BDF Methods for Large-Scale Differential Riccati Equations*

Peter Benner† Hermann Mena‡

Abstract

We consider the numerical solution of differential Riccati equations. We review the existing methods and investigate whether they are suitable for large-scale problems arising in LQR and LQG design for semi-discretized partial differential equations. Based on this review, we suggest an efficient matrix-valued implementation of the BDF for differential Riccati equations.

Keywords: differential Riccati equations, backward differentiation formulas, linear-quadratic regulator, optimal control, algebraic Riccati equation

AMS(MOS) subject classification: 65L06, 93A15, 93C20

1 Introduction

Consider the differential Riccati equation (DRE)

$$\begin{aligned} &\dot{X}(t) + Q(t) + X(t)A(t) + B(t)X(t) - X(t)R(t)X(t) = 0 \\ &X(t_0) = X_0 \qquad t_0 \le t \le T \end{aligned} \tag{1}$$

where $t \in [t_0, T]$ and $A(t) \in \mathbb{R}^{n\times n}$, $B(t) \in \mathbb{R}^{m\times m}$, $Q(t) \in \mathbb{R}^{m\times n}$, $R(t) \in \mathbb{R}^{n\times m}$, $X(t) \in \mathbb{R}^{m\times n}$ are smooth matrix-valued functions.

The DRE arises in several applications, especially in control theory. Here, we will be particularly interested in the case of symmetric DREs, where (1) is called symmetric if $Q(t), R(t)$ are square, symmetric matrices and $B(t) = A(t)^T$ for all $t \in [t_0, T]$. Throughout

*This work is supported by the DAAD-funded joint Ph.D. programme of EPN, Quito (Ecuador) – TU Berlin (Germany) and by the *Sonderforschungsbereich 393, "Numerische Simulation auf massiv parallelen Rechnern"*, TU Chemnitz (Germany).

†Fakultät für Mathematik Technische Universität Chemnitz, D-09107 Chemnitz, Germany. benner@mathematik.tu-chemnitz.de.

‡Departamento de Matemáticas, Escuela Politécnica Nacional, Quito, Ecuador. hmena@server.epn.edu.ec

this paper we assume that there exists a unique solution on $[t_0, T]$ (for a discussion of existence and uniqueness of DREs see [1, 20, 34]). With this assumption, it follows that the solution $X(t)$ is symmetric as $X(t)^T$ is also a solution of (1). Symmetric DREs arise from linear-quadratic optimal control problems like LQR and LQG design with finite-time horizon, in H_∞ control of linear-time varying systems, as well as in differential games; see, e.g., [1, 14, 18, 32]. Unfortunately, in most control problems, fast and slow modes are present. This implies that the associated DRE will be fairly stiff which in turn demands for implicit methods to solve such DREs numerically. Therefore, we will focus here on the stiff case.

Large-scale DREs result from semi-discretized optimal control problems for instationary partial differential equations (PDEs) of parabolic or hyperbolic type, like, e.g., the heat equation, reaction-diffusion or convection-diffusion equations, the wave equation, etc., with point or boundary control; see, e.g., [17, 23, 24, 25]. Imposing a quadratic cost functional, the solution of the optimal control is often given via feedback control where the feedback operator is given in terms of an operator-valued differential Riccati equation, see [28, 23, 24, 25]. In order to solve such a problem numerically, the PDE has to be discretized appropriately. Under suitable conditions for the finite-dimensional approximation, it can be shown that the solutions of the finite-dimensional linear-quadratic optimal control problems arising from a semi-discretization of the PDE in space converge to the optimal control for the infinite-dimensional problem. Hence, in order to apply such a feedback control strategy to PDE control, we need to solve the large-scale symmetric DREs resulting from the semi-discretization. Typically, $A(t)$ represents a discretized elliptic operator and $Q(t), R(t)$ are semi-definite with low rank. Hence, we will derive numerical methods capable of exploiting the given structure of the coefficient matrices.

