3. $\begin{bmatrix} Q & S^T \\ S & R \end{bmatrix} \geq 0$.
4. A stabilizing solution X_d of (1) exists, that is, a solution X_d exists such that the eigenvalues of $(A - K(X_d)B) - \lambda E$ lie in the open unit circle: $(\sigma(A - K(X_d)B, E) \subset \mathcal{D}_1(0))$. It is unique, and furthermore, $R + BX_dB^T > 0$.

(4)

For sufficient conditions for 4. to hold, see, e.g., [39, Theorem 13.1.3].

In principle, by inverting E, (1) can be reduced to the case $E = I$. However, this introduces unnecessary rounding errors and, if E is ill-conditioned, even numerical instability. Therefore, inverting E is avoided here. Considering E is important, as in the PDE control problems $E \neq I$, and as E is large and sparse, its inverse in general would be large and dense, so that the large, sparse generalized DARE would be transformed into a large, dense one for which there do not exist any suitable numerical methods yet.

The solution of DAREs has been an extremely active area of research, see, e.g., [21, 44, 52] for an overview. The usual solution methods for DAREs such as the Schur vector method [46], symplectic SR methods [12, 23], the matrix sign function [7, 15, 26, 51], the matrix disk function [7, 15, 39, 55] or the doubling method [52, 42] do not make (full) use of the sparse structure of A, E and require in general $\mathcal{O}(n^3)$ flops and workspace of size $\mathcal{O}(n^2)$ even for sparse problems, and are therefore not suitable here.

The numerical solution of several types of large-scale matrix equations with sparse coefficient matrices arising in control theory has been the target of numerous papers in the last decade. Significant progress has been made in particular for continuous Lyapunov and algebraic Riccati equations, e.g., [8, 13, 53, 41, 49, 33]. For an overview and further references, see [9]. It is the aim of this paper to extend some of these results to DAREs. We will follow in particular the approach taken in [13]. That is, we will make use of the fact that $\mathcal{R}(X) = 0$ defines a system of nonlinear equations and can hence be solved by an appropriate Newton method as proposed in [32, 3]. Newton's method is reviewed in Section 2. The main computational cost in the algorithm stems from the numerical solution of the Stein equation. Section 3 proposes low rank Smith and ADI iterations for its solution based on [13, 17, 41, 49]. Both methods compute an approximate low-rank Cholesky factor of the desired solution. Numerical experiments are reported in Section 4.

2 Newton's method for discrete-time algebraic Riccati equations

The function $\mathcal{R}(X)$ in (1) is a rational matrix function, $\mathcal{R}(X) = 0$ defines a system of nonlinear equations. Inspired by Kleinman's formulation of a Newton method for continuous-time algebraic Riccati equations [37], Hewer [32] proposed a Newton method for solving DAREs. The algorithm was extended to the generalized equation as given in (1) by Arnold and Laub [3]. A discussion of its convergence properties can be found in [39, 44].

Newton's method for the numerical solution of DAREs can be formulated as given in Algorithm 1.

Algorithm 1 Newton's Method for the DARE

Input: The coefficient matrices A, B, C, E, Q, R, S of the DARE (1), and a starting guess X_0, so that $\sigma(A - K(X_0)B, E) \subset \mathcal{D}_1(0)$ and $R + BX_0B^T > 0$.
Output: An approximate solution X_{k+1} of the DARE (1) and an estimate N_k for the error matrix $X^* - X_{k+1}$, where X^* is the stabilizing solution of $\mathcal{R}(X) = 0$.

for $k = 0, 1, 2, \ldots$ **do**

1. $K_k \leftarrow K(X_k)$.
2. $A_k \leftarrow A - K_kB$.
3. $\mathcal{R}_k \leftarrow \mathcal{R}(X_k)$.
4. Solve for N_k in the Stein equation

$$A_kN_kA_k^T - EN_kE^T = -\mathcal{R}_k.$$

5. $X_{k+1} \leftarrow X_k + N_k$.

end for

We have the following convergence result for Algorithm 1 [32, 39, 44].

Theorem 2.1 *If the assumptions (4) hold, and X_0 is stabilizing, then for the iterates produced by Algorithm 1 we have:*

a) All iterates X_k are stabilizing, i.e., $\sigma(A - K(X_k)B, E) \subset \mathcal{D}_1(0)$ for all $k \in \mathbb{N}_0$.

b) $X_d \leq \ldots \leq X_{k+1} \leq X_k \leq \ldots \leq X_1$.

c) $\lim_{k\to\infty} X_k = X_d$.

d) There exists a constant $\gamma > 0$ such that

$$\|X_{k+1} - X_d\| \leq \gamma \|X_k - X_d\|^2, \qquad k \geq 1,$$

i.e., the X_k converge globally quadratic to X_d.

The formulation of Algorithm 1 is analogous to the standard formulation of Newton's method as given, e.g., in [22, Algorithm 5.1.1] for the solution of nonlinear equations. Because of its robustness in the presence of rounding errors, in dense matrix computations one prefers to calculate the Newton step explicitly as in Algorithm 1 using the Stein equation

$$A_k N_k A_k^T - E N_k E^T = -\mathcal{R}_k \tag{5}$$

rather than to use the mathematically equivalent formulation of the Newton step [3, 32, 39, 44],

$$\begin{aligned} A_k X_{k+1} A_k^T - E X_{k+1} E^T &= -CQC^T + CS^T K_k^T + K_k S C^T - K_k R K_k^T \\ &=: -\mathcal{C}(X_k) = -\mathcal{C}_k, \end{aligned} \tag{6}$$

which determines X_{k+1} directly. The coefficient matrices of the two Stein equations are the same, but the right-hand-sides are different; see Algorithm 2.

Algorithm 2 Newton-Hewer Method for the DARE

Input: The coefficient matrices A, B, C, E, Q, R, S of the DARE (1), and a starting guess X_0, so that $\sigma(A - K(X_0)B, E) \subset \mathcal{D}_1(0)$ and $R + BX_0B^T > 0$.
Output: An approximate solution X_{k+1} of the DARE (1).

for $k = 0, 1, 2, \ldots$ **do**

1. $K_k \leftarrow K(X_k)$.
2. $A_k \leftarrow A - K_k B$.
3. $\mathcal{C}_k \leftarrow \mathcal{C}(X_k)$.
4. Solve for X_{k+1} in the Stein equation

$$A_k X_{k+1} A_k^T - E X_{k+1} E^T = -\mathcal{C}_k.$$

end for

The problem for the successful application of the Newton method is to find a stabilizing initial guess X_0. There exist stabilization procedures for discrete-time linear systems (see, e.g., [2, 38, 52]). But these may give large initial errors $\|X_d - X_0\|$. The procedure suggested in [38] is even unfeasable for numerical computations as it is based on explicitly summing up $A^k BB^T (A^T)^k$ for k up to n, thereby often causing overflow already for small values of n. Suitable initialization procedures are suggested in [10, 11] which work quite well in practice, but

cannot exploit sparsity of the problem and are therefore not suitable for large-scale problems. An iterative procedure for stabilizing large-scale discrete-time systems is suggested in [25] and can be used for our purposes.

Despite the ultimate rapid convergence indicated by Theorem 2.1 d), the iteration may initially converge slowly. This can be due to a large initial error $\|X_d - X_0\|$ or a disastrously large first Newton step resulting in a large error $\|X_d - X_1\|$. In both cases, it is possible that many iterations are required to find the region of rapid convergence. An ill-conditioned Stein equation makes it difficult to compute an accurate Newton step. An inaccurately computed Newton step can cause the usual convergence theory to break down in practice. Sometimes rounding errors or a poor choice of X_0 cause Newton's method to converge to a non-stabilizing solution. Fortunately, in PDE control as considered here, we often have $\sigma(E, A) \subset \mathcal{D}_1(0)$, so that $X_0 = 0$ is an appropriate starting guess.

The computational cost for Algorithms 1 and 2 mainly depends upon the cost for the numerical solution of the Stein equation (5), (6), resp.. This can be done using the Bartels–Stewart algorithm [5, 6] or an extension to the case $E \neq I$ [27, 28, 48]. The Bartels-Stewart algorithm is the standard direct method for the solution of Stein equations of small to moderate size. This method requires the computation of a Schur decomposition, and thus is not appropriate for large scale problems. The cost for the solution of the Stein equation is $\approx 73n^3$ flops. See [52] for a discussion of an efficient implementation. In [52], the following idea based on [31, 56] how to solve stable nonnegative Stein equations

$$AXA^H - X = -C \tag{7}$$

where $C \in \mathbb{C}^{n\times n}$ is positive semidefinite and $A \in \mathbb{C}^{n\times n}$ is stable, is also discussed: The unique solution X is positive semidefinite and hence allows for a Cholesky factorization $X = LL^H$. Let U be the unitary matrix that reduces A to upper Schur form $T = UAU^H$, such that we have

$$(U^HTU)(LL^H)(U^HT^HU) - LL^H = -BB^H$$

where $BB^H = C$ and $B \in \mathbb{C}^{n\times m}, m \leq n$. Or, equivalently,

$$T(UL)(L^HU^H)T^H - (UL)(L^HU^H) = -(UB)(B^HU^H).$$

Using the RQ factorizations of UB and UL we obtain

$$TZZ^HT^H - ZZ^H = -RR^H,$$

where Z and R are triangular matrices. This equation can be solved efficiently in Z, the Cholesky factor of the solution of the original equation (7), see [52] for details. The memory space needed for the data is $2n^2$, while the computational cost is about $\approx cn^3$ flops, where c is of the order of unity. Although this method

is not suitable in the context of large sparse DAREs, we will make use of the idea of computing only a (Cholesky) factor of the desired solution.

Other iterative schemes for solving the Stein equation (5), (6), resp., have been developed. Most of these were first formulated for the continuous-time case. Examples are the Smith method [54], the sign-function method [51], and the alternating direction implicit (ADI) iteration method [57]. Unfortunately, all of these methods compute the solution in dense form and hence require $\mathcal{O}(n^2)$ storage. In case the solution to the Stein equation has low numerical rank (i.e., the eigenvalues decay rapidly) one can take advantage of this low rank structure to obtain approximate solutions in low rank factored form [14]. If the effective rank is $r \ll n$, then the storage is reduced from $\mathcal{O}(n^2)$ to $\mathcal{O}(nr)$. This approach will be discussed in detail in Section 3.

