Low-rank second-order splitting of large-scale differential Riccati equations

Tony Stillfjord

***Abstract*—We apply first- and second-order splitting schemes to the differential Riccati equation. Such equations are very important in e.g. linear quadratic regulator (LQR) problems, where they provide a link between the state of the system and the optimal input. The methods can also be extended to generalized Riccati equations, e.g. arising from LQR problems given in implicit form. In contrast to previously proposed schemes such as BDF or Rosenbrock methods, the splitting schemes exploit the fact that the nonlinear and affine parts of the problem, when considered in isolation, have closed-form solutions. We show that if the solution possesses low-rank structure, which is frequently the case, then this is preserved by the method. This feature is used to implement the methods efficiently for large-scale problems. The proposed methods are expected to be competitive, as they at most require the solution of a small number of linear equation systems per time step. Finally, we apply our low-rank implementations to the Riccati equations arising from two LQR problems. The results show that the rank of the solutions stay low, and the expected orders of convergence are observed.**

***Index Terms*—Differential Riccati equation, Riccati differential equation, splitting methods, large-scale, low-rank**

I. Introduction

We consider the differential Riccati equation

$$\begin{aligned} &\dot{P}(t) = A^\mathsf{T}P(t) + P(t)A + Q - P(t)SP(t), \quad t \in [0,T] \\ &P(0) = P_0. \end{aligned} \tag{1}$$

This is a semi-linear matrix-valued evolution equation for $P(t) \in \mathbb{R}^{N\times N}$. Such equations arise in many areas, for example in linear quadratic regulator (LQR) problems and optimal state estimation [2], [10], [27]. The main application that we have in mind are the LQR problems, where the goal is to minimize the functional

$$J(u) = \int_0^T \langle \tilde{Q}y, y\rangle + \langle Ru, u\rangle \,\mathrm{d}t,$$

subject to the state and output equations

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

Under certain assumptions on the involved matrices, it can be shown that the optimal input u^* is given in feedback form as $u^*(t) = -R^{-1}B^\mathsf{T}P(T-t)x(t)$, where $P(t)$ solves the Riccati equation (1) with $S = BR^{-1}B^\mathsf{T}$, $Q = C^T\tilde{Q}C$ and $P(T) = 0$, see e.g. [2]. One can also consider the implicit case,

$$M\dot{x} = Ax + Bu,$$

with an invertible mass matrix M. This gives rise to a generalized Riccati equation similar to (1), see e.g. [21], which we handle in Section IV.

In general, we suppose that Equation (2) with $x \in \mathbb{R}^N$ is the discretization of the corresponding PDE. Thus A would prototypically be the discretization of an elliptic differential operator. Alternatively, one could consider the infinite-dimensional case directly by allowing x to belong to a Hilbert or Banach space. Then Equation (1) is operator-valued and can e.g. be treated in the space of Hilbert–Schmidt operators [23], [34]. This corresponds to the "undiscretized" case where $N \to \infty$.

T. Stillfjord is with the Department of Numerical Analysis, Lund University, Lund, Sweden. e-mail: tony@maths.lth.se.

Manuscript received Month1 Date1, 2014; revised Month2 Date2, 201X.

The aim of this paper is to introduce efficient splitting methods for large-scale problems of the type (1). With large-scale we mean that the dimension N is large, and by efficient we mean that the methods should produce small errors without unneccesary computational effort. To this end, we introduce the operators

$$\mathcal{F}P = A^\mathsf{T}P + PA + Q \quad \text{and} \tag{3}$$

$$\mathcal{G}P = -PSP. \tag{4}$$

The motivation for dividing the problem into these two parts is that the subproblems

$$\dot{P} = \mathcal{F}P, \quad P(0) = P_0, \quad \text{and} \tag{5}$$

$$\dot{P} = \mathcal{G}P, \quad P(0) = P_0, \tag{6}$$

are easier to solve separately than the full problem (1): both have closed-form solutions and (5) is additionally affine. In the following we denote the solution operators to these problems by $\mathcal{T}_\mathcal{F}(t)$ and $\mathcal{T}_\mathcal{G}(t)$, respectively. Thus, $\mathcal{T}_\mathcal{F}(t)P_0$ is the solution to (5) at time t.

We propose to use the Lie and Strang splitting schemes [33]. For a introduction to splitting methods in general, we refer to [20, Section IV], [15, Section II.5] and [24]. In our context, the schemes are given by the time-stepping operators

$$\begin{aligned} \mathcal{L}_h &= \mathcal{T}_\mathcal{G}(h)\mathcal{T}_\mathcal{F}(h) \quad \text{and} \\ \mathcal{S}_h &= \mathcal{T}_\mathcal{G}(h/2)\mathcal{T}_\mathcal{F}(h)\mathcal{T}_\mathcal{G}(h/2), \end{aligned} \tag{7}$$

respectively, where $h > 0$ is the time step. The Lie and Strang splitting approximations to the solution of (1) at time $t = nh$ are then given by $\mathcal{L}_h^n P_0$ and $\mathcal{S}_h^n P_0$, respectively. We note that these schemes are first- and second-order accurate, respectively, in the current matrix setting. That is, if $P(t)$ solves (1), then

$$\begin{aligned} \|\mathcal{L}_h^n P_0 - P(nh)\| &\leq Ch \quad \text{and} \\ \|\mathcal{S}_h^n P_0 - P(nh)\| &\leq Ch^2, \end{aligned}$$

for all n such that $nh \leq T$. This can be shown as indicated in [20, Section 4], through proving consistency by e.g. Taylor expansions, and stability for small enough h by the local Lipschitz continuity of the nonlinearity. However, this approach leads to error bounds that depend on the matrices involved. In the case of a discretized PDE, these error bounds might tend to infinity when the discretization is refined. Hence, a stiff error analysis is called for, but this is much more difficult and out of the scope of this technical note. Such an analysis is, however, performed in [17] for a first-order method similar to the Lie splitting, under the restriction $S = I$.