Besides the vast variety of linear-quadratic problems that can be solved if an efficient DRE solver is available, the task of solving large-scale DREs appears also to become an increasingly important issue in nonlinear optimal control problems of tracking-type and stabilization problems for classes of nonlinear instationary PDEs. LQG design on short time intervals is the main computational ingredient in recently proposed receding horizon and model predictive control approaches for such problems, see [16, 17, 15]. The ability to efficiently solve large-scale DREs is therefore of paramount importance there, too.

This paper is organized as follows. First, we will review (without guaranteeing completeness) the existing methods for solving DREs numerically in Section 2. We will also discuss whether these methods are suitable for large-scale computations. In Section 3 we then suggest a possible implementation of the backward differentiation formulas (BDF) methods based on exploiting the structure arising in semi-discretized optimal control problems for PDEs. It has been observed before [8] that the nonlinear systems of equations that have to be solved in each time step are algebraic Riccati equation (AREs). The sparsity structure of the state coefficient matrices as well as the low-rank structure of the constant and quadratic term in the resulting AREs allows to apply the recently suggested low-rank ADI Newton method for AREs [4, 5, 6]. We therefore review this approach in Section 4. A brief summary and an outlook on future work conclude the paper.

2 Numerical Methods for Solving DREs

The numerical methods for DREs of the form (1) can essentially be distinguished into five classes. In this section, we will briefly review the methods and discuss their suitability for solving large-scale problems. Note that the solution matrix of the DRE is a symmetric $n \times n$ matrix. Even in case symmetry is exploited, the storage needed is of size $n(n+1)/2$. For example, for a semi-discretized 2D PDE problem with say, $11,000$ degrees of freedom, this would require about 1 GB of storage for each time step if double precision is to be used! Therefore, we will examine the available methods regarding their potential to circumvent the storage of $X(t)$ as a square matrix.

The naive approach. The first idea is to vectorize the DRE, i.e., to unroll the matrices into vectors and to integrate the resulting system of n^2 differential equations using any kind of numerical integration scheme. This approach is not suitable for large-scale problems, as for implicit methods, nonlinear systems of equations with n^2 unknowns have to be solved in each time step. This can be reduced exploiting symmetry to $n(n+1)/2$, but still this would require $\mathcal{O}(n^2)$ workspace.

Linearization. The second type of methods is based on transforming the quadratic DRE into the system of linear first-order matrix differential equations

$$\begin{aligned} \frac{d}{dt}\begin{bmatrix} U(t) \\ V(t) \end{bmatrix} &= \underbrace{\begin{bmatrix} -A(t) & R(t) \\ Q(t) & A(t)^T \end{bmatrix}}_{:=H(t)} \begin{bmatrix} U(t) \\ V(t) \end{bmatrix}, \qquad t \in (t_0, T], \\ \begin{bmatrix} U(t_0) \\ V(t_0) \end{bmatrix} &= \begin{bmatrix} U_0 \\ V_0 \end{bmatrix}, \end{aligned} \tag{2}$$

where $U(t) \in \mathbb{R}^{n\times n}$, $V(t) \in \mathbb{R}^{n\times n}$, $V_0 U_0^{-1} = X(t_0)$ for some $U_0 \in \mathbb{R}^{n\times n}$ invertible and some $V_0 \in \mathbb{R}^{n\times n}$. If the solution of (1) exists on the interval $[t_0, T]$, then the solution of (2) exists, $U(t)$ is invertible on $[t_0, T]$, and

$$X(t) = V(t)U^{-1}(t). \tag{3}$$

Conversely, if the solution of (2) exists and $U(t)$ is nonsingular for all $t \in [t_0, T]$, then the solution of (1) exists in the same interval and is given by (3). The linear differential equation (2) is a Hamiltonian differential equation. In the time-invariant case, this allows an efficient integration for dense problems, [26], using numerical methods for the Hamiltonian eigenproblem.

Another approach which is applicable to time-varying systems uses the fundamental solution of the linear first-order ordinary differential equation. This method, called now the Davison-Maki method, is proposed in [9]. A modified variant, avoiding some numerical instabilities due to the inversion of possibly ill-conditioned matrices, is proposed in [19]. As the exponential of the $2n \times 2n$-matrix $H(t_0)$ is required, this cannot be applied in the large-scale case.