3 Smith's Method and ADI Iteration

In this section, iterative methods for solving the equivalent Stein equations

$$A_k Y A_k^T - E Y E^T = -\mathcal{R}_k \tag{8}$$

and

$$A_k Y A_k^T - E Y E^T = -\mathcal{C}_k \tag{9}$$

with

$$A_k = A - K_k B, \quad K_k = K(X_k), \quad \mathcal{R}_k = \mathcal{R}(X_k), \quad \mathcal{C}_k = \mathcal{C}(X_k)$$

will be discussed. In particular, it will be assumed that the solution Y has numerically low rank. This is usually the case if the DARE stems from a discretized PDE control problem as illustrated in Figure 1, where the eigenvalues of the solution of the DARE corresponding to the discrete heat equation example from the SLICOT Model Reduction Benchmark Collection [19] are shown, see also Example 4.2. Here, the numerical rank of X is 31 as compared to $n = 200$.

The symmetric Stein equation

$$S - A S A^T = V, \tag{10}$$

where A and V are given $n \times n$ matrices and V is symmetric, has a unique solution S (necessarily symmetric) provided $z\overline{w} \neq 1$ for every pair of eigenvalues z, w of A, see [39, pp. 104–105]. In particular, this is the case when all eigenvalues of A lie in the open unit disk. Then the unique solution is given by

$$S = \sum_{j=0}^{\infty} A^j V A^{Tj} \tag{11}$$

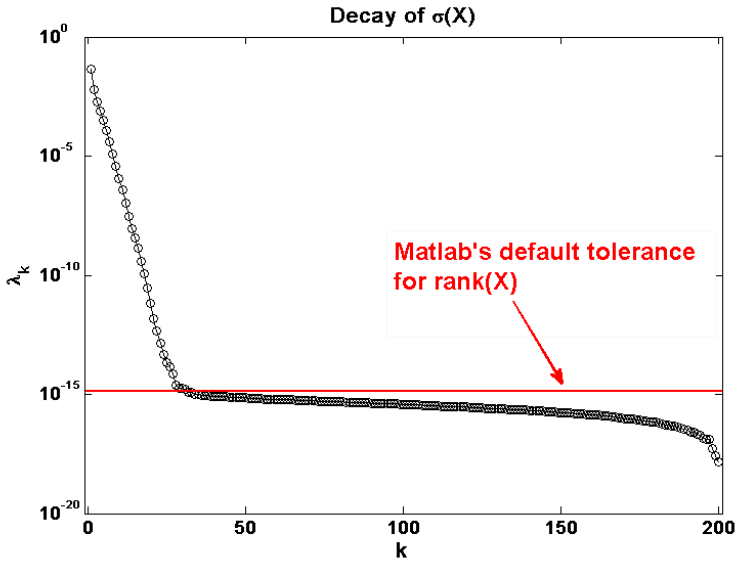


Figure 1: Decay of the eigenvalues of the DARE solution corresponding to the discrete heat equation example from [19] with $n = 200$, $m = p = 1$.

(the series converges because $|z_0| < 1$ for every $z_0 \in \sigma(A)$). Moreover, if all eigenvalues of A are in the open unit disk, then we have for the unique solution S of (10) $S \geq 0$ if $V \geq 0$.

From this we have, that for given matrices $A_k, E \in \mathbb{R}^{n\times n}$ the Stein equations (8) and (9) have a unique solution if and only if $\lambda_r\lambda_s \neq 1$ for any $\lambda_r, \lambda_s \in \sigma(A_k, E)$. Hence, as the Stein equation (8) (or (9), respectively) arises in the Newton iteration, Theorem 2.1a) guarantees here the existence of a unique symmetric solution of the Stein equation.

3.1 Smith's method

The Smith iteration [54] is derived from the symmetric Stein equation (10) and its representation (11)

$$S_0 = 0, \qquad S_{j+1} = V + AS_jA^T.$$

If all eigenvalues of A lie in the open unit disk, this iteration converges and the iterates can be written as

$$S_j = \sum_{i=0}^{j-1} A^i V A^{Ti}, \qquad j = 1, 2, \ldots.$$

Its rate of convergence is quite slow, in general. The accelerated version [54] — the so-called squared Smith method — is based on the recursion

$$S_0 = V, \qquad S_{j+1} = S_j + A^{2^j} S_j A^{T2^j}. \tag{12}$$

Its iterates can be written as

$$S_j = \sum_{i=1}^{2^j} A^{i-1} V A^{T(i-1)}.$$

Despite the quadratic convergence rate, one should be reluctant to apply the squared method to large, sparse equations. The matrices A^{2^j}, which have to be squared explicitly in each step of the iteration, are dense even if A is sparse.

We will discuss how to use the standard Smith iteration efficiently for solving the Stein equations (8) and (9) in the context of large, sparse equations. The idea behind this is based on a low-rank version of (12) as proposed, e.g., in [14].

The Smith iteration for (8) is given by

$$EY_{j+1}E^T = A_k Y_j A_k^T + \mathcal{R}_k, \tag{13}$$

while for (9) it is given by

$$EY_{j+1}E^T = A_k Y_j A_k^T + \mathcal{C}_k. \tag{14}$$

The last term on the right hand side of (14) can be expressed as

$$\begin{aligned} \mathcal{C}_k &= CQC^T - CS^T K_k^T - K_k S C^T + K_k R K_k^T \\ &= \begin{bmatrix} C & -K_k \end{bmatrix} \begin{bmatrix} Q & S^T \\ S & R \end{bmatrix} \begin{bmatrix} C^T \\ -K_k^T \end{bmatrix}. \end{aligned}$$

As $\left[\begin{smallmatrix} Q & S^T \\ S & R \end{smallmatrix}\right] \geq 0$, we can obtain a (Cholesky) factorization

$$\begin{aligned} &\begin{bmatrix} C & -K_k \end{bmatrix} \begin{bmatrix} Q & S^T \\ S & R \end{bmatrix} \begin{bmatrix} C^T \\ -K_k^T \end{bmatrix} \\ &= \begin{bmatrix} C & -K_k \end{bmatrix} \begin{bmatrix} L & 0 \\ SL^{-T} & W \end{bmatrix} \begin{bmatrix} L^T & L^{-1}S^T \\ 0 & W^T \end{bmatrix} \begin{bmatrix} C^T \\ -K_k^T \end{bmatrix} \\ &= \begin{bmatrix} CL & 0 \\ -K_k S L^{-T} & -K_k W \end{bmatrix} \begin{bmatrix} L^T C^T & -L^{-1}S^T K_k^T \\ 0 & -W^T K_k^T \end{bmatrix} = UU^T \end{aligned} \tag{15}$$

with $Q = LL^T$, and $WW^T = R - SL^{-1}L^{-T}S^T$. The latter expression is positive semidefinite due to Assumption 3. on page 4. Therefore, $UU^T \geq 0$. When the

iteration (14) is started with $Y_0 = 0$, then $Y_1 = E^{-1}UU^TE^{-T}$ and all subsequent iterates

$$Y_j = \sum_{i=0}^{j-1} (E^{-1}A_k)^i E^{-1}UU^TE^{-T}(A_k^TE^{-T})^i, \qquad j = 1, 2, \ldots$$

are symmetric positive semidefinite and converge to a positive semidefinite matrix Y_*. Based on the assumption that the spectrum of the positive semidefinite matrix Y_* decays to zero very rapidly, we can expect that Y_* can be written using a factorization ZZ^T for some $Z \in \mathbb{R}^{n\times r}, r \ll n$. Now, if we assume that $Y_j = Z_jZ_j^T$ for $\operatorname{rank}(Z_j) = r_j \ll n$, $Z_j \in \mathbb{R}^{n\times r_j}$ and observe that $\operatorname{rank}(UU^T + A_kY_jA_k^T) \leq m + p + r_j \ll n$, we see that we can solve (14) for a low-rank representation of the (Cholesky) factor of Y_j. In particular, we have

$$\begin{aligned}(EZ_{j+1})(Z_{j+1}^TE^T) &= (A_kZ_j)(Z_j^TA_k^T) + UU^T[A_kZ_j \;\; U]\begin{bmatrix}(Z_j^TA_k^T)\\ U^T\end{bmatrix}\\ &= MM^T. \end{aligned} \tag{16}$$

Hence,

$$EZ_{j+1} = [A_kZ_j \;\; U]$$

yields one possible representation of the solution Y_{j+1}. Thus, the Smith algorithm can be reformulated in terms of the (Cholesky) factor Z_j of Y_j. There is no need to compute Y_j at each iteration; only Z_j is needed.

Incorporating this Smith iteration into Algorithm 2 we obtain Algorithm 3.

Remark 3.1 *For $Z_0 = 0$ it is straightforward to see that*

$$Z_j = E^{-1}\left[(A_kE^{-1})^{j-1}U \;\; (A_kE^{-1})^{j-2}U \;\; \ldots \;\; (A_kE^{-1})U \;\; U\right].$$

If $Z_j \in \mathbb{R}^{n\times r_j}$, then M in (16) is in $\mathbb{R}^{n\times(r_j+p+m)}$ and, as $E \in \mathbb{R}^{n\times n}$, $Z_{j+1} \in \mathbb{R}^{n\times(r_j+p+m)}$. The dimension of Z_{j+1} will increase by $p+m$ in each iteration step. Hence, if $p+m$ is large and/or the convergence is slow, the number of columns of Z_{j+1} will easily reach unmanageable levels of memory requirements. But if Z_{j+1} is of low rank r_{j+1}, then we can approximate it as follows. The rank-revealing LQ decomposition [20, 29] of M yields

$$\Pi M = \begin{bmatrix}L_{11} & 0\\ L_{21} & \Omega\end{bmatrix} V, \qquad L_{11} \in \mathbb{R}^{r_{j+1}\times r_{j+1}}, L_{21} \in \mathbb{R}^{(n-r_{j+1})\times r_{j+1}},$$

where L_{11} is lower triangular, $V \in \mathbb{R}^{n\times(n+r_j)}$ is orthogonal, Π is a permutation, and $\Omega \approx 0$ can be regarded as negligible. If we partition V in the form

$$V = \begin{bmatrix}V_1\\ V_2\end{bmatrix}, \qquad V_1 \in \mathbb{R}^{r_{j+1}\times n},$$

Algorithm 3 Newton-Hewer-Smith Method for the DARE

Input: The coefficient matrices A, B, C, E, Q, R, S of the DARE (1), and a starting guess X_0 in terms of its (Cholesky) factor T_0, so that $\sigma(A - K(X_0)B, E) \subset \mathcal{D}_1(0)$ and $R + BX_0B^T > 0$.