For large-scale problems, even representing the solution $P(t)$ can be a problem due to excessive storage requirements, and one is forced to utilize structural properties. One such property is that of low rank, i.e. that we can accurately approximate $P(t) \approx z(t)z(t)^\mathsf{T}$ where $z(t) \in \mathbb{R}^{N\times r}$ with $r \ll N$. It has been observed that the solutions to the Riccati equation commonly exhibit such behaviour, see e.g. [3]. Partial results on when this is to be expected in the related area of algebraic Riccati equations (ARE) can be found in [4], and for Lyapunov equations the issue is e.g. discussed in [26], [32]. We therefore describe how to implement the schemes in a way that preserves low-rank structure.

Many of the currently available methods for solving the Riccati equation are based on matrix factorizations, which clearly is not feasible in the large-scale case. The idea so far in the large-scale case has been to apply the matrix versions of common time-stepping methods, e.g. BDF methods [6], [25] or Rosenbrock methods [7], [25], and realise that in each step an ARE or a number of Lyapunov equations have to be solved. Low-rank algorithms for the solution of these equations exist and we refer to [9], [31] for recent surveys, see also [5]. There are two main classes, the Krylov-like projection

methods, e.g. [18], [30], and the methods based on the low-rank ADI iteration proposed in [22]. In both classes, one of the basic operations depends on solving linear equation systems involving A.

The splitting methods constitute a new class of methods, for which the basic operation is solving $\dot{x} = Ax$ over the same short time interval as the time step of the method. Using a standard implicit scheme for this only requires the solution of one or two linear equation systems, which is less than or equal to what is commonly needed in the projection- or ADI-based methods. If the properties of A can be utilized to employ a better tailored method, the efficiency can further be much improved. Hence we expect the splitting methods to be very competitive. A further advantage of the proposed splitting methods is that they both yield approximations that are positive semi-definite, like the solution to (1). This is in contrast to e.g. the BDF and Rosenbrock methods, and in fact to any interesting "one-step" or linear multi-step method, which only have this property for the methods of order one [13].

A brief outline of this note is as follows. In Section II we present the assumptions on the involved matrices and give the solutions to the subproblems. Section III demonstrates how to formulate the splitting methods in a low-rank context. The extension to generalized Riccati equations is done in Section IV, and finally Section V presents the results of two numerical experiments.

II. Problem setting

We make the following assumption on the matrices in (1):

Assumption 1. *The matrices A, Q and S all belong to $\mathbb{R}^{N\times N}$. Further, P_0, Q and S are all symmetric and positive semi-definite.*

We note that Assumption 1 is fulfilled in the LQR setting if $B \in \mathbb{R}^{N\times m}$ for $m \leq N$, and $R \in \mathbb{R}^{m\times m}$ is positive definite. Further, Assumption 1 guarantees the existence of a unique symmetric positive semi-definite solution $P(t)$ for $t \geq 0$, see e.g. [1, Theorem 4.1.6]. Also note that while we assume that the problem is autonomous, the results extend in a straightforward manner also to the time-dependent case.

Consider now the exact solutions to the subproblems (5) and (6). By differentiation, one confirms at once that

$$\mathcal{T}_{\mathcal{F}}(t)P_0 = \mathrm{e}^{tA^\mathsf{T}}P_0\mathrm{e}^{tA} + \int_0^t \mathrm{e}^{sA^\mathsf{T}}Q\mathrm{e}^{sA}\mathrm{d}s. \tag{8}$$

For the nonlinear subproblem, we have the following lemma:

Lemma 1. *Let Assumption 1 be fulfilled. Then the solution $\mathcal{T}_{\mathcal{G}}(t)P_0$ to the problem* (6) *with $t \geq 0$ is unique and given by*

$$\mathcal{T}_{\mathcal{G}}(t)P_0 = (I + tP_0S)^{-1}P_0.$$

Proof. We first note that since P_0 and S are both symmetric positive semi-definite, there exists a nonnegative, diagonal matrix D and a nonsingular matrix V such that $P_0S = VDV^{-1}$, see e.g. [19, Corollary 7.6.2b]. Thus the function

$$P(t) = (I + tP_0S)^{-1}P_0 = V(I + tD)^{-1}V^{-1}P_0$$

is well defined for all nonnegative times t, and uniformly bounded by $\|V\|\|V^{-1}\|\|P_0\|$. As $(I+tP_0S)P(t) = P_0$ is constant, we therefore get

$$\begin{aligned} 0 &= \big(I + (t+h)P_0S\big)P(t+h) - (I + tP_0S)P(t) \\ &= (I + tP_0S)\big(P(t+h) - P(t)\big) + hP_0SP(t+h), \end{aligned}$$

and letting $h \to 0$ yields the continuity of $P(t)$. Dividing both sides of the above equality by h further yields the existence of the limit

$$\lim_{h\to 0} \frac{P(t+h) - P(t)}{h} = -(I + tP_0S)^{-1}P_0SP(t),$$

from which we conclude that $P(t)$ is differentiable and satisfies (6). That this solution is unique e.g. follows from taking $A = Q = 0$ in [1, Theorem 4.1.6]. □

III. Low-rank splitting methods

In order to implement the splitting schemes efficiently, we consider now the low-rank formulation of the solution operators $\mathcal{T}_{\mathcal{F}}(h)$ and $\mathcal{T}_{\mathcal{G}}(h)$. We assume throughout this section that there are low-rank factorizations z and Q_f of P_0 and Q, respectively, i.e.