Chandrasekhar's method. The third type of algorithms is applicable to symmetric time-invariant DREs and is based on the transformation of (1) into two coupled systems of nonlinear differential equations, the so-called *Chandrasekhar system*

$$\begin{aligned} \dot{L} &= (K^T G^T - A^T)L, & L(0) = L_0 \in \mathbb{R}^{n\times l} \\ \dot{K} &= -G^T L L^T, & K(0) = G^T X_0 \in \mathbb{R}^{m\times n} \end{aligned} \tag{4}$$

where $Q(t) \equiv CC^T$, $R(t) \equiv GG^T$. The solution of the DRE can be recovered from that of the Chandrasekhar system. The method can be adapted to the time-varying case, see [21], but there are several numerical difficulties involved in integrating (4), see [35]. In general, the method is unstable and is therefore not considered here any further although it is suitable for large-scale problems [2].

Superposition methods. This type of methods is based on the superposition property of Riccati solutions, see [13]. The general solution of a DRE can be expressed as a nonlinear combination of at most five independent solutions. This class of methods requires integration of the DRE several times with different initial conditions before applying the complex superposition formulas and the computational complexity therefore is too high to apply these formulae to the large-scale problems considered here.

Matrix-versions of standard ODE methods. These methods solve the DRE using matrix-valued algorithms based on standard numerical algorithms (see [7, 10]) for solving ordinary differential equations (ODEs). As we are concerned with stiffness, we only consider implicit methods here. In order to use the given structure as much as possible, we are interested in methods which, written in matrix form, yield an algebraic Riccati equation (ARE) as the nonlinear system of equations to be solved in each time step. It turns out that there is a vast variety of methods that are applicable here, e.g., the backward differentiation formulas (BDF), the midpoint and trapezoidal rules.

The BDF schemes allow an efficient implementation for the large-scale problems considered here. Moreover, BDF schemes are particularly suitable for stiff ODEs. Therefore, we will concentrate on this class of methods.

In the next section, we describe the matrix-valued implementation of the BDF for DREs.

3 BDF methods for large-scale DREs

We briefly describe the BDF method for DREs in matrix-valued form similar to [7]. We will then discuss how this scheme can be implemented for large-scale problems. For this purpose, we only consider symmetric DREs of the form

$$\begin{aligned} \dot{X}(t) &= Q(t) + A^T(t)X(t) + X(t)A(t) - X(t)R(t)X(t) \equiv F(t, X(t)) \\ X(t_0) &= X_0, \qquad t_0 \le t \le T. \end{aligned} \tag{5}$$

p	β	α_0	α_1	α_2	α_3	α_4
1	1	1				
2	$\frac{2}{3}$	$\frac{4}{3}$	$-\frac{1}{3}$			
3	$\frac{6}{11}$	$\frac{18}{11}$	$-\frac{9}{11}$	$\frac{2}{11}$		
4	$\frac{12}{25}$	$\frac{48}{25}$	$-\frac{36}{25}$	$\frac{16}{25}$	$-\frac{3}{25}$	
5	$\frac{60}{137}$	$\frac{300}{137}$	$-\frac{300}{137}$	$\frac{200}{137}$	$-\frac{75}{137}$	$-\frac{12}{137}$

Table 1: coefficients of the BDF p-step methods for $p < 6$.

The BDF applied to the DRE (5) yield

$$X_{k+1} = \sum_{j=0}^{p-1} \alpha_j X_{k-j} + h\beta F(t_{k+1}, X_{k+1})$$

where h is the step size, $t_{k+1} = h + t_k$, $X_{k+1} \equiv X(t_{k+1})$ and α_j, β are the coefficients for the p-step BDF formula, given in Table 1.