Output: An approximate solution X_{k+1} of the DARE (1) in terms of its (Cholesky) factor T_{k+1}.

Compute the Cholesky factorization $\begin{bmatrix} Q & S^T \\ S & R \end{bmatrix} = LL^T$.

for $k = 0, 1, 2, \ldots$ **do**

1. $K_k \leftarrow K(X_k)$ (making use of the fact that $X_k = T_k T_k^T$).
2. $A_k \leftarrow A - K_k B$.
3. $U \leftarrow [C \quad -K_k]L$.
4. $Z_0 \leftarrow 0$.

for $j = 0, 1, 2, \ldots$ **do**

5. Solve for Z_{j+1} in $EZ_{j+1} = [A_k Z_j \quad U]$.

end for

6. $T_{k+1} \leftarrow Z_{j+1}$

end for

and set $\Omega = 0$, we obtain the full-rank approximation

$$\widetilde{M} = \Pi^T \begin{bmatrix} L_{11} \\ L_{21} \end{bmatrix} V_1 = \Pi \widetilde{L} V_1, \qquad \widetilde{L} \in \mathbb{R}^{n \times r_{j+1}}.$$

In any unitary invariant norm, we have

$$||M - \widetilde{M}|| = ||\Omega||.$$

Now we approximate

$$(EZ_{j+1})(Z_{j+1}^T E^T) \approx \widetilde{M}\widetilde{M}^T = \Pi^T \widetilde{L}\widetilde{L}^T \Pi.$$

Hence

$$EZ_{j+1} \approx \Pi^T \widetilde{L}$$

yields one possible approximate low-rank representation of the solution Y_{j+1} of (14) such that $Z_{j+1} \in \mathbb{R}^{n \times r_{j+1}}$.

Alternatively, as suggested for the Smith iteration in [30], a thin singular value decomposition [29] of M can be employed,

$$M = U \Sigma V^T,$$

where, with $\ell = r_j + p + m$, $V \in \mathbb{R}^{\ell \times \ell}, V^T V = I, U \in \mathbb{R}^{n \times \ell}$ has orthonormal columns and $\Sigma = \operatorname{diag}(\sigma_1, \ldots, \sigma_\ell) \in \mathbb{R}^{\ell \times \ell}, \sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_\ell$. Partition this decomposition as

$$M = [U_1 \quad U_2] \begin{bmatrix} \Sigma_1 & \\ & \Sigma_2 \end{bmatrix} \begin{bmatrix} V_1^T \\ V_2^T \end{bmatrix},$$

where $\Sigma_1 \in \mathbb{R}^{r_{j+1}\times r_{j+1}}, V_1 \in \mathbb{R}^{\ell\times r_{j+1}}, U_1 \in \mathbb{R}^{n\times r_{j+1}}$. In case Σ_2 is negligible, we can approximate M by

$$\widehat{M} = U_1\Sigma_1 V_1^T.$$

By the Schmidt-Eckart-Young-Mirsky-theorem we have

$$\|M - \widehat{M}\| = \sigma_{r_{j+1}+1}.$$

Now we approximate

$$(EZ_{j+1})(Z_{j+1}^T E^T) \approx \widehat{M}\widehat{M}^T = U_1\Sigma_1 V_1^T V_1\Sigma_1 U_1^T = U_1\Sigma_1^2 U_1^T.$$

Hence

$$EZ_{j+1} \approx U_1\Sigma_1$$

yields another possible approximate low-rank representation of the solution Y_{j+1} of (14) such that $Z_{j+1} \in \mathbb{R}^{n\times r_{j+1}}$.

Therefore, instead of computing the solution Y of (9) directly, we only compute an approximation to its low rank factor Z with $Y = ZZ^T$. In order to make use of this representation of the solution we have to modify Algorithm 3 by replacing Step 7. appropriately, see Algorithm 4. That is, we simply ignore the trailing part of L which is less than or equal to some given tolerance. While going from the jth to the $(j+1)$st step, the number of columns of Z_{j+1} generally does not increase. Indeed an increase will only occur if the rank of Z_{j+1} is larger than that of Z_j. In any case, there can be at most n additional columns added at any step which is the same as in the unmodified version.

Let us assume that $X_k = T_k T_k^T$ with $T_k \in \mathbb{R}^{n\times s_k}$. In order to obtain Z_{j+1} one has to

- compute the (Cholesky) factorization of $\begin{bmatrix} Q & S^T \\ S & R \end{bmatrix} \in \mathbb{R}^{(p+m)\times(p+m)}$, this requires $\mathcal{O}((p+m)^3/3)$ flops [29] but only needs to be computed once before the iteration is started;
- evaluate K_k, this involves five matrix-matrix products of matrices of size $m \times n$ and $n \times s_k$, $m \times s_k$ and $s_k \times m$ (for $R + (BT_k)(T_k^T B^T)$, re-using BT_k and exploiting symmetry), $s_k \times n$ and $n \times n$, $m \times s_k$ and $s_k \times n$ (for $(BT_k)(T_k^T A)$, re-using BT_k from the previous computation), and $m \times p$ and $p \times n$ (for SC^T, which should be precomputed), and a linear solve with a system matrix of size $m \times m$ and a right hand side of size $m \times n$ which requires $\mathcal{O}(2m^3/3)$ flops [29] plus forward and backward substitution;
- compute U, this involves one matrix-matrix product of matrices of size $n \times (p+m)$ and $(p+m) \times (p+m)$;
- compute $A_k Z_j = AZ_j - K_k(BZ_j)$ for $j = 0, 1, \ldots$, this involves three matrix-matrix products of matrices of size $n \times n$ and $n \times r_j$, $m \times n$ and $n \times r_j$, and $n \times m$ and $m \times r_j$ in each iteration step;

Algorithm 4 Low Rank Newton-Hewer-Smith Method for the DARE

Input: The coefficient matrices A, B, C, E, Q, R, S of the DARE (1), and a starting guess X_0 in terms of its (Cholesky) factor T_0, so that $\sigma(A - K(X_0)B, E) \subset \mathcal{D}_1(0)$ and $R + BX_0B^T > 0$.

Output: An approximate solution X_{k+1} of the DARE (1) in terms of its low rank (Cholesky) factor T_{k+1}.

Compute the Cholesky factorization $\begin{bmatrix} Q & S^T \\ S & R \end{bmatrix} = LL^T$.

for $k = 0, 1, 2, \ldots$ **do**

1. $K_k \leftarrow K(X_k)$ (making use of the fact that $X_k = T_kT_k^T$).
2. $A_k \leftarrow A - K_kB$.
3. $\mathcal{C}_k \leftarrow \mathcal{C}(X_k)$.
4. $U \leftarrow [C \quad -K_k]L$.
5. $Z_0 \leftarrow 0$.

for $j = 0, 1, 2, \ldots$ **do**

6. Compute the *RRLQ* factorization $\Pi[A_kZ_j \;\; U] = \begin{bmatrix} L_{11} & 0 \\ L_{21} & \Omega \end{bmatrix} V$,

$$L_{11} \in \mathbb{R}^{r_{j+1} \times r_{j+1}}, V = \begin{bmatrix} V_1 \\ V_2 \end{bmatrix}, V_1 \in \mathbb{R}^{r_{j+1} \times n}$$

7. $\widetilde{L} \leftarrow \begin{bmatrix} L_{11} \\ L_{12} \end{bmatrix}$.
8. Solve for Z_{j+1} in $EZ_{j+1} = \Pi\widetilde{L}$.

end for

9. $T_{k+1} \leftarrow Z_{j+1}$

end for

- compute the rank-revealing LQ decomposition of $M = [A_k Z_j \quad U]$, this requires $4n(m+p+r_j)r_{j+1} - 2(n+m+p+r_j)r_{j+1}^2 + \frac{4}{3}r_{j+1}^3$ flops (see, e.g., [29, Section 5.4.1]) noting that Q need not be accumulated;
- solve $EZ_{j+1} = \Pi\widetilde{L}$ for Z_{j+1} for $j = 0, 1, \ldots$, this involves a (sparse) linear solve with a system matrix of size $n \times n$ and a right hand side of size $n \times r_{j+1}$. Please note, that the (sparse) LU decomposition of E needs to be computed only once, the rest of the computations are solely forward and backward solves. In the case of a DARE coming from a discretized PDE control problem, E will often be positive definite so that a (sparse) Cholesky decomposition can be employed and the factorization cost can be halved.

Hence, the overall flop count for the computation of Z_{j+1} depends mainly on the choice of the solver for the linear systems in the outer loop as well as on the cost of the rank-revealing LQ decomposition in the inner loop and the pre-computable (Cholesky) factorizations.