$$P_0 = zz^\mathsf{T} \quad \text{and} \quad Q = Q_fQ_f^\mathsf{T}.$$

A. The affine subproblem

By Equation (8) we have

$$\begin{aligned} &\mathrm{e}^{hA^\mathsf{T}}zz^\mathsf{T}\mathrm{e}^{hA} + \int_0^h \mathrm{e}^{sA^\mathsf{T}}Q_fQ_f^\mathsf{T}\mathrm{e}^{sA}\mathrm{d}s \\ &= \big(\mathrm{e}^{hA^\mathsf{T}}z\big)\big(\mathrm{e}^{hA^\mathsf{T}}z\big)^\mathsf{T} + \int_0^h \big(\mathrm{e}^{\tau A^\mathsf{T}}Q_f\big)\big(\mathrm{e}^{\tau A^\mathsf{T}}Q_f\big)^\mathsf{T}\mathrm{d}\tau. \end{aligned} \tag{9}$$

Denote the integral $I_Q(h)$ and consider an approximation by a quadrature formula with weights w_k and nodes τ_k:

$$I_Q(h) \approx h\sum_{k=0}^{s} w_k\big(\mathrm{e}^{\tau_kA^\mathsf{T}}Q_f\big)\big(\mathrm{e}^{\tau_kA^\mathsf{T}}Q_f\big)^\mathsf{T}.$$

Then we also have $I_Q(h) \approx \hat{y}\hat{y}^\mathsf{T}$, where

$$\hat{y} = \Big[\sqrt{hw_1}\mathrm{e}^{\tau_1A^\mathsf{T}}Q_f, \sqrt{hw_2}\mathrm{e}^{\tau_2A^\mathsf{T}}Q_f, \ldots, \sqrt{hw_s}\mathrm{e}^{\tau_sA^\mathsf{T}}Q_f\Big].$$

By this notation we mean that the s matrices $\sqrt{hw_j}\mathrm{e}^{\tau_jA^\mathsf{T}}Q_f$ are placed side by side. This new matrix $\hat{y}$ has more columns than either z or Q_f, and likely also more than its rank. A better low-rank candidate $I_Q(h) \approx yy^\mathsf{T}$ can be found by applying a column-compression technique. For instance, consider applying the rank-revealing QR factorization (RRQR) to $\hat{y}^\mathsf{T}$. This yields a factorization

$$\hat{y}^\mathsf{T}\tilde{P} = Q\begin{bmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{bmatrix}$$

where $\tilde{P}$ is a permutation matrix, $R_{11} \in \mathbb{R}^{r\times r}$, the norm of R_{22} is small, and Q is orthogonal. Thus (in MATLAB notation),

$$\hat{y}^\mathsf{T} \approx Q_{:,1:r}\begin{bmatrix} R_{11} & R_{12} \end{bmatrix}\tilde{P}^\mathsf{T},$$

so

$$\hat{y}\hat{y}^\mathsf{T} \approx yy^\mathsf{T} = \Big(\tilde{P}\begin{bmatrix} R_{11} & R_{12} \end{bmatrix}^\mathsf{T}\Big)\Big(\tilde{P}\begin{bmatrix} R_{11} & R_{12} \end{bmatrix}^\mathsf{T}\Big)^\mathsf{T},$$

where y belongs to $\mathbb{R}^{N\times r}$. Finally, the low-rank approximation ww^T to $\mathcal{T}_{\mathcal{F}}(h)P_0$ is given by forming

$$\hat{w} = \Big[\mathrm{e}^{hA^\mathsf{T}}z,\, y\Big]$$

and again employing column compression to achieve $ww^\mathsf{T} \approx \hat{w}\hat{w}^\mathsf{T}$. In practice, one would use a high-order quadrature formula and a RRQR tolerance which is small enough that the corresponding errors are negligible compared to the local error of the method.

We need to compute terms of the form $\mathrm{e}^{sA^\mathsf{T}}z$ efficiently in the above procedure. This can be done in many different ways, essentially divided into two categories. The first category consists of numerical linear algebra techniques where $\mathrm{e}^{sA^\mathsf{T}}z$ is treated as a matrix-vector product. See e.g. [12] for a recent comparison of four common approaches in the large-scale case. In the second category are methods that consider $\mathrm{e}^{sA^\mathsf{T}}z$ as the solution $x(s)$ to the system of ODEs $\dot{x} = A^\mathsf{T}x$, $x(0) = z$. Here we refer to [16]. We note that employing a standard implicit Runge-Kutta method to $\dot{x} = A^\mathsf{T}x$ is efficient enough

for the proposed methods to be competitive, as this means that the basic computation only requires the solution of a few linear equation systems involving A. However, the efficiency can be much improved if the problem possesses special structural properties. For example, if A is the discretization of a differential operator on a simple domain, then pseudo-spectral methods based on the FFT [11] or dimension splitting [20] can yield a vast improvement. In summary, the specific method needs to be chosen according to the characteristics of A, but even a non-optimal choice will yield good performance.

B. The nonlinear subproblem

Now consider the nonlinear subproblem (6). By Lemma 1, the solution is given by

$$\mathcal{T}_{\mathcal{G}}(h)P_0 = (I + hzz^{\mathsf{T}}S)^{-1}zz^{\mathsf{T}}.$$

We rewrite this expression by using the following special case of the Woodbury matrix inversion formula [14]:

$$(I + YZ)^{-1} = I - Y(I + ZY)^{-1}Z \tag{10}$$

This equality can be shown by simply multiplying with $I+YZ$ from the left and from the right. We set $Y = hz$ and $Z = z^{\mathsf{T}}S$, which yields

$$\begin{aligned}\mathcal{T}_{\mathcal{G}}(h)P_0 &= \big(I - hz(I + hz^{\mathsf{T}}Sz)^{-1}z^{\mathsf{T}}S\big)zz^{\mathsf{T}} \\ &= z\big(I - (I + hz^{\mathsf{T}}Sz)^{-1}hz^{\mathsf{T}}Sz\big)z^{\mathsf{T}} \\ &= z(I + hz^{\mathsf{T}}Sz)^{-1}z^{\mathsf{T}}.\end{aligned}$$