Hence we obtain the Riccati-BDF difference equation

$$-X_{k+1} + h\beta(Q_{k+1} + A_{k+1}^T X_{k+1} + X_{k+1}A_{k+1} - X_{k+1}R_{k+1}X_{k+1}) + \sum_{j=0}^{p-1} \alpha_j X_{k-j} = 0.$$

Re-arranging terms, we see that this is an ARE for X_{k+1},

$$\begin{aligned}(h\beta Q_{k+1} \;+\; \sum_{j=0}^{p-1} \alpha_j X_{k-j}) \;+\; (h\beta A_{k+1} - \frac{1}{2}I)^T X_{k+1} + \\ +\; X_{k+1}(h\beta A_{k+1} - \frac{1}{2}I) \;-\; X_{k+1}(h\beta R_{k+1})X_{k+1} \;=\; 0 \end{aligned} \tag{6}$$

that can be solved via any method for AREs. Assuming that

$$\begin{aligned} Q_k &= C_k^T C_k, \qquad C_k \in \mathbb{R}^{p\times n}, \\ R_k &= B_k B_k^T, \qquad B_k \in \mathbb{R}^{n\times m}, \\ X_k &= Z_k Z_k^T, \qquad Z_k \in \mathbb{R}^{n\times z_k}, \end{aligned}$$

the ARE (6) can be written as

$$\hat{C}_{k+1}^T \hat{C}_{k+1} + \hat{A}_{k+1}^T Z_{k+1} Z_{k+1}^T + Z_{k+1} Z_{k+1}^T \hat{A}_{k+1} - Z_{k+1} Z_{k+1}^T \hat{B}_{k+1} \hat{B}_{k+1}^T Z_{k+1} Z_{k+1}^T = 0, \tag{7}$$

where

$$\begin{aligned} \hat{A}_{k+1} &= h\beta A_{k+1} - \frac{1}{2}I, \\ \hat{B}_{k+1} &= \sqrt{h\beta} B_{k+1}, \\ \hat{C}_{k+1} &= [\,\sqrt{h\beta} C_{k+1}, \sqrt{\alpha_0} Z_k, \ldots, \sqrt{\alpha_{p-1}} Z_{k+1-p}\,]. \end{aligned}$$

If $z_k \ll n$ for all times, and (7) can be solved efficiently by exploiting sparsity in A_{k+1} as well as the low-rank nature of the constant and quadratic terms, this can serve as the basis for a DRE solver for large-scale problems. It should be noted that for $p \geq 2$, some of the α_j are negative. This can be treated using complex arithmetic and replacing all transposes in (7) by conjugate complex transposes, but in general it will be more efficient to split the constant term into

$$\hat{C}_{k+1}^T \hat{C}_{k+1} - \tilde{C}_{k+1}^T \tilde{C}_{k+1}$$

where $\hat{C}_{k+1}$ only contains the factors corresponding to positive α_j and $\tilde{C}_{k+1}$ the factors corresponding to negative α_j. We will show how this can be exploited in the ARE solver below.

In our numerical implementation, we benefit from recent algorithmic progress in solving large-scale AREs resulting from semi-discretized control problems for AREs [4, 5, 6]. We will discuss the details of this approach in the next section.

4 Numerical Solution of Large-Scale AREs

Since the ARE (6) is a nonlinear system of equations, it is quite natural to apply Newton's method to find its solutions. This approach has been investigated; details and further references can be found in [36, 22, 29, 33, 3]. To make the derivation more concise, we will use in this section the generic form of an ARE as it arises in LQR and LQG problems, given by

$$0 \;=\; \mathcal{R}(P) \;:=\; C^T C + A^T P + PA - PBB^T P. \tag{8}$$

The case important here, i.e., constant terms of the form $\hat{C}\hat{C}^T - \tilde{C}^T\tilde{C}$, will be explained in Remark 4.1 below.