The iteration (13) does not allow a factored representation of the solution Y_{j+1} as $\mathcal{R}_k$ might be indefinite. But (in contrast to the continuous-time case) $\mathcal{R}_k$ can be split explicitly into its positive and negative semidefinite part $\mathcal{P}_k$ and $\mathcal{N}_k$, resp.,

$$\mathcal{R}_k = \mathcal{P}_k - \mathcal{N}_k$$

with

$$\begin{aligned} \mathcal{P}_k &= CQC^T + AX_kA^T, \\ \mathcal{N}_k &= EX_kE^T + (AX_kB^T + CS^T)(R + BX_kB^T)^{-1}(BX_kA^T + SC^T). \end{aligned}$$

Splitting the iterates into their positive and negative semidefinite parts Y_j^P and Y_j^N, resp.,

$$Y_j = Y_j^P - Y_j^N$$

we have

$$E(Y_{j+1}^P - Y_{j+1}^N)E^T = A_k(Y_j^P - Y_j^N)A_k^T + (\mathcal{P}_k - \mathcal{N}_k),$$

that is, we can iterate on the positive and negative semidefinite parts separately

$$EY_{j+1}^P E^T = A_k Y_j^P A_k^T + \mathcal{P}_k, \tag{17}$$

$$EY_{j+1}^N E^T = A_k Y_j^N A_k^T + \mathcal{N}_k. \tag{18}$$

Now, with $Q = LL^T$ and $X_k = T_k T_k^T, T_k \in \mathbb{R}^{n \times s_k}$, we have

$$\mathcal{P}_k = CQC^T + AX_kA^T = CLL^TC^T + AT_kT_k^TA^T = [CL \quad AT_k] \begin{bmatrix} L^TC^T \\ T_k^TA^T \end{bmatrix}.$$

Writing $Y_j^P = Z_j^P (Z_j^P)^T$ we obtain

$$EZ_{j+1}^P (Z_{j+1}^P)^T E^T = \begin{bmatrix} A_k Z_j^P & CL & AT_k \end{bmatrix} \begin{bmatrix} (Z_j^P)^T A_k^T \\ L^T C^T \\ T_k^T A^T \end{bmatrix}.$$

Hence,

$$EZ_{j+1}^P = \begin{bmatrix} A_k Z_j^P & CL & AT_k \end{bmatrix}$$

is one possible representation of the (Cholesky) factor Z_{j+1}^P of the positive semidefinite part Y_{j+1}^P of Y_{j+1}. If $Z_j^P \in \mathbb{R}^{n \times r_j^P}$, then $EZ_{j+1}^P \in \mathbb{R}^{n \times (r_j^P + p + s_k)}$ and, as $E \in \mathbb{R}^{n \times n}$, $Z_{j+1}^P \in \mathbb{R}^{n \times (r_j^P + p + s_k)}$. The dimension of Z_{j+1}^P would increase in each iteration step. But if Z_{j+1}^P is of low rank r_{j+1}^P, then we can approximate it using a rank-revealing LQ factorization as before.

Moreover, with $X_k = T_k T_k^T$ and $R + BX_k B^T = S_k S_k^T$ we have

$$\begin{aligned} \mathcal{N}_k &= EX_k E^T + (AX_k B^T + CS^T)(R + BX_k B^T)^{-1}(BX_k A^T + SC^T) \\ &= EX_k E^T + (AX_k B^T + CS^T) S_k^{-T} S_k^{-1} (BX_k A^T + SC^T) \\ &= \begin{bmatrix} ET_k & (AX_k B^T + CS^T) S_k^{-T} \end{bmatrix} \begin{bmatrix} T_k^T E^T \\ S_k^{-1}(BX_k A^T + SC^T) \end{bmatrix}. \end{aligned}$$

From this, we obtain

$$EY_{j+1}^N E^T = \begin{bmatrix} A_k Z_j^N & ET_k & (AX_k B^T + CS^T) S_k^{-T} \end{bmatrix} \begin{bmatrix} (Z_j^N)^T A_k^T \\ T_k^T E^T \\ S_k^{-1}(BX_k A^T + SC^T) \end{bmatrix}.$$

Hence, with $Y_j^N = Z_j^N (Z_j^N)^T$

$$EZ_{j+1}^N = \begin{bmatrix} A_k Z_j^N & ET_k & (AX_k B^T + CS^T) S_k^{-T} \end{bmatrix}$$

is one possible representation of a factor Z_{j+1}^N of the negative semidefinite part Y_{j+1}^N of Y_{j+1}. If $Z_j^N \in \mathbb{R}^{n \times r_j^N}$, then $EZ_{j+1}^N \in \mathbb{R}^{n \times (r_j^N + m + s_k)}$ and, as $E \in \mathbb{R}^{n \times n}$, $Z_{j+1}^N \in \mathbb{R}^{n \times (r_j^N + m + s_k)}$. The dimension of Z_{j+1}^N would increase in each iteration step. But if Z_{j+1}^N is of low rank r_{j+1}^N, then we can approximate it using a rank-revealing LQ factorization as before.

From Z_{j+1}^P and Z_{j+1}^N, one can determine $Y_{j+1} = Z_{j+1}^P (Z_{j+1}^P)^T - Z_{j+1}^N (Z_{j+1}^N)^T$. The overall flop count for this computation of Y_{j+1} is higher than the one for the iteration (14). As for (14), K_k and A_k have to be set up. In case this is done in a reasonable way, $S_k^{-1}(BX_k A^T + SC^T)$ is computed as by-product when setting up K_k. Instead of the (Cholesky) factorization of $\begin{bmatrix} Q & S^T \\ S & R \end{bmatrix} \in \mathbb{R}^{(p+m)\times(p+m)}$, here only the factorization of $Q = LL^T \in \mathbb{R}^{p \times p}$ is needed. While before, $A_k Z_j$

was computed, we now need $A_k Z_j^P$, $A_k Z_j^N$, AT_k and ET_k. Moreover, two instead of one rank-revealing QL factorizations have to be computed and two linear systems with the system matrix E have to be solved (this can be done using one factorization of E).

As the solution Y_{j+1} is constructed from two different iterations, one should use some kind of defect correction in order to increase the accuracy of the solution and to generate the (Cholesky) factor of Y_{j+1}.

The first idea that might come to mind is to use the following result from [45].

Theorem 3.2 *Let X be an Hermitian solution of*

$$0 = \mathcal{R}(X),$$

and let $\widetilde{X}$ be an Hermitian approximation to X. Let $V = X - \widetilde{X}$. If $\widetilde{R} = R + B\widetilde{X}B^T$ and $I + VB^T\widetilde{R}^{-1}B$ are nonsingular, then V satisfies the equation

$$0 = \mathcal{R}(\widetilde{X}) - EVE^T + \widetilde{A}V\widetilde{A}^T - (\widetilde{A}VB^T)^T(\widetilde{R} + BVB^T)^{-1}(\widetilde{A}VB^T)$$

where

$$\widetilde{A}^T = -B^T\widetilde{R}^{-1}CS^T + (I - B^T\widetilde{R}^{-1}B\widetilde{X})A^T.$$

Unfortunately, the resulting algorithm can not be used here, as in each iteration step, one has to solve (1). Using the approach discussed above, this would lead to a solution in terms of (Cholesky) factors of the positive and the negative semidefinite part of the solution and not to a (Cholesky) factor of the solution itself.

A different option might be to make use of the following observation which allows us to solve the DARE in each iteration step so that the desired (Cholesky) factor of the solution itself is computed. A "formidable computation" [39, p.312] shows that the DARE (1) can be rewritten as

$$EYE^T - \breve{A}Y\breve{A}^T = \breve{Q} + (CS^T - K(Y)R)R^{-1}(CS^T - K(Y)R)^T \qquad (19)$$

with

$$\begin{aligned} \breve{A} &= A - K(Y)B \\ \breve{Q} &= CQC^T - CS^TR^{-1}SC^T = C\left(Q - S^TR^{-1}S\right)C^T \end{aligned}$$

and $K(Y)$ as in (3) as a symmetric Stein equation. As $Q - S^TR^{-1}S$ is the Schur complement of R in the positive semidefinite matrix $\begin{bmatrix} Q & S^T \\ S & R \end{bmatrix}$ [34], $\breve{Q}$ is symmetric positive semidefinite. Now, denote the solution of (1) obtained by the

Smith iteration (17)–(18) by Y^*. Fix the right hand side of (19) by evaluating $K(Y^*) = K^*$,

$$EYE^T - \breve{A}Y\breve{A}^T = \breve{Q} + (CS^T - K^*R)R^{-1}(CS^T - K^*R)^T. \tag{20}$$

We will use this as a defect correction and update our approximate solution Y^* by $\breve{Y}$ which is the solution of

$$E\breve{Y}E^T = \breve{A}\breve{Y}\breve{A}^T + \breve{Q} + (CS^T - K^*R)R^{-1}(CS^T - K^*R)^T. \tag{21}$$

This may be done by the Smith iteration (14) where we substitute A_k by $\breve{A}$ and $\mathcal{C}_k$ by $\mathcal{C}_k^* = \breve{Q} + (CS^T - K^*R)R^{-1}(CS^T - K^*R)^T$.

A disadvantage of the Smith iteration is that its convergence is linear and the rate is bounded by $\rho_k = \max\{|\lambda|, \lambda \in \sigma(A_k, E)\}$. As ρ_k can be close to 1, this may be very slow. The ADI iteration discussed next offers an alternative.

3.2 ADI Iteration

The alternating direction implicit (ADI) iteration was first introduced [47] to solve linear systems arising from the discretization of elliptic boundary value problems. In general, the ADI iteration is used to solve linear systems of the form

$$My = b,$$

where M is symmetric positive definite and can be split into the sum of two positive definite matrices $M = M_1 + M_2$ for which the following iteration is efficient:

$$\begin{aligned} y_0 &= 0, \\ (M_1 + \mu_j I)y_{j-1/2} &= b - (M_2 - \mu_j I)y_{j-1}, \\ (M_2 + \eta_j I)y_j &= b - (M_1 - \eta_j I)y_{j-1/2}, \end{aligned}$$

for $j = 1, 2, 3, \ldots$. The ADI shift parameters μ_j and η_j are determined from spectral bounds on M_1 and M_2 to improve the convergence rate.

In [17], the ADI iteration is applied to the equation

$$X - \widehat{A}X\widehat{B} = \widehat{C}, \qquad \widehat{A} \in \mathbb{R}^{s\times s}, \widehat{B} \in \mathbb{R}^{t\times t}, \widehat{C} \in \mathbb{R}^{s\times t}. \tag{22}$$

Let X_0 be an initial approximate solution of this equation. Then the ADI iteration generates new iterates X_{M+N} as follows

$$M := M + 1; \qquad X_{M+N}(I - \mu_M\widehat{B}) = (\widehat{A} - \mu_M I)X_{M+N-1}\widehat{B} + \widehat{C}, \tag{23}$$

$$N := N + 1; \qquad (I - \eta_N\widehat{A})X_{M+N} = \widehat{A}X_{M+N-1}(\widehat{B} - \eta_N I) + \widehat{C}, \tag{24}$$

where one starts with $M = N = 0$. Either (23) or (24) can be used to determine X_{M+N} from X_{M+N-1}. When (23) and (24) are used in a strictly alternating fashion, the ADI method obtained is analogous to the "classical" ADI iteration for Sylvester's equation discussed by Wachspress [57]. In [17], strict alternation between the formulas (23) and (24) is not required as, e.g., the structures and sizes of the matrices $\widehat{A}$ and $\widehat{B}$ may make the computation of X_{M+N} in one of the equations, say (23) faster than by the other one. In this case the computational effort to solve (22) may be reduced by applying (23) more often than (24), even if such an approach would result in slightly lower convergence than strict alternation. The convergence of the iteration (23)–(24) for different ratios M/N is analyzed in [17] as well as the choice of the iteration parameters. There is still the problem of storing the usually dense $n \times n$ matrix X_{M+N}. This storage can be avoided by observing that, for the problems under consideration, the spectrum of the positive semidefinite matrix $X_{M+N} = Z_{M+N} Z_{M+N}^T$ often decays to zero rapidly. Here Z_{M+N} can be considered as a Cholesky factor of X_{M+N}. We expect that X_{M+N} can be approximated accurately by a factorization $\widetilde{Z}\widetilde{Z}^T$ for some $\widetilde{Z} \in \mathbb{R}^{n \times r}$ with $r \ll n$.