The computation of $z^{\mathsf{T}}Sz$ is cheap in many cases, e.g. if a low-rank or Cholesky factorization for S is available or cheaply computable, or if sparsity structure can be utilized. At the worst, the cost is that of r matrix-vector multiplications if $z \in \mathbb{R}^{N\times r}$. Note that in the LQR case, we have $z^{\mathsf{T}}Sz = z^{\mathsf{T}}BR^{-1}B^{\mathsf{T}}z$. Since $B \in \mathbb{R}^{N\times m}$ and $R \in \mathbb{R}^{m\times m}$ where generally $m \ll N$, the computation is certainly efficient in this case. Given $z^{\mathsf{T}}Sz$, we can Cholesky factorize the inner matrix as

$$I + hz^{\mathsf{T}}Sz = LL^{\mathsf{T}}.$$

This means that

$$\mathcal{T}_{\mathcal{G}}(h)P_0 = (zL^{-\mathsf{T}})(zL^{-\mathsf{T}})^{\mathsf{T}},$$

which is the sought after low-rank factorization of $\mathcal{T}_{\mathcal{G}}(h)P_0$. Note that it has the same rank as P_0, in contrast to the affine subproblem where the approximation can be of higher rank than P_0.

To summarize, we outline the procedure for taking one Lie splitting time step in Algorithm 1. This depends on the low-rank factor of $I_Q(h)$, which only needs to be computed once. For completeness, we summarize also this computation in Algorithm 2. We omit presenting the Strang splitting in algorithmic form as it is very similar to Algorithm 1.

Algorithm 1 Computing the low-rank factor of $\mathcal{L}_h P$

Input: Low-rank factors z and y such that $P = zz^{\mathsf{T}}$ and yy^{T} approximates the integral $I_Q(h)$ in (9)
Solve $\dot{x}(t) = A^{\mathsf{T}}x(t)$, $x(0) = z$ for $t \in [0, h]$
Form $\tilde{w} = [x(h), y]$
Column-compress $w \approx \tilde{w}$ by e.g. RRQR
Cholesky factorize $I + hw^{\mathsf{T}}Sw =: LL^{\mathsf{T}}$
Solve $xL^{\mathsf{T}} = w$
Output: x

Algorithm 2 Computing the low-rank factor of $I_Q(h)$

Input: Low-rank factor Q_f such that $Q = Q_f Q_f^{\mathsf{T}}$. Quadrature weights w_k and nodes τ_k for $k = 0, \ldots, s$.
for $k = 0$ to s **do**
Solve $\dot{x_k}(t) = A^{\mathsf{T}}x_k(t)$, $x_k(0) = Q_f$ for $t \in [0, \tau_k]$
end for
Form $\tilde{y} = \big[\sqrt{hw_1}x_1(h), \sqrt{hw_2}x_2(h), \ldots, \sqrt{hw_s}x_s(h)\big]$
Column-compress $y \approx \tilde{y}$ by e.g. RRQR
Output: y

IV. Generalized Riccati equations

In many LQR problems, the state equation is instead of the form

$$M\dot{x} = Ax + Bu,$$

where M is a mass matrix arising from a finite element discretization. We will assume that M is invertible, to avoid the extra difficulties connected with differential-algebraic equations, see [21] for this case. We can thus theoretically convert the state equation to the usual form,

$$\dot{x} = M^{-1}Ax + M^{-1}Bu. \tag{11}$$

However, even if both A and M are sparse matrices, $M^{-1}A$ is usually a dense matrix, so in practice we do not want to do this. Instead we seek a way to formulate our methods to implicitly handle the matrix M. The Riccati equation corresponding to (11) is

$$\begin{aligned}\dot{\tilde{P}}(t) = {} & A^{\mathsf{T}}M^{-\mathsf{T}}\tilde{P}(t) + \tilde{P}(t)M^{-1}A + Q \\ & - \tilde{P}(t)M^{-1}BR^{-1}B^{\mathsf{T}}M^{-\mathsf{T}}\tilde{P}(t), \quad \tilde{P}(0) = \tilde{P}_0,\end{aligned}$$

with the feedback $u^*(t) = -R^{-1}B^{\mathsf{T}}M^{-\mathsf{T}}\tilde{P}(T-t)x(t)$. By setting $P = M^{-\mathsf{T}}\tilde{P}M^{-1}$ (cf. [29]) we see that we can reformulate this as the generalized Riccati equation

$$\begin{aligned}M^{\mathsf{T}}\dot{P}(t)M = {} & A^{\mathsf{T}}P(t)M + M^{\mathsf{T}}P(t)A + Q \\ & - M^{\mathsf{T}}P(t)BR^{-1}B^{\mathsf{T}}P(t)M,\end{aligned} \tag{12}$$

with the initial condition $P(0)$ satisfying $M^{\mathsf{T}}P(0)M = \tilde{P}_0$, and with the feedback $u^*(t) = -R^{-1}B^{\mathsf{T}}P(T-t)Mx(t)$.

Consider now splitting (12) into its affine and nonlinear parts, as for (1) previously. By cancelling the factors M^{T} and M we see at once that the nonlinear subproblem is the same. Similarly to Equation (8), by differentiation one sees that the solution to the affine subproblem is given by

$$\begin{aligned}P(t) = {} & \mathrm{e}^{hM^{-\mathsf{T}}A^{\mathsf{T}}}P_0\,\mathrm{e}^{hAM^{-1}} \\ & + \int_0^h \mathrm{e}^{sM^{-\mathsf{T}}A^{\mathsf{T}}}M^{-\mathsf{T}}QM^{-1}\mathrm{e}^{sAM^{-1}}\,\mathrm{d}s.\end{aligned} \tag{13}$$

Thus, the only necessary change to the algorithm is to instead solve $M^{\mathsf{T}}\dot{x} = A^{\mathsf{T}}x$, $x(0) = z$, over $[0, h]$ to compute $\mathrm{e}^{hM^{-\mathsf{T}}A^{\mathsf{T}}}z$, and to solve $M^{\mathsf{T}}\dot{x} = A^{\mathsf{T}}x$, $x(0) = M^{-\mathsf{T}}Q_f$, over different intervals $[0, s]$ to approximate the integral. Note that the initial data is transformed by M in the second case, but not in the first.