Observing that the (Frechét) derivative of $\mathcal{R}$ at P is given by the Lyapunov operator

$$\mathcal{R}'_P : Q \to (A - BB^T P)Q + Q(A - BB^T P)^T,$$

Newton's method for AREs can be written as

$$\begin{aligned} N_\ell &:= \left(\mathcal{R}'_{P_\ell}\right)^{-1} \mathcal{R}(P_\ell), \\ X_{\ell+1} &:= X_\ell + N_\ell. \end{aligned}$$

Then one step of the Newton iteration for a given starting matrix P_0 can be implemented as follows:

1. $A_\ell \leftarrow A - BB^T P_\ell$.
2. Solve the Lyapunov equation $A_\ell^T N_\ell + N_\ell A_\ell = -\mathcal{R}(P_\ell)$.
3. $P_{\ell+1} \leftarrow P_\ell + N_\ell$.

Assume a stabilizing P_0 is given such that A_0 is stable. (In the applications considered here, we can use the fact that for a small time step, the approximate solution $X_k \approx X(t_k)$ will in general be a good stabilizing starting value.) Then all A_ℓ are stable and the iterates P_ℓ converge to P_* quadratically. (Here: $P_* = X_{k+1} \approx X(t_{k+1})$.) In order to make this iteration work for large-scale problems, we need a Lyapunov equation solver that employs the structure of A_ℓ as "sparse + low-rank perturbation" by avoiding to form A_ℓ explicitly, and which computes a low-rank approximation to the solution of the Lyapunov equation. A relevant method is derived in detail in [6, 30] and is described in the following.

First, we re-write Newton's method for AREs such that the next iterate is computed directly from the Lyapunov equation in Step 2,

$$A_\ell^T P_{\ell+1} + P_{\ell+1} A_\ell = -C^T C - P_\ell B B^T P_\ell =: -W_\ell W_\ell^T. \tag{9}$$

Assuming that $P_\ell = Z_\ell Z_\ell^T$ for $\operatorname{rank}(Z_\ell) \ll n$ and observing that $\operatorname{rank}(W_\ell) \leq m + p \ll n$, we need only a numerical method to solve Lyapunov equations having a low-rank right hand side which returns a low-rank approximation to the (Cholesky) factor of its solution. For this purpose, we can use a modified version of the *alternating directions implicit (ADI)* method for Lyapunov equations of the form $FQ + QF^T = -WW^T$ with F stable, $W \in \mathbb{R}^{n \times n_w}$. The ADI iteration can be written as [37]

$$\begin{aligned} (F^T + p_j I) Q_{(j-1)/2} &= -WW^T - Q_{j-1}(F - p_j I), \\ (F^T + \overline{p_j} I) Q_j^T &= -WW^T - Q_{(j-1)/2}(F - \overline{p_j} I), \end{aligned} \tag{10}$$

where $\overline{p}$ denotes the complex conjugate of $p \in \mathbb{C}$. If the shift parameters p_j are chosen appropriately, then $\lim_{j\to\infty} Q_j = Q$ with a superlinear convergence rate. Starting this iteration with $Q_0 = 0$ and observing that for stable F, Q is positive semidefinite, it follows that $Q_j = Y_j Y_j^T$ for some $Y_j \in \mathbb{R}^{n \times r_j}$. Inserting this factorization into the above iteration, re-arranging terms and combining two iteration steps, we obtain a *factored ADI iteration* that in each iteration step yields n_w new columns of a full-rank factor of Q (see [6, 27, 30] for several variants of this method):

$$\begin{aligned} &V_1 \leftarrow \sqrt{-2\operatorname{Re}(p_1)}(F^T + p_1 I)^{-1} W, \qquad Y_1 \leftarrow V_1 \\ &\text{FOR } \; j = 2, 3, \ldots \\ &\qquad V_j \leftarrow \frac{\sqrt{\operatorname{Re}(p_j)}}{\sqrt{\operatorname{Re}(p_{j-1})}} \left(I - (p_j + \overline{p_{j-1}})(F^T + p_j I)^{-1} \right) V_{j-1}, \\ &\qquad Y_j \leftarrow \left[\begin{array}{cc} Y_{j-1} & V_j \end{array} \right]. \\ &\text{END FOR} \end{aligned}$$