Here, we will restrict our discussion to a strict alternation between the two equations. With $\widehat{A} = E^{-1}A_k$, $\widehat{B} = A_k^T E^{-T}$, and $\widehat{C} = E^{-1}\mathcal{R}_k E^{-T}$, (22) is equivalent to (8) and we obtain from (23)

$$X_{i-\frac{1}{2}}(I - \mu_i A_k^T E^{-T}) = (E^{-1}A_k - \mu_i I)X_{i-1}A_k^T E^{-T} + E^{-1}\mathcal{R}_k E^{-T},$$

while (24) yields

$$(I - \eta_i E^{-1}A_k)X_i = E^{-1}A_k X_{i-\frac{1}{2}}(A_k^T E^{-T} - \eta_i I) + E^{-1}\mathcal{R}_k E^{-T}.$$

In order to get rid of the inverses of E, one can manipulate both equations in an obvious way to reach

$$\begin{aligned} EX_{i-\frac{1}{2}}(E^T - \mu_i A_k^T) &= (A_k - \mu_i E)X_{i-1}A_k^T + \mathcal{R}_k, \\ (E - \eta_i A_k)X_i E^T &= A_k X_{i-\frac{1}{2}}(A_k^T - \eta_i E^T) + \mathcal{R}_k. \end{aligned} \tag{25}$$

Similarly, with $\widehat{A} = E^{-1}A_k$, $\widehat{B} = A_k^T E^{-T}$, and $\widehat{C} = E^{-1}\mathcal{C}_k E^{-T}$, (22) is equivalent to (9) and we obtain

$$\begin{aligned} EX_{i-\frac{1}{2}}(E^T - \mu_i A_k^T) &= (A_k - \mu_i E)X_{i-1}A_k^T + E\widehat{C}E^T, \\ (E - \eta_i A_k)X_i E^T &= A_k X_{i-\frac{1}{2}}(A_k^T - \eta_i E^T) + E\widehat{C}E^T. \end{aligned} \tag{26}$$

While $\mathcal{R}_k$ in (25) may be indefinite, $E^T\widehat{C}E$ in (26) is positive semidefinite, see (15).

In order to derive a formulation in terms of (Cholesky) factors, let us rewrite the iteration (26) in a single equation. In particular, we have

$$\begin{aligned} X_{i-\frac{1}{2}}(I-\mu_i A_k^T E^{-T}) &= (E^{-1}A_k-\mu_i I)X_{i-1}A_k^T E^{-T}+\widehat{C}, \\ (I-\eta_i E^{-1}A_k)X_i &= E^{-1}A_k X_{i-\frac{1}{2}}(A_k^T E^{-T}-\eta_i I)+\widehat{C}. \end{aligned}$$

Hence, with

$$X_{i-\frac{1}{2}} = (E^{-1}A_k-\mu_i I)X_{i-1}A_k^T E^{-T}(I-\mu_i A_k^T E^{-T})^{-1}+\widehat{C}(I-\mu_i A_k^T E^{-T})^{-1},$$

we have

$$\begin{aligned} X_i &= (I-\eta_i E^{-1}A_k)^{-1}\left(E^{-1}A_k X_{i-\frac{1}{2}}(A_k^T E^{-T}-\eta_i I)+\widehat{C}\right) \\ &= (I-\eta_i E^{-1}A_k)^{-1}\cdot \\ &\quad \Big(E^{-1}A_k(E^{-1}A_k-\mu_i I)X_{i-1}A_k^T E^{-T}(I-\mu_i A_k^T E^{-T})^{-1}(A_k^T E^{-T}-\eta_i I) \\ &\quad + E^{-1}A_k\widehat{C}(I-\mu_i A_k^T E^{-T})^{-1}(A_k^T E^{-T}-\eta_i I)+\widehat{C}\Big). \end{aligned}$$

As

$$(I-\mu_i A_k^T E^{-T})^{-1}(A_k^T E^{-T}-\eta_i I) = (A_k^T E^{-T}-\eta_i I)(I-\mu_i A_k^T E^{-T})^{-1}$$

we have

$$\begin{aligned} X_i &= (I-\eta_i E^{-1}A_k)^{-1}\Big(E^{-1}A_k(E^{-1}A_k-\mu_i I)X_{i-1}A_k^T E^{-T}(A_k^T E^{-T}-\eta_i I) \\ &\quad + E^{-1}A_k\widehat{C}(A_k^T E^{-T}-\eta_i I)+\widehat{C}(I-\mu_i A_k^T E^{-T})\Big)(I-\mu_i A_k^T E^{-T})^{-1}. \end{aligned}$$

Obviously, in order to obtain a symmetric X_i we have to restrict the choice of μ_i and η_i such that $\eta_i = \overline{\mu}_i$,

$$\begin{aligned} X_i &= (I-\overline{\mu}_i E^{-1}A_k)^{-1}\Big(E^{-1}A_k(E^{-1}A_k-\mu_i I)X_{i-1}(A_k^T E^{-T}-\overline{\mu}_i I)A_k^T E^{-T} \\ &\quad + E^{-1}A_k\widehat{C}(A_k^T E^{-T}-\overline{\mu}_i I)+\widehat{C}(I-\mu_i A_k^T E^{-T})\Big)(I-\mu_i E^{-1}A_k)^{-T}. \end{aligned}$$

Observe that

$$\begin{aligned} &E^{-1}A_k\widehat{C}(A_k^T E^{-T}-\overline{\mu}_i I)+\widehat{C}(I-\mu_i A_k^T E^{-T}) \\ &\quad = E^{-1}A_k\widehat{C}A_k^T E^{-T}-\overline{\mu}_i E^{-1}A_k\widehat{C}+\widehat{C}-\mu_i\widehat{C}A_k^T E^{-T} \\ &\quad = E^{-1}A_k\widehat{C}A_k^T E^{-T}+(I-\overline{\mu}_i E^{-1}A_k)\widehat{C}(I-\mu_i A_k^T E^{-T}) \\ &\qquad -|\mu_i|^2 E^{-1}A_k\widehat{C}A_k^T E^{-T} \\ &\quad = (1-|\mu_i|^2)E^{-1}A_k\widehat{C}A_k^T E^{-T}+(I-\overline{\mu}_i E^{-1}A_k)\widehat{C}(I-\mu_i A_k^T E^{-T}). \end{aligned}$$

Algorithm 5 ADI-Newton's Method for the DARE

Input: The coefficient matrices A, B, C, E, Q, R, S of the DARE (1), and a starting guess X_0, so that $\sigma(A - K(X_0)B, E) \subset \mathcal{D}_1(0)$ and $R + BX_0B^T > 0$.
Output: An approximate solution X_{k+1} of the DARE (1).

for $k = 0, 1, 2, \ldots$ **do**
1. $K_k \leftarrow K(X_k)$.
2. $A_k \leftarrow A - K_kB$.
3. $\mathcal{C}_k \leftarrow \mathcal{C}(X_k)$.
4. $Y_0 \leftarrow 0$.

for $j = 1, 2, \ldots$ **do**
5. Choose shift μ_j
6. $Y_j \leftarrow E^{-1}\mathcal{C}_kE^{-T}$.
7. $H \leftarrow (1 - |\mu_j|^2)E^{-1}A_kY_jA_k^TE^{-T}$.
8. $H \leftarrow H + E^{-1}A_k(E^{-1}A_k - \mu_jI)Y_{j-1}(A_k^TE^{-T} - \overline{\mu}_jI)A_k^TE^{-T}$.
9. $Y_j \leftarrow Y_j + (I - \overline{\mu}_jE^{-1}A_k)^{-1}H(I - \mu_jE^{-1}A_k)^{-T}$.

end for
10. $X_k \leftarrow Y_j$.

end for

Hence, we obtain the single equation

$$\begin{aligned} X_i &= \widehat{C} + (I - \overline{\mu}_iE^{-1}A_k)^{-1}\Big((1 - |\mu_i|^2)E^{-1}A_k\widehat{C}A_k^TE^{-T} + \\ &\quad E^{-1}A_k(E^{-1}A_k - \mu_iI)X_{i-1}(A_k^TE^{-T} - \overline{\mu}_iI)A_k^TE^{-T}\Big)(I - \mu_iE^{-1}A_k)^{-T}. \end{aligned} \tag{27}$$

If X_{i-1} is positive semidefinite, then so is X_i.

As

$$(A_k^TE^{-T} - \overline{\mu}_iI)A_k^TE^{-T}(I - \mu_iE^{-1}A_k)^{-T} = (A_k^TE^{-T} - \overline{\mu}_iI)(E^TA_k^{-T} - \mu_iI)^{-1},$$

the spectral radius

$$\rho_{ADI} = \rho\left(\prod_{j=1}^{\ell}(A_k^{-1}E - \mu_iI)^{-1}(E^{-1}A_k - \overline{\mu}_iI)\right)$$

determines the rate of convergence, where ℓ is the number of iteration steps and shifts used. The minimization of ρ_{ADI} with respect to the shift parameters μ_i is given by

$$\min_{\mu_1,\ldots,\mu_\ell}\max_{\lambda\in\sigma(E^{-1}A_k)}\frac{|(\lambda - \overline{\mu}_1)\cdots(\lambda - \overline{\mu}_\ell)|}{|(\frac{1}{\lambda} - \mu_1)\cdots(\frac{1}{\lambda} - \mu_\ell)|}. \tag{28}$$

In [17] the choice of shifts for the more general ADI iteration (23), (24) applied to (22) is discussed. Although here we are considering a special case of that

situation, the proposed choice of the shifts can not be carried over directly, as the shifts μ_M and η_N in (23) and (24) are not related to each other, while we require $\mu_j = \overline{\eta_j}$ in order to obtain a symmetric solution from a symmetric starting guess.