V. Numerical experiments

Example 1. As a first example, we study a linear quadratic regulator problem as described in Section I. We describe first the continuous version of the problem. Let $\Omega = (0, 1)$ and let the state space be $H = L^2(\Omega)$. We take A to be the Laplacian $\Delta : \mathcal{D}(A) \subset H \to H$ with periodic boundary conditions. The control space will be $U = \mathbb{R}^m$ with $m = 10$, and $B : U \to H$ will be the sum of m evenly spaced interval sources:

$$Bu = \sum_{j=1}^{m} u_j \chi_{[x_j, x_j + 1/(4m)]},$$

where $x_j = j/m$. Thus B is a linear bounded operator. We take $R \in \mathbb{R}^{m\times m}$ to be the identity matrix.

To define C, we choose first the real trigonometric orthonormal basis for H: $\{1\} \cup \{e_k\}_{k=1}^{\infty} \cup \{f_k\}_{k=1}^{\infty}$, where

$$e_k(x) = \sqrt{2}\cos(2\pi kx) \quad \text{and} \quad f_k(x) = \sqrt{2}\sin(2\pi kx).$$

Then we set

$$C\Big(a_0 + \sum_{k=0}^{\infty} a_k e_k + b_k f_k\Big) = a_0 + \sum_{k=0}^{m_c} a_k e_k + b_k f_k,$$

for a small m_c, i.e. we simply truncate the sum. This C can be thought of as representing measuring equipment that can only measure low-frequency signals.

We discretize the problem in space by $2M+1$ equidistantly spaced nodes, corresponding to $2M+1$ frequencies in the spectral domain, and take $M = 1000$. The discretization of C has a natural low-rank factorization cc^{T} in the spectral basis, where c is a matrix of dimension $(2M+1)\times(2m_c+1)$. In order to work in the same basis we instead consider $E^{\mathsf{T}}cc^{\mathsf{T}}E$, where E denotes the orthogonal transformation matrix between the two different bases. Let Q_M be the discretization of $Q = C^{\mathsf{T}}C$. Since

$$Q_M = (E^{\mathsf{T}}cc^{\mathsf{T}}E)^{\mathsf{T}}(E^{\mathsf{T}}cc^{\mathsf{T}}E) = E^{\mathsf{T}}cc^{\mathsf{T}}cc^{\mathsf{T}}E = E^{\mathsf{T}}cc^{\mathsf{T}}E,$$

we can also low-rank factorize $Q_M = ww^{\mathsf{T}}$ with $w = E^{\mathsf{T}}c$. For this experiment we choose $m_c = 4$, which yields a matrix of low rank. We use the initial condition $P_0 = 0$, corresponding to the cost functional given in the introduction, but similar results are obtained for other values of P_0.

The Lie and Strang splitting schemes were implemented in MATLAB according to Algorithm 1 (with a slight modification for the Strang case) and applied to the problem described above. In order to compute expressions of the form $\mathrm{e}^{sA^{\mathsf{T}}}z$, FFT was used. Further, approximating the integrals $I_Q(h)$ was done using 14th-order Gauss quadrature, and the RRQR tolerance was set to 10^{-15}.

In Figure 1 we see that the methods exhibit the expected error behaviour: the Lie splitting converges with order 1 and the Strang splitting with order 2. Note that we measure the errors in the Frobenius norm, $\|\cdot\|_{\mathrm{F}}$, both here and in Example 2. As computing $\mathcal{T}_{\mathcal{F}}(h)z$ is likely to be significantly more expensive than computing $\mathcal{T}_{\mathcal{G}}(h)z$, we expect the Strang splitting to be more efficient than the Lie splitting. This is confirmed in the efficiency plot Figure 2 which shows the achieved error against the computation time. Finally, Figure 3 shows that the rank of the approximation indeed does remain low, $r = 9$, while $n = 2001$.

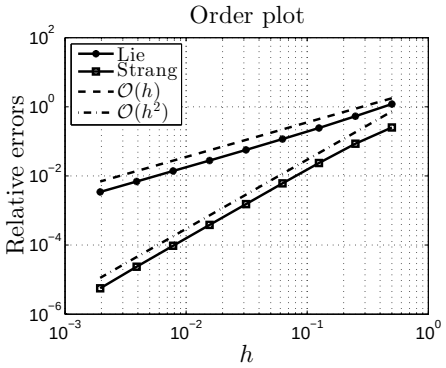

Fig. 1. The relative errors $\|\mathcal{L}_h^n P_0 - P_{\text{ref}}\|_{\mathrm{F}}/\|P_{\text{ref}}\|_{\mathrm{F}}$ and $\|\mathcal{S}_h^n P_0 - P_{\text{ref}}\|_{\mathrm{F}}/\|P_{\text{ref}}\|_{\mathrm{F}}$ when approximating the solution to (1) for Example 1. The different step sizes are $h = T/n$, with $n = 2, 4, \ldots, 512$ and $T = 1$. The reference solution P_{ref} was also computed by the Strang splitting, albeit with a finer temporal step size of $h = 1/2048$. The spatial discretization has $N = 2001$ nodes.

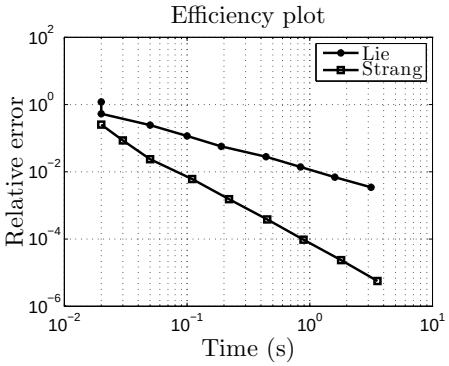

Fig. 2. The same experimental setup as in Figure 1 for Example 1, but the relative errors are now plotted against the computation time. We see that the Strang splitting outperforms the Lie splitting scheme for all error levels, as expected.