It should be noted that all V_j's have the same number of columns as $W \in \mathbb{R}^{n \times n_w}$, i.e., at each iteration j, we have to solve w linear systems of equations with the same coefficient matrix $F^T + p_j I$. If convergence of the factored ADI iteration with respect to a suitable stopping criterion is achieved after $j_{\max}$ steps, then $Y_{j_{\max}} = [V_1, \ldots, V_{j_{\max}}] \in \mathbb{R}^{n \times j_{\max} n_w}$, where $V_j \in \mathbb{R}^{n \times n_w}$. For large n and small n_w we therefore expect that $r_j := j_{\max} n_w \ll n$. In

that case, we have computed a low-rank approximation $Y_{j_{\max}}$ to a factor Y of the solution, that is $Q = YY^T \approx Y_{j_{\max}} Y_{j_{\max}}^T$. In case $n_w \cdot j_{\max}$ becomes large, a column compression technique from [12] can be applied to reduce the number of columns in $Y_{j_{\max}}$ without adding significant error.

For an implementation of this method, we need a strategy to select the shift parameters p_j. We will not discuss this problem here in detail; see [30, 27] and references therein for a detailed discussion. A numerically inexpensive, heuristic algorithm that gives good performance in practice can be found in [30]. Usually, a finite number of shifts is computed in advance and applied cyclically if the ADI method needs more iterations than the number of available shifts.

Since A_ℓ is stable for all ℓ we can apply the modified ADI iteration to (9). Then, $W = \left[\begin{array}{cc} C^T & P_\ell B \end{array}\right]$ and hence, $n_w = m + p$, so that usually $n_w \ll n$.

Remark 4.1 *The solution of the AREs (7) arising for BDF methods with $p > 1$, where the constant term is replaced by*

$$\hat{C}_{k+1}^T \hat{C}_{k+1} - \tilde{C}_{k+1}^T \tilde{C}_{k+1}$$

as described in the last section, only requires to split the Lyapunov equation (9) into the two equations

$$\begin{aligned} A_\ell^T \hat{P}_{\ell+1} + \hat{P}_{\ell+1} A_\ell &= -\hat{C}^T \hat{C} - P_\ell B B^T P_\ell, \\ A_\ell^T \tilde{P}_{\ell+1} + \tilde{P}_{\ell+1} A_\ell &= -\tilde{C}^T \tilde{C}. \end{aligned}$$

Then $P_{\ell+1} = \hat{P}_{\ell+1} - \tilde{P}_{\ell+1}$. The two Lyapunov equations can be solved simultaneously by the factored ADI iteration as the linear systems of equations to be solved in each step have the same coefficient matrices.

Note that the above algorithm can be implemented in real arithmetic by combining two steps, even if complex shifts need to be used, which may be the case if A_ℓ is nonsymmetric. A complexity analysis of the factored ADI method depends on the method used for solving the linear systems in each iteration step. If applied to $F = A_\ell^T$ from (9), we have to deal with the situation that A_ℓ is a shifted sparse matrix plus a low-rank perturbation. If we can solve for the shifted linear system of equations in (10) efficiently, the low-rank perturbation can be dealt with using the Sherman-Morrison-Woodbury formula [11] in the following way: let ℓ be the index of the Newton iterates and let j be the index of the ADI iterates used to solve the ℓth Lyapunov equation, respectively, and set $K_\ell := B^T P_\ell$. Then

$$\left(F^T + p_j^{(\ell)} I_n\right)^{-1} = \left(A + p_j^{(\ell)} I_n - B K_\ell\right)^{-1} = \left(I_n + L_\ell (I_m - K_\ell L_\ell)^{-1} K_\ell\right) (A + p_j^{(\ell)} I_n)^{-1},$$

where $L_\ell := (A + p_j^{(\ell)} I_n)^{-1} B$. Hence, all linear systems of equations to be solved in one iteration step have the same coefficient matrix $A + p_j^{(\ell)} I_n$. If $A + p_j^{(\ell)} I_n$ is a banded matrix or can be re-ordered to become banded, then a direct solver can be employed. If workspace permits, it is desirable to compute a factorization of $A + p_j^{(\ell)} I_n$ for each different shift