The choice of the shift parameters μ_i is related to a rational approximation problem. The conventional approach to the computation of these parameters in other ADI settings is to cover the spectrum by a domain $\Omega \in \mathbb{C}$ and to solve the minmax problem with respect to Ω instead of the spectrum. This approximation theory based approach generally requires the knowledge of certain bounds of the spectrum. Heuristic approaches have also been proposed. See, e.g., [18, 49] and the references therein.

In Section 4 we propose to use a quite simple, numerically inexpensive, heuristic algorithm which replaces the spectrum by some approximations to the largest and the smallest eigenvalues in the spectrum $\sigma(E^{-1}A_k)$. Assume that the largest ℓ and the smallest $\jmath$ eigenvalues have been approximated, $\mathcal{S} = \{\widetilde{\lambda}_1, \ldots, \widetilde{\lambda}_\ell, \widetilde{\lambda}_{n-\jmath+1}, \ldots, \widetilde{\lambda_n}\}$. Then the minmax problem is replaced by

$$\min_{\mu_1,\ldots,\mu_\ell} \max_{\lambda\in\mathcal{S}} \frac{|(\lambda-\overline{\mu}_1)\cdots(\lambda-\overline{\mu}_\ell)|}{|(\frac{1}{\lambda}-\mu_1)\cdots(\frac{1}{\lambda}-\mu_\ell)|}. \tag{29}$$

Next a heuristic optimization method is employed in order to compute suitable shifts. For an efficient solution of large-scale DAREs, the solution of (28) will require further studies in the future.

Using (15) we can write

$$\widehat{C} = \left(E^{-1}U\right)\left(U^TE^{-T}\right) = MM^T, \qquad M \in \mathbb{R}^{n\times(p+m)}.$$

Finally, with $|\mu_i| \leq 1$, $X_i = Z_iZ_i^T$, $Z_i \in \mathbb{R}^{n\times r_j}$ and

$$\begin{aligned}\sqrt{1-|\mu_i|^2}(I-\overline{\mu}_iE^{-1}A_k)^{-1}E^{-1}A_kM &= \widehat{M},\\ (I-\overline{\mu}_iE^{-1}A_k)^{-1}E^{-1}A_k(E^{-1}A_k-\mu_iI)Z_{i-1} &= \widetilde{M},\end{aligned}$$

we have

$$\begin{aligned}\sqrt{1-|\mu_i|^2}(E-\overline{\mu}_iA_k)^{-1}A_kM &= \widehat{M} \in \mathbb{R}^{n\times(p+m)},\\ (E-\overline{\mu}_iA_k)^{-1}A_kE^{-1}(A_k-\mu_iE)Z_{i-1} &= \widetilde{M} \in \mathbb{R}^{n\times r_{j-1}}.\end{aligned}$$

With this we can write (27) in terms of a factorization

$$X_i = Z_iZ_i^T = MM^T + \widehat{M}\widehat{M}^T + \widetilde{M}\widetilde{M}^T = \begin{bmatrix} M & \widehat{M} & \widetilde{M}\end{bmatrix}\begin{bmatrix} M^T\\ \widehat{M}^T\\ \widetilde{M}^T\end{bmatrix}.$$

Hence,

$$Z_i = \begin{bmatrix} M & \widehat{M} & \widetilde{M}\end{bmatrix} \in \mathbb{R}^{n\times(2(p+m)+r_j)}$$

is one possible representation of the (Cholesky) factor of the solution X_i. The dimension of Z_j would increase by $2(p+m)$ in each iteration step. But if Z_j is of low rank r_j, then we can approximate it using a rank-revealing LQ factorization as before. The derivation of the algorithm is straightforward, hence we omit its statement.

4 Numerical Examples

All numerical tests were done in MATLAB R2009a on a Pentium M notebook with 512 MB main memory. The iterations were stopped in all algorithms as soon as $||Z_{j+1}Z_{j+1}^T - Z_jZ_j^T||_F/||Z_{j+1}Z_{j+1}^T||_F$ and $||T_{k+1}T_{k+1}^T - T_kT_k^T||_F/||T_{k+1}T_{k+1}^T||_F$ were less than a given tolerance.

The examples considered are optimal control problems of the following form:

$$\min \mathcal{J}(u), \quad \mathcal{J}(u) = \frac{1}{2}\sum_{k=0}^{\infty} y_k^T Q y_k + u_k^T R u_k + 2y_k^T S u_k$$

subject to a fully discretized parabolic PDE constraint

$$\begin{aligned} Ex_{k+1} &= Ax_k + Bu_k, && (30)\\ y_k &= Cx_k. && (31) \end{aligned}$$

In all examples we have $m = p = 1$ and choose $R = 1, Q = 1, S = 0$.

In order to compare the computed solutions to an 'exact' one, the MATLAB routine `dare` was used to produce the "exact" solution X_{exact}.

In our examples we used only one shift for the ADI iteration. The minmax problem (28) is replaced by

$$\min_{\mu} \max_{\lambda \in \mathcal{S}} \frac{|\lambda - \overline{\mu}|}{|\frac{1}{\lambda} - \mu|}, \tag{32}$$

where $\mathcal{S}$ is either the set of all eigenvalues computed by `eig` or eigenvalue approximations computed by `eigs`. Assume that the largest ℓ and the smallest $\jmath$ eigenvalues have been approximated so that $\mathcal{S} = \{\widetilde{\lambda}_1, \ldots, \widetilde{\lambda}_\ell, \widetilde{\lambda}_{n-\jmath+1}, \ldots, \widetilde{\lambda_n}\}$. Next, `fminsearch` is employed as a heuristic optimization method in order to compute a suitable shift which we simply call μ_* in the following. The standard setting for all examples presented is $\ell = 8, j = 2$. Moreover, `eigs` was called with `OPTS.tol =` 10^{-2} as more accuracy in this computation does not increase the accuracy of the computed solution or convergence rate of the algorithm. Note that an optimization with respect to the full spectral set did in no case improve

the convergence of the overall algorithms, i.e., the computed $\ell + j$ Ritz values were sufficient to obtain a suitable shift. Thus, we only report results based on $\mathcal{S}$ being composed of the computed $\ell + j$ Ritz values. When further developing the algorithms proposed in this paper, a multi-shift selection strategy needs to be developed. A first candidate for this would be Penzl's heuristic [49] that often yields good results in the ADI iteration for continuous Lyapunov equations.

Example 4.1 The data of this example come from the autonomous linear-quadratic optimal control problem of one dimensional heat flow and is taken from [1, Example 4.2]. The model equations are

$$\begin{aligned} \frac{\partial}{\partial t}x(t,\eta) &= \alpha\frac{\partial^2}{\partial \eta^2}x(t,\eta) + b(\eta)u(t) \qquad \eta \in (0,1); t > 0 \\ x(t,0) &= 0 = x(t,1) \qquad\qquad t > 0 \\ x(0,\eta) &= x_0(\eta) \qquad\qquad \eta \in (0,1) \\ y(x) &= \int_0^1 c(\eta)x(t,\eta)d\eta, \qquad\qquad x > 0, \end{aligned}$$

where $\alpha > 0$ and $x_0, b, c \in L_2(0,1)$. Using a standard linear finite element (FE) discretization in space, one obtains an ordinary differential equation

$$M_N\dot{x}(t) = K_N x(t) + Fu(t), \quad y(t) = Cx(t),$$

where

$$M_N = \frac{1}{6N}\begin{bmatrix} 4 & 1 & & & \\ 1 & 4 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & 4 & 1 \\ & & & 1 & 4 \end{bmatrix}, \; K_N = -\alpha N\begin{bmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \\ & & & -1 & 2 \end{bmatrix},$$

$F = b_N$ and $C = c_N^T$, $M_N, K_N \in \mathbb{R}^{N\times N}$. The vectors b_N and c_N are obtained from the L_2 inner products of indicator functions for subsets of $(0,1)$ with the FE basis functions $\{\phi_i^N\}_{i=1}^{N-1}$:

$$(b_N)_j = \int_0^1 \beta(s)\phi_i^N(s)ds, \quad \text{and} \quad (c_N)_j = \int_0^1 \gamma(s)\phi_i^N(s)ds, \quad j = 1,\ldots,N-1,$$

where the functions $\beta, \gamma \in L_2(0,1)$ are given by

$$\beta(s) = \gamma(s) = \begin{cases} 1, & s \in [0.1, 0.5], \\ 0, & \text{otherwise.} \end{cases}$$

(Note: β, γ are parameter-dependent in [1] and may differ — here we chose them equal for simplicity.) Employing the semi-implicit Euler method with stepsize Δt yields the difference equation

$$(M - \Delta t K)x_{k+1} = Mx_k + \Delta t F u_k,$$

that is, $A = M$, $E = M - \Delta t K$, and $B = \Delta t F$ in (30).

In the computational experiments reported in Tables 1 – 2, we set $\alpha = 0.05$ (default in [1]). `fminsearch` was started with different starting guesses μ_0. The resulting shift is given as μ_*. Different n, tol and h have been considered. Here only the results for $n = 1000$, $h = 0.1$ and $h = 0.01$ as well as $tol = 10^{-8}$ and $tol = 10^{-12}$ are presented. Here and in the following, $\rho(A, E)$ denotes the spectral radius of $A - \lambda E$, i.e., $\rho(A, E) = \max_{\lambda \in \sigma(A,E)}\{|\lambda|\}$.

No matter how the parameters were chosen, all methods needed 4 Newton steps. The number of Smith iteration steps was usually quite high, about twice as many as needed for the unshifted ADI iteration. Obviously, neither the Smith iteration nor the unshifted ADI iteration yield feasible methods for solving the DARE with Newton's method in this example. On the other hand, using our fairly simple heuristic for choosing a single shift, the ADI iteration with shift converges up to 100 times faster than the ADI iteration without a shift. The number of iterations needed varied from Newton step to Newton step, being larger at first.

Changing n while keeping the rest of the parameters fixed, does not change the number of iterations needed for convergence or the accuracy achieved.