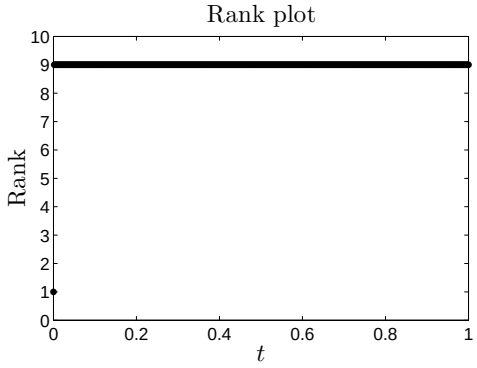

Fig. 3. The rank of $\mathcal{S}_h^n P_0$ when approximating the solution to (1) for Example 1. Here, $n = 0, \ldots, 512$ and $h = 1/512$. The spatial discretization has $N = 2001$ nodes. Note how the rank increases from 1 to 9 in the first step and then remains at this value throughout the rest of the integration.

Example 2. As a second example we consider the real-world application of optimal cooling of steel. We refer to [8], [28] for a detailed description of the problem. In short, a section of rail needs to be cooled after manufacturing. This is done by spraying a cooling fluid onto the outside of the rail. In order to preserve desirable material properties, the temperature differences inside the rail should be kept small. We thus have an optimal control problem, where the controls are chosen as the temperatures of the cooling fluid sprayed onto different parts of the boundary. The system is of the form

$$M\dot{x} = Ax + Bu,$$
$$y = Cx,$$

as in Section IV. Here, M and A arise from a finite element discretization of the cross section of the rail, and the operator C forms differences between the temperatures in certain nodes. By refining the finite element discretization, differently sized matrices can be acquired. For this experiment, we used the sizes $N = 1357$ and $N = 5177$, with the matrices $B \in \mathbb{R}^{N\times 7}$ and $C \in \mathbb{R}^{6\times N}$. We again used $Q = C^T C$, and $P_0 = 0$, but the weighting factor R for the input was set to $10^{-5}I$. The last two parameters are the final time, $T = 20$, and the RRQR tolerance, which was set to 10^{-10}. To approximate the solutions to the linear systems $M\dot{x} = Ax$ we used the third-order RadauIIA method, which is an implicit Runge–Kutta method.

Also in this problem we observe the expected error behaviour, as

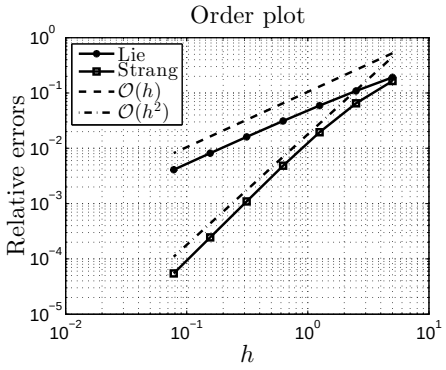

Fig. 4. The relative errors $\|\mathcal{L}_h^n P_0 - P_{\text{ref}}\|_{\text{F}}/\|P_{\text{ref}}\|_{\text{F}}$ and $\|\mathcal{S}_h^n P_0 - P_{\text{ref}}\|_{\text{F}}/\|P_{\text{ref}}\|_{\text{F}}$ when approximating the solution to (12) for Example 2 with $N = 1357$. The different step sizes are $h = T/n$, with $n = 4, 8, \ldots, 256$. The reference solution P_{ref} was also computed by the Strang splitting, albeit with a finer temporal step size of $h = 1/1024$.

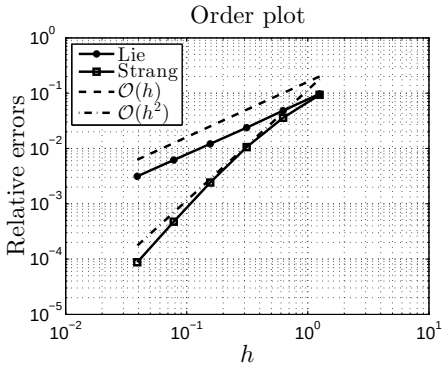

Fig. 5. The relative errors $\|\mathcal{L}_h^n P_0 - P_{\text{ref}}\|_{\text{F}}/\|P_{\text{ref}}\|_{\text{F}}$ and $\|\mathcal{S}_h^n P_0 - P_{\text{ref}}\|_{\text{F}}/\|P_{\text{ref}}\|_{\text{F}}$ when approximating the solution to (12) for Example 2 with $N = 5177$. The different step sizes are $h = T/n$, with $n = 16, 32, \ldots, 512$. The reference solution P_{ref} was also computed by the Strang splitting, albeit with a finer temporal step size of $h = 1/2048$.

seen in Figures 4 and 5, for $N = 1357$ and $N = 5177$, respectively. Figures 6 and 7 show the corresponding efficiency plots, where the Strang splitting again outperforms the Lie splitting. Figures 8 and 9 show the rank of the approximation over time for the two different cases. We note that while the rank is higher than in Example 1 it is still much less than the dimension of the problem. We also observe that it is of similar size regardless of the spatial discretization, which indicates that it is a property of the continuous version of the problem, and that further refinements of the grid will not substantially increase the necessary rank.

Finally, to further validate our results, we also computed the solution to Example 2 with $N = 1357$ using the second-order trapezoidal rule for time-stepping. We solved the resulting (dense) algebraic Riccati equations with MATLAB's built-in solver "care", a computation that took about four hours. Comparing this to the splitting computations, which required less than two minutes each, clearly demonstrates the benefits of the low-rank approach. We used the same step sizes for the different methods, which means that the difference between the trapezoidal method and the Lie splitting should be of size $\mathcal{O}(h)$, while comparing it to the Strang splitting should yield a difference of size $\mathcal{O}(h^2)$. In Figure 10, we show these (relative) errors over time. As the step size is $20/256 \approx 0.08$, these results correspond well with the expectations.