parameter beforehand (usually, very few parameters are used). These factorizations can then be used in each iteration step of the ADI iteration. In particular, if A is symmetric positive definite, as will be the case in many applications from PDE constrained optimal control problems, and can be re-ordered in a narrow band matrix, then each factorization requires $\mathcal{O}(n)$ flops, and the total cost $\mathcal{O}(\ell_{\max} \max(j_{\max})n)$ scales with n as desired. If iterative solvers are employed for the linear systems, it should be noted that only one Krylov space needs to be computed (see [27] for details) and hence we obtain an efficient variant of the factored ADI iteration.

Stopping criteria for the modified ADI iteration can be based either on the fact that $\|V_j\| \to 0$ very rapidly or on the residual norm $\|FY_jY_j^T + Y_jY_j^TF^T + WW^T\|$; see [31] for an efficient way to compute the Frobenius norms of the residuals. Moreover, the stopping criteria should be based on the tolerance for the accuracy provided by the BDF method.

5 Conclusions and Outlook

Solving large-scale differential Riccati equations is a central issue in many control design problems for instationary partial differential equations. Linear-quadratic optimization problems on a finite time-horizon immediately lead to the problem of solving DREs. As LQG design on short time intervals is the main computational ingredient in recently proposed receding horizon and model predictive control approaches for stabilization and tracking-type control problems, the ability to efficiently solve large-scale DREs is of paramount importance for nonlinear optimal control problems, too.

By rewriting the BDF method for stiff ordinary differential equations in terms of matrix operations, it turns out that the nonlinear equations to be solved in each time step are algebraic Riccati equations that can efficiently be solved using Newton's method with stating guess taken from the last time step. For large-scale problems arising in LQG design, it is possible to efficiently implement Newton's method for AREs based on a low-rank version of the ADI method for Lyapunov equations. We expect this approach to become a computationally efficient core procedure for LGQ design or LQG-based receding horizon control for instationary PDE-constrained optimal control problems. An implementation of the proposed algorithm is in preparation and will be tested for several PDE control problems. The results of these tests will be reported elsewhere.

References

[1] H. Abou-Kandil, G. Freiling, V. Ionescu, and G. Jank. *Matrix Riccati Equations in Control and Systems Theory.* Birkhäuser, Basel, Switzerland, 2003.

[2] H.T. Banks and K. Ito. A numerical algorithm for optimal feedback gains in high dimensional linear quadratic regulator problems. *SIAM J. Cont. Optim.*, 29(3):499–515, 1991.

[3] P. Benner. Computational methods for linear-quadratic optimization. *Supplemento ai Rendiconti del Circolo Matematico di Palermo, Serie II*, No. 58:21–56, 1999.

[4] P. Benner. Efficient algorithms for large-scale quadratic matrix equations. *Proc. Appl. Math. Mech.*, 1(1):492–495, 2002.

[5] P. Benner. Solving large-scale control problems. *IEEE Control Systems Magazine*, 14(1):44–59, 2004.

[6] P. Benner, J.-R. Li, and T. Penzl. Numerical solution of large Lyapunov equations, Riccati equations, and linear-quadratic control problems. Unpublished manuscript, 2000.

[7] C. Choi and A.J. Laub. Constructing Riccati differential equations with known analytic solutions for numerical experiments. *IEEE Trans. Automat. Control*, 35:437–439, 1990.

[8] C. Choi and A.J. Laub. Efficient matrix-valued algorithms for solving stiff Riccati differential equations. *IEEE Trans. Automat. Control*, 35:770–776, 1990.

[9] E.J. Davison and M.C. Maki. The numerical solution of the matrix Riccati differential equation. *IEEE Trans. Automat. Control*, 18:71–73, 1973.

[10] L. Dieci. Numerical integration of the differential Riccati equation and some related issues. *SIAM J. Numer. Anal.*, 29(3):781–815, 1992.

[11] G.H. Golub and C.F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, Baltimore, third edition, 1996.