$\Delta t = 0.1$, $\rho(A, E) \approx 0.9902$				
	X_{dare}	Z_{smith}	Z_{adi} $\mu_* = 0.96328$	Z_{adi} no shift
Newton steps		4	4	4
Smith/ADI steps		711–719	17–18	374–378
$\mathcal{R}(X)$	$2.2 \cdot 10^{-12}$	$7.1 \cdot 10^{-12}$	$2.1 \cdot 10^{-13}$	$3.5 \cdot 10^{-12}$
$\|\|X_{dare} - X\|\|_F / \|\|X\|\|_F$		$4.9 \cdot 10^{-7}$	$4.6 \cdot 10^{-9}$	$2.4 \cdot 10^{-7}$
$\mathrm{rank}(ZZ^T)$	13	11	11	11
$\Delta t = 0.01$, $\rho(A, E) \approx 0.999$				
	X_{dare}	Z_{smith}	Z_{adi} $\mu_* = 0.99306$	Z_{adi} no shift
Newton steps		4	4	4
Smith/ADI steps		5926–5990	31–35	3138–3172
$\mathcal{R}(X)$	$2.3 \cdot 10^{-13}$	$7.2 \cdot 10^{-11}$	$1.3 \cdot 10^{-12}$	$3.6 \cdot 10^{-11}$
$\|\|X_{dare} - X\|\|_F / \|\|X\|\|_F$		$5.0 \cdot 10^{-6}$	$1.1 \cdot 10^{-8}$	$2.5 \cdot 10^{-6}$
$\mathrm{rank}(ZZ^T)$	16	16	16	16

Table 1: Example 1, $tol = 10^{-8}$, $n = 1000$, $\mu_0 = -1$.

Example 4.2 Here, we use again a discretized control problem for the heat equation, this time taken from the SLICOT Model Reduction Benchmark Collection [19]. (The eigenvalues of the DARE solution are shown in Figure 1.) In contrast to Example 4.1, this time a finite differences discretization is used,

$\Delta t = 0.1$				
	X_{dare}	Z_{smith}	Z_{adi} $\mu_* = 0.96325$	Z_{adi} no shift
Newton steps		4	4	4
Smith/ADI steps		1175–1188	26	606–613
$\mathcal{R}(X)$	$2.2 \cdot 10^{-12}$	$1.1 \cdot 10^{-13}$	$1.1 \cdot 10^{-13}$	$1.1 \cdot 10^{-13}$
$\|\|X_{dare} - X\|\|_F / \|\|X\|\|_F$		$6.1 \cdot 10^{-11}$	$1.3 \cdot 10^{-11}$	$3.6 \cdot 10^{-11}$
$\text{rank}(ZZ^T)$	13	11	11	11
$\Delta t = 0.01$				
	X_{dare}	Z_{smith}	Z_{adi} $\mu_* = 0.99306$	Z_{adi} no shift
Newton steps		4	4	4
Smith/ADI steps		10544–10658	48	5447–5506
$\mathcal{R}(X)$	$2.3 \cdot 10^{-13}$	$1.6 \cdot 10^{-14}$	$1.5 \cdot 10^{-14}$	$1.5 \cdot 10^{-14}$
$\|\|X_{dare} - X\|\|_F / \|\|X\|\|_F$		$4.7 \cdot 10^{-10}$	$3.2 \cdot 10^{-11}$	$2.2 \cdot 10^{-10}$
$\text{rank}(ZZ^T)$	16	16	16	16

Table 2: Example 1, $tol = 10^{-12}$, $n = 1000$, $\mu_0 = 0$.

where the spatial domain is discretized into segments of length $h = \frac{1}{N+1}$. Suppose for example that one wants to heat in a point of the rod located at 1/3 of the length and wants to record the temperature at 2/3 of the length. We obtain the semi-discretized system:

$$\begin{aligned} \dot{x}(t) &= Kx(t) + Fu(t), \qquad x(0) = 0, \\ y(t) &= Cx(t), \end{aligned}$$

where

$$K = -\frac{\alpha}{h^2}\begin{bmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \\ & & & -1 & 2 \end{bmatrix} \in \mathbb{R}^{N\times N}, F = e_{(N+1)/3} \in \mathbb{R}^N, C = e_{2N/3} \in \mathbb{R}^N,$$

and $x(t) \in \mathbb{R}^N$ is the solution evaluated at each x value in the discretization for t. Here, e_k denotes the kth unit vector. Now if we want to completely discretize the system, for example using Crank-Nicholson we obtain:

$$\begin{aligned} Ex_{k+1} &= Ax_k + Bu_k, \qquad x_0 = 0 \\ y_k &= Cx_k \end{aligned}$$

where $E = I_N - \frac{\Delta t}{2}K$, $A = I_N + \frac{\Delta t}{2}K$, $B = \Delta t F$, and Δt is the time step.

The data provided in the benchmark collection [19] yields matrices K, F and C with $N = 200, m = p = 1$.

As $\rho(A, E) \approx 0.9975$, the iteration for the Stein equation converges quite slowly, for the results see Table 3.

When using the Smith iteration on this example, the iterations converged after 3 Newton iteration, while in each iteration step the Smith iteration took 1343 steps. The rank of X_{exact} is 31, the rank of the computed low-rank factorization factor Z is 39, the rank of ZZ^T is 25. When looking at the singular values of X_{exact} and ZZ^T, one can see that this is a numerical accuracy problem, the first about 25 singular values are of the order of 10^{-8} to 10^{-14}, while the rest is even smaller. Without the low-rank reduction of Z_j in every step, the factor Z_j would increase to the size $\mathbb{R}^{200\times 2688}$, while due to the low-rank reduction, the factor Z_j is at most of size $\mathbb{R}^{200\times 39}$. The ADI iteration converged much quicker than the Smith iteration, moreover the use of a shift allowed the computation of a more accurate solution.

In order to obtain a better convergence behavior for the Stein equation, we use the same data, but set $K = 0.5 \cdot K$. This reduces $\rho(A, E)$ to ≈ 0.4987. As can be seen from Table 3, the iteration for the Stein equation converges much faster for all algorithms considered. Here, our heuristic to choose the shift does not work very well, the computed shift is close to 0 which is almost the same as using no shift.

Using a smaller *tol* does give more accurate results, but no significant chance in the number of iterations needed.

Example 4.3 For the final example, we set up

$$B = C = [v, \ldots, v] \in \mathbb{R}^{100}$$

with $v = [1, 1, 0, \ldots,]^T \in \mathbb{R}^{10}$ and

$$A = I, K = \text{tridiag}(D, T, D) \in \mathbb{R}^{100\times 100},$$

with $D = \text{diag}(-121, \ldots, -121) \in \mathbb{R}^{10\times 10}$ and

$$T = \begin{bmatrix} 484 & -115.5 & & \\ -126.5 & \ddots & \ddots & \\ & \ddots & \ddots & -115.5 \\ & & -126.5 & 484 \end{bmatrix} = \text{tridiag}(-126.5, 484, -115.5) \in \mathbb{R}^{10\times 10}.$$

The K matrix is constructed using the MATLAB function `fdm2D_matrix` from LYAPACK [50]. We include this fairly small example, as it shows that the number of Newton steps also may depend on the algorithm used for the inner iteration to solve the Stein equations. This number is slightly larger for the Smith iteration than for the ADI iteration with shift.

original $K, \rho(A,E) \approx 0.9975$				
	X_{dare}	Z_{smith}	Z_{adi} $\mu_* = -0.99197$	Z_{adi} no shift
Newton steps		3	3	3
Smith/ADI steps		1343	85	737
$\mathcal{R}(X)$	$1.8 \cdot 10^{-12}$	$2.6 \cdot 10^{-4}$	$3.4 \cdot 10^{-10}$	$1.3 \cdot 10^{-4}$
$\|\|X_{dare} - X\|\|_F / \|\|X\|\|_F$		$1.8 \cdot 10^{-6}$	$3.7 \cdot 10^{-8}$	$9.4 \cdot 10^{-7}$
$\text{rank}(X)/\text{rank}(ZZ^T)$	31	25	25	26
$K = 0.5K, \rho(A,E) \approx 0.4987$				
	X_{dare}	Z_{smith}	Z_{adi} $\mu_* = -0.45332$	Z_{adi} no shift
Newton steps		3	3	3
Smith/ADI steps		12	10	7
$\mathcal{R}(X)$	$2.8 \cdot 10^{-12}$	$1.1 \cdot 10^{-8}$	$3.3 \cdot 10^{-12}$	$2.6 \cdot 10^{-9}$
$\|\|X_{dare} - X\|\|_F / \|\|X\|\|_F$		$5.5 \cdot 10^{-10}$	$2.2 \cdot 10^{-10}$	$1.3 \cdot 10^{-10}$
$\text{rank}(X)/\text{rank}(ZZ^T)$	16	10	10	10

Table 3: Example 2, $n = 200$, $tol = 10^{-8}$, $\mu_0 = 0$.

	X_{dare}	Z_{smith}	Z_{adi} $\mu_* = -0.85361$
Newton steps		10	8
Smith/ADI steps		10–128	8–12
$\mathcal{R}(X)$	$1.3 \cdot 10^{-11}$	$1.1 \cdot 10^{-11}$	$1.6 \cdot 10^{-12}$
$\|\|X_{dare} - X\|\|_F / \|\|X\|\|_F$		$4.3 \cdot 10^{-12}$	$2.2 \cdot 10^{-14}$
$\text{rank}(X)/\text{rank}(ZZ^T)$	6	6	6

Table 4: Example 3, $tol = 10^{-12}$, $n = 100$, $\Delta t = 0.01$, $\mu_0 = -1$.

5 Conclusions

This paper addresses the numerical solution of large, sparse DAREs based on the Newton method. Its primary step involves the solution of large, sparse, stable Stein equations. We have presented two iterative methods which deliver low-rank approximations to the desired solution, a Smith and an ADI iteration. The ADI iteration can be accelerated significantly by introducing suitable shift parameters. We presented a simple heuristic algorithm for determining a set of ADI parameters. Finally, the algorithms are used to numerically solve an optimal control problem for parabolic PDEs.

Future work will include a detailed study of the problem of choosing the ADI parameters as well as the adaptation of the precision to which the Stein equation is solved to that of the Newton recursion as in [24] done for continuous-time

algebraic Riccati equations.

Acknowledgments

The first author's work was supported by the DFG project *Numerische Lösung von Optimalsteuerungsproblemen für instationäre Diffusions-Konvektions- und Diffusions-Reaktionsgleichungen*, grant BE3715/1-1.