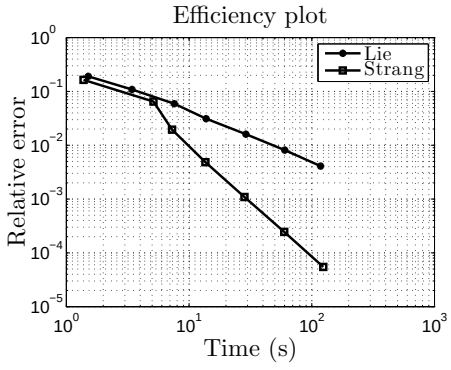

Fig. 6. The same experimental setup as in Figure 4, for Example 2 with $N = 1357$, but the relative errors are now plotted against the computation time. We see that the Strang splitting outperforms the Lie splitting scheme for all error levels.

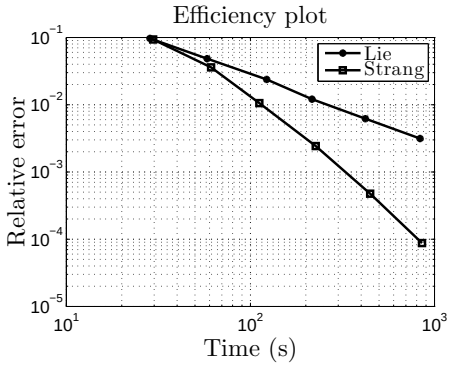

Fig. 7. The same experimental setup as in Figure 4, for Example 2 with $N = 5177$, but the relative errors are now plotted against the computation time. The Strang splitting outperforms the Lie splitting scheme for all error levels also here.

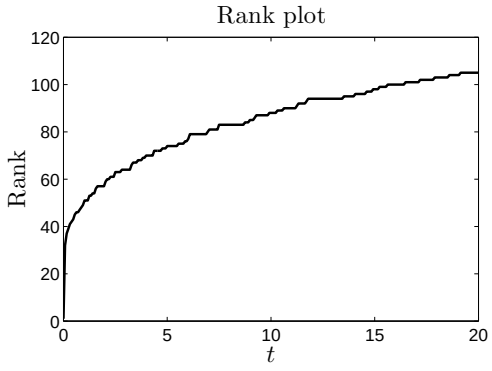

Fig. 8. The rank of $\mathcal{S}_h^n P_0$ when approximating the solution to (12) for Example 2 with $N = 1357$. Here, $n = 0, \ldots, 256$ and $h = 20/256$.

VI. Conclusions

We have shown how to efficiently implement low-rank splitting methods for the differential Riccati equation, as well as for the generalized version. The numerical experiments indicate that the methods converge with the expected orders. As the Strang splitting is essentially as cheap as the Lie splitting, while achieving better accuracy, we clearly recommend using the Strang splitting except for when the solution is not at all smooth. While the presented examples are not extremely large-scale, the results nevertheless indicate that increasing the problem dimensions even further should present no inherent difficulties. An in-depth comparison between the proposed

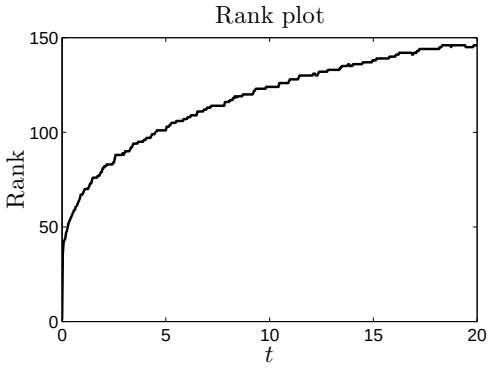

Fig. 9. The rank of $\mathcal{S}_h^n P_0$ when approximating the solution to (12) for Example 2 with $N = 5177$. Here, $n = 0, \ldots, 512$ and $h = 20/512$.

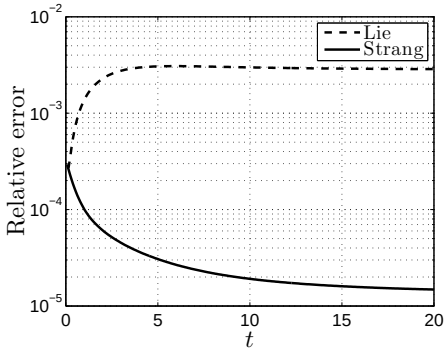

Fig. 10. The relative differences between the trapezoidal rule method and the two splitting methods over the time interval $[0, 20]$ when approximating the solution to (12) for Example 2 with $N = 1357$. The number of steps was $N = 256$, giving a step size of $h = 0.0781$.

splitting methods and projection-based or ADI-based methods is out of the scope of this technical note, but it is subject to ongoing work which will be published elsewhere.

ACKNOWLEDGMENT

The author would like to thank Eskil Hansen and the anonymous referees for valuable input in the preparation of this work, and Hermann Mena for providing him with Example 2.