[12] S. Gugercin, D.C. Sorensen, and A.C. Antoulas. A modified low-rank Smith method for large-scale Lyapunov equations. *Numer. Algorithms*, 32(1):27–55, 2003.

[13] J. Harnard, P. Winternitz, and R. L. Anderson. Superposition principles for matrix Riccati equations. *J. Math. Phys.*, 24:1062–1072, 1983.

[14] A. Ichikawa and H. Katayama. Remarks on the time-varying H_∞ Riccati equations. *Sys. Control Lett.*, 37(5):335–345, 1999.

[15] K. Ito and K. Kunisch. Receding horizon optimal control for infinite dimensional systems. *ESAIM: Control Optim. Calc. Var.* To appear.

[16] K. Ito and K. Kunisch. On asymptotic properties of receding horizon optimal control. *SIAM J. Cont. Optim.*, 40:1455–1472, 2001.

[17] K. Ito and K. Kunisch. Receding horizon control with incomplete observations. Preprint, October 2003.

[18] O.L.R. Jacobs. *Introduction to Control Theory*. Oxford Science Publications, Oxford, UK, 2nd edition, 1993.

[19] C. Kenney and R.B. Leipnik. Numerical integration of the differential matrix Riccati equation. *IEEE Trans. Automat. Control*, AC-30:962–970, 1985.

[20] H.W. Knobloch and H. Kwakernaak. *Lineare Kontrolltheorie*. Springer-Verlag, Berlin, 1985. In German.

[21] D.G. Lainiotis. Generalized Chandrasekhar algorithms: Time-varying models. *IEEE Trans. Automat. Control*, AC-21:728–732, 1976.

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

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

[24] I. Lasiecka and R. Triggiani. *Control Theory for Partial Differential Equations: Continuous and Approximation Theories I. Abstract Parabolic Systems*. Cambridge University Press, Cambridge, UK, 2000.

[25] I. Lasiecka and R. Triggiani. Control theory for partial differential equations: Continuous and approximation theories II. abstract hyperbolic-like systems over a finite time horizon. In *Encyclopedia of Mathematics and its Applications*, volume 75, pages 645–1067. Cambridge University Press, Cambridge, 2000.

[26] A.J. Laub. Schur techniques for Riccati differential equations. In D. Hinrichsen and A. Isidori, editors, *Feedback Control of Linear and Nonlinear Systems*, pages 165–174. Springer-Verlag, New York, 1982.

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

[28] J.L. Lions. *Optimal Control of Systems Governed by Partial Differential Equations*. Springer-Verlag, Berlin, FRG, 1971.

[29] V. Mehrmann. *The Autonomous Linear Quadratic Control Problem, Theory and Numerical Solution*. Number 163 in Lecture Notes in Control and Information Sciences. Springer-Verlag, Heidelberg, July 1991.

[30] T. Penzl. *Numerische Lösung großer Lyapunov-Gleichungen*. Logos–Verlag, Berlin, Germany, 1998. Dissertation, Fakultät für Mathematik, TU Chemnitz, 1998.

[31] T. Penzl. Eigenvalue decay bounds for solutions of Lyapunov equations: the symmetric case. *Sys. Control Lett.*, 40:139–144, 2000.

[32] I.R. Petersen, V.A. Ugrinovskii, and A.V.Savkin. *Robust Control Design Using H^∞ Methods*. Springer-Verlag, London, UK, 2000.

[33] P.H. Petkov, N.D. Christov, and M.M. Konstantinov. *Computational Methods for Linear Control Systems*. Prentice-Hall, Hertfordshire, UK, 1991.

[34] W.T. Reid. *Riccati Differential Equations*. Academic Press, New York, 1972.

[35] D.L. Russell. *Mathematics of Finite-Dimensional Control Systems*, volume 43 of *Lecture Notes in Pure and Applied Mathematics*. Marcel Dekker Inc., New York, 1979.

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

[37] E.L. Wachspress. Iterative solution of the Lyapunov matrix equation. *Appl. Math. Letters*, 107:87–90, 1988.