References

[1] J. Abels and P. Benner, *CAREX – a collection of benchmark examples for continuous-time algebraic Riccati equations (version 2.0)*, SLICOT Working Note 1999-14, Nov. 1999. Available from `www.slicot.de`.

[2] E. Armstrong and G. T. Rublein, *A stabilization algorithm for linear discrete constant systems*, IEEE Trans. Automat. Control, AC–21 (1976), pp. 629–631.

[3] W. Arnold, III. and A. Laub, *Generalized eigenproblem algorithms and software for algebraic Riccati equations*, Proc. IEEE, 72 (1984), pp. 1746–1754.

[4] H. Banks and K. Kunisch, *The linear regulator problem for parabolic systems*, SIAM J. Cont. Optim., 22 (1984), pp. 684–698.

[5] A. Y. Barraud, *A numerical algorithm to solve* $A^T XA - X = Q$, IEEE Trans. Automat. Control, AC-22 (1977), pp. 883–885.

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

[7] P. Benner, *Contributions to the Numerical Solution of Algebraic Riccati Equations and Related Eigenvalue Problems*, Logos–Verlag, Berlin, Germany, 1997. *Also:* Dissertation, Fakultät für Mathematik, TU Chemnitz–Zwickau, 1997.

[8] ——, *Solving large-scale control problems*, IEEE Control Systems Magazine, 14 (2004), pp. 44–59.

[9] ——, *Editorial: Large-scale matrix equations of special type (special issue)*, Numer. Lin. Alg. Appl., 15 (2008), pp. 747–754.

[10] P. Benner and H. Fassbender, *Initializing Newton's method for discrete-time algebraic Riccati equations using the butterfly SZ algorithm*, in Proc.

1999 IEEE Intl. Symp. CACSD, Kohala Coast-Island of Hawai'i, Hawai'i, USA, August 22–27, 1999 (CD-Rom), O. Gonzalez, ed., 1999, pp. 70–74.

[11] ——, *A hybrid method for the numerical solution of discrete-time algebraic Riccati equations*, Contemp. Math., 280 (2001), pp. 255–269.

[12] P. Benner, H. Fassbender, and D. Watkins, *SR and SZ algorithms for the symplectic (butterfly) eigenproblem*, Linear Algebra Appl., 287 (1999), pp. 41–76.

[13] P. Benner, J.-R. Li, and T. Penzl, *Numerical solution of large Lyapunov equations, Riccati equations, and linear-quadratic control problems*, Numer. Lin. Alg. Appl., 15 (2008), pp. 755–777.

[14] P. Benner, E. Quintana-Ortí, and G. Quintana-Ortí, *Numerical solution of discrete stable linear matrix equations on multicomputers*, Parallel Algorithms and Appl., 17 (2002), pp. 127–146.

[15] ——, *Solving linear-quadratic optimal control problems on parallel computers*, Optim. Meth. Softw., 23 (2008), pp. 879–909.

[16] R. Byers, D. Mackey, V. Mehrmann, and H. Xu, *Symplectic, BVD, and palindromic approaches to discrete-time control problems*, Preprint 14-2008, Institut für Mathematik, Technische Universität Berlin, http://www.math.tu-berlin.de/preprints/abstracts/Report-14-2008.rdf.html, 2008.

[17] D. Calvetti, N. Levenberg, and L. Reichel, *Iterative methods for* $X - AXB = C$, J. Comput. Appl. Math., 86 (1997), pp. 73–101.

[18] D. Calvetti and L. Reichel, *Application of ADI iterative methods to the restoration of noisy images*, SIAM J. Matrix Anal. Appl., 17 (1996), pp. 165–186.

[19] Y. Chahlaoui and P. Van Dooren, *A collection of benchmark examples for model reduction of linear time invariant dynamical systems*, SLICOT Working Note 2002–2, Feb. 2002. Available from `www.slicot.org`.

[20] T. Chan, *Rank revealing QR factorizations*, Linear Algebra Appl., 88/89 (1987), pp. 67–82.

[21] B. Datta, *Numerical Methods for Linear Control Systems*, Elsevier Academic Press, 2004.

[22] J. Dennis and R. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Prentice Hall, Englewood Cliffs, New Jersey, 1983.

[23] H. Fassbender, *Symplectic methods for the symplectic eigenproblem*, Kluwer Academic/Plenum Publisher, 2000.

[24] F. Feitzinger, T. Hylla, and E. W. Sachs, *Inexact Kleinman-Newton method for Riccati equations*, SIAM J. Matrix Anal. Appl., 31 (2009), pp. 272–288.

[25] K. Gallivan, X. Rao, and P. van Dooren, *Singular Riccati equations stabilizing large-scale systems*, Linear Algebra Appl., 415 (2006), pp. 359–372.

[26] J. Gardiner and A. Laub, *Parallel algorithms for algebraic Riccati equations*, Internat. J. Control, 54 (1991), pp. 1317–1333.

[27] J. Gardiner, A. Laub, J. Amato, and C. Moler, *Solution of the Sylvester matrix equation* $AXB + CXD = E$, ACM Trans. Math. Software, 18 (1992), pp. 223–231.

[28] J. Gardiner, M. Wette, A. Laub, J. Amato, and C. Moler, *Algorithm 705: A Fortran-77 software package for solving the Sylvester matrix equation* $AXB^T + CXD^T = E$, ACM Trans. Math. Software, 18 (1992), pp. 232–238.

[29] G. Golub and C. Van Loan, *Matrix Computations*, Johns Hopkins University Press, Baltimore, third ed., 1996.

[30] S. Gugercin, D. Sorensen, and A. Antoulas, *A modified low-rank Smith method for large-scale Lyapunov equations*, Numer. Algorithms, 32 (2003), pp. 27–55.

[31] S. Hammarling, *Numerical solution of the discrete-time, convergent, non-negative definite Lyapunov equation*, Sys. Control Lett., 17 (1991), pp. 137–139.

[32] G. Hewer, *An iterative technique for the computation of steady state gains for the discrete optimal regulator*, IEEE Trans. Automat. Control, AC-16 (1971), pp. 382–384.

[33] 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), pp. 53–62.

[34] R. Horn and C. Johnson, *Matrix Analysis*, Cambridge University Press, 1985.

[35] V. Ionescu, C. Oară, and M. Weiss, *General matrix pencil techniques for the solution of algebraic Riccati equations: A unified approach*, IEEE Trans. Automat. Control, 42 (1997), pp. 1085–1097.

[36] K. Ito, *Finite-dimensional compensators for infinite-dimensional systems via galerkin-type approximation*, SIAM Journal on Control and Optimization, 28 (1990), pp. 1251–1269.

[37] D. Kleinman, *On an iterative technique for Riccati equation computations*, IEEE Trans. Automat. Control, AC-13 (1968), pp. 114–115.

[38] ——, *Stabilizing a discrete, constant, linear system with application to iterative methods for solving the Riccati equation*, IEEE Trans. Automat. Control, AC-19 (1974), pp. 252–254.

[39] P. Lancaster and L. Rodman, *The Algebraic Riccati Equation*, Oxford University Press, Oxford, 1995.

[40] I. Lasiecka and R. Triggiani, *Differential and Algebraic Riccati Equations with Application to Boundary/Point Control Problems: Continuous Theory and Approximation Theory*, vol. 164 of Lecture Notes in Control and Information Sciences, Springer, Berlin, 1991.

[41] J.-R. Li and J. White, *Low rank solution of Lyapunov equations*, SIAM J. Matrix Anal. Appl., 24 (2002), pp. 260–280.

[42] W.-W. Lin and S.-F. Xu, *Convergence analysis of structure-preserving doubling algorithms for Riccati-type matrix equations*, SIAM J. Matrix Anal. Appl., 28 (2006), pp. 26–39.

[43] V. Mehrmann, *Existence, uniqueness and stability of solutions to singular, linear-quadratic control problems*, Linear Algebra Appl., 121 (1989), pp. 291–331.

[44] ——, *The Autonomous Linear Quadratic Control Problem, Theory and Numerical Solution*, no. 163 in Lecture Notes in Control and Information Sciences, Springer-Verlag, Heidelberg, July 1991.

[45] V. Mehrmann and E. Tan, *Defect correction methods for the solution of algebraic Riccati equations*, IEEE Trans. Automat. Control, 33 (1988), pp. 695–698.

[46] T. Pappas, A. Laub, and N. Sandell, *On the numerical solution of the discrete-time algebraic Riccati equation*, IEEE Trans. Automat. Control, AC-25 (1980), pp. 631–641.

[47] D. Peaceman and H. Rachford, *The numerical solution of parabolic and elliptic differential equations*, J. SIAM, 3 (1955), pp. 28–41.

[48] T. Penzl, *Numerical solution of generalized Lyapunov equations*, Adv. Comp. Math., 8 (1997), pp. 33–48.

[49] ——, *A cyclic low-rank Smith method for large sparse Lyapunov equations*, SIAM J. Sci. Comput., 21 (2000), pp. 1401–1418.

[50] ——, LYAPACK *Users Guide*, Tech. Rep. SFB393/00-33, Sonderforschungsbereich 393 *Numerische Simulation auf massiv parallelen Rechnern*, TU Chemnitz, 09107 Chemnitz, FRG, 2000. Available from `http://www.tu-chemnitz.de/sfb393/sfb00pr.html`.

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

[52] V. SIMA, *Algorithms for Linear-Quadratic Optimization*, vol. 200 of Pure and Applied Mathematics, Marcel Dekker, Inc., New York, NY, 1996.

[53] V. SIMONCINI, *A new iterative method for solving large-scale Lyapunov matrix equations*, SIAM J. Sci. Comput., 29 (2007), pp. 1268–1288.

[54] R. SMITH, *Matrix equation* $XA+BX = C$, SIAM J. Appl. Math., 16 (1968), pp. 198–201.

[55] X. SUN AND E. QUINTANA-ORTÍ, *Spectral division methods for block generalized Schur decompositions*, Math. Comp., 73 (2004), pp. 1827–1847.

[56] A. VARGA, *A note on Hammarling's algorithm for the discrete Lyapunov equation*, Sys. Control Lett., 15 (1990), pp. 273–275.

[57] E. WACHSPRESS, *Iterative solution of the Lyapunov matrix equation*, Appl. Math. Letters, 107 (1988), pp. 87–90.

Chemnitz Scientific Computing Preprints – ISSN 1864-0087