REFERENCES

[1] H. Abou-Kandil, G. Freiling, V. Ionescu, G. Jank, *Matrix Riccati Equations in Control and Systems Theory*. Basel: Birkhäuser, 2003.
[2] K. J. Åström, *Introduction to stochastic control theory*. New York: Academic Press, 1970.
[3] P. Benner, "Solving Large-Scale Control Problems," *IEEE Contr. Syst. Mag.*, vol. 24, no. 1, pp. 44–59, 2004.
[4] P. Benner, Z. Bujanović, "On the solution of large-scale algebraic Riccati equations by using low-dimensional invariant subspaces," *MPI Magdeburg Preprints*, MPIMD/14-15, 2014.
[5] P. Benner, J.-R. Li, T. Penzl, "Numerical solution of large-scale Lyapunov equations, Riccati equations, and linear-quadratic optimal control problems," *Numer. Linear Algebra Appl.*, vol. 15, no. 9, pp. 755–777, 2008.
[6] P. Benner, H. Mena, "Numerical solution of the infinite-dimensional LQR-problem and the associated differential Riccati equations," *MPI Magdeburg Preprints*, MPIMD/12-13, 2012.
[7] P. Benner, H. Mena, "Rosenbrock methods for solving Riccati differential equations," *IEEE Trans. Automat. Control*, vol. 58, no. 11, pp. 2950–2956, 2013.
[8] P. Benner, J. Saak, "A semi-discretized heat transfer model for optimal cooling of steel profiles", in *Dimension Reduction of Large-Scale Systems* (vol. 45 of Lecture Notes in Computational Science and Engineering), P. Benner, V. Mehrmann, and D. Sorensen, Eds. Berlin/Heidelberg, Germany: Springer, 2005, pp. 353-356.
[9] P. Benner, J. Saak, "Numerical solution of large and sparse continuous time algebraic matrix Riccati and Lyapunov equations: A state of the art survey," *GAMM-Mitt.*, vol. 36, pp. 32–52, 2013.
[10] R. R. Bitmead, M. Gevers, "Riccati difference and differential equations: convergence, monotonicity and stability," in *The Riccati Equation* (Comm. Control. Engrg. Ser.), S. Bittanti, A. J. Laub, J. C. Willems, Eds. Berlin: Springer, 1991, pp. 263–291.
[11] J. P. Boyd, *Chebyshev and Fourier spectral methods*. New York: Dover, 2001.
[12] M. Caliari, P. Kandolf, A. Ostermann, S. Rainer, "Comparison of software for computing the action of the matrix exponential," *BIT Numer. Math.*, vol. 54, pp. 113–128, 2014.
[13] L. Dieci, T. Eirola, "Positive definiteness in the numerical solution of Riccati differential equations," *Numer. Math.*, vol. 67, no. 3, pp. 303–313, 1994.
[14] W. W. Hager, "Updating the inverse of a matrix," *SIAM Rev.*, vol. 31, no. 2, pp. 221–239, 1989.
[15] E. Hairer, C. Lubich, G. Wanner, *Geometric Numerical Integration* (Springer Series in Comput. Math. 31). Berlin: Springer, 2006.
[16] E. Hairer, G. Wanner, *Solving ordinary differential equations*, vol. 2. Berlin: Springer, 2010.
[17] E. Hansen, T. Stillfjord, "Convergence analysis for splitting of the abstract differential Riccati equation," *SIAM J. Numer. Anal.*, vol. 52, no. 6, pp. 3128–3139, 2014.
[18] M. Heyouni, K. Jbilou, "An extended block Arnoldi algorithm for large-scale solutions to the continuous-time algebraic Riccati equation," *Electron. T. Numer. Ana.*, vol. 33, pp. 53-62, 2009.
[19] R. A. Horn, C. R. Johnson, *Matrix Analysis*, 2nd ed. Cambridge: Cambridge University Press, 1985.
[20] W. Hundsdorfer, J. G. Verwer, *Numerical solution of time-dependent advection-diffusion-reaction equations* (Springer Series in Comput. Math. 33). Berlin: Springer, 2003.
[21] P. Kunkel, V. Mehrmann, "Numerical solution of differential algebraic Riccati equations," *Linear Algebra Appl.*, vol. 137/138, pp. 39–65, 1990.
[22] J.-R. Li, J. White, "Low rank solution of Lyapunov equations," *SIAM J. Matrix Anal. Appl.*, vol. 24, no. 1, pp. 260–280, 2002.
[23] J. L. Lions, *Optimal Control of Systems Governed by Partial Differential Equations*. Berlin: Springer, 1971.
[24] R. I. McLachlan, G. R. W. Quispel, "Splitting methods," *Acta Numer.*, vol. 11, pp. 341–434, 2002.
[25] H. Mena, "Numerical Solution of Differential Riccati Equations Arising in Optimal Control Problems for Parabolic Partial Differential Equations," Ph.D. dissertation, Escuela Politécnica Nacional, Quito, Ecuador, 2012.
[26] T. Penzl, "Eigenvalue decay bounds for solutions of Lyapunov equations: the symmetric case," *Syst. Control Lett.*, vol. 40, no. 2, pp. 139–144, 2000.
[27] W. T. Reid, *Riccati differential equations*. New York: Academic Press, 1972.
[28] J. Saak, "Effiziente numerische Lösung eines Optimalsteuerungsproblems für die Abkühlung von Stahlprofilen," Diplomarbeit, Univ. Bremen, Bremen, Germany, 2003.
[29] J. Saak, "Efficient Numerical Solution of Large Scale Algebraic Matrix Equations in PDE Control and Model Order Reduction," Ph.D. dissertation, Chemnitz Univ., Chemnitz, Germany, 2009.
[30] V. Simoncini, "A new iterative method for solving large-scale Lyapunov matrix equations," *SIAM J. Sci. Comput.*, vol. 29, no. 3, pp. 1268–1288, 2007.
[31] V. Simoncini, "Computational methods for linear matrix equations," preprint, Università di Bologna, 2014.
[32] D. C. Sorensen, Y. Zhou, "Bounds on eigenvalue decay rates and sensitivity of solutions to Lyapunov equations," Dept. of Comp. Appl. Math., Rice Univ., Houston, TX, Tech. Rep. TR02-07, June, 2002.
[33] G. Strang, "On the construction and comparison of difference schemes," *SIAM J. Numer. Anal.*, vol. 5, no. 3, pp. 506–517, 1968.
[34] R. Temam, "Sur l'équation de Riccati associée à des opérateurs non bornés, en dimension infinie," *J. Funct. Anal.*, vol. 7, pp. 85–115, 1971.

Tony Stillfjord is a Ph.D. student of Numerical Analysis at Lund University, Sweden.