Journal of Computational and Applied Mathematics 340 (2018) 602–614

Contents lists available at ScienceDirect

Journal of Computational and Applied Mathematics

journal homepage: www.elsevier.com/locate/cam

Numerical low-rank approximation of matrix differential equations

Hermann Mena [a,b], Alexander Ostermann [a], Lena-Maria Pfurtscheller [a], Chiara Piazzola [a,*]

[a] *Institut für Mathematik, Universität Innsbruck, Austria*
[b] *Department of Mathematics, Yachay Tech, Urcuquí, Ecuador*

A R T I C L E   I N F O

Article history:
Received 29 May 2017
Received in revised form 17 January 2018

MSC:
65L05
65F30
49J20

Keywords:
Dynamical low-rank approximation
Differential Lyapunov equations
Differential Riccati equations
Linear quadratic regulator problem
Splitting integrators
El Niño simulation

A B S T R A C T

The efficient numerical integration of large-scale matrix differential equations is a topical problem in numerical analysis and of great importance in many applications. Standard numerical methods applied to such problems require an unduly amount of computing time and memory, in general. Based on a dynamical low-rank approximation of the solution, a new splitting integrator is proposed for a quite general class of stiff matrix differential equations. This class comprises differential Lyapunov and differential Riccati equations that arise from spatial discretizations of partial differential equations. The proposed integrator handles stiffness in an efficient way, and it preserves the symmetry and positive semidefiniteness of solutions of differential Lyapunov equations. Numerical examples that illustrate the benefits of this new method are given. In particular, numerical results for the efficient simulation of the weather phenomenon El Niño are presented.

1. Introduction

Matrix differential equations arise quite naturally in many applications in science and engineering. Perhaps the most studied matrix differential equations are differential Riccati (DREs) and differential Lyapunov equations (DLEs) as they play a crucial role in many applications. For instance, they arise in optimal control problems like linear quadratic regulator and linear quadratic Gaussian design, H_∞ control of linear time-varying systems, optimal filtering, differential games, model reduction of linear time-varying systems, damping optimization in mechanical systems, and control of shear flows subject to stochastic excitations. For an overview and more details, we refer to [1–3].

In the literature, there is a large variety of approaches to compute the solution of DREs and DLEs, see, e.g., [4,5]. However, due to efficiency reasons, standard methods are not suitable for large-scale problems, in general. A major source of large-scale problems are partial differential equations (PDEs). In two and three space dimensions, their spatial discretization leads to systems with a significant number of degrees of freedom. The system matrices have often a particular structure, and they are usually sparse. In general, the arising differential equations are stiff which in turn requires integrators that handle the stiffness in an efficient way.

The problem of solving large-scale DREs/DLEs has recently received considerable attention. Various integrators based on low-rank approximations have been developed. In particular, in [6,7] the authors proposed efficient numerical methods

* Corresponding author.
E-mail addresses: mena@yachaytech.edu.ec (H. Mena), alexander.ostermann@uibk.ac.at (A. Ostermann), lena-maria.pfurtscheller@uibk.ac.at (L.-M. Pfurtscheller), chiara.piazzola@uibk.ac.at (C. Piazzola).

https://doi.org/10.1016/j.cam.2018.01.035
0377-0427/© 2018 The Author(s). Published by Elsevier B.V. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

which are based on a matrix valued version of backward differentiation formulas (BDFs) and Rosenbrock methods, respectively. Further, a low-rank splitting method for large-scale DREs has been introduced in [8]. A different approach to compute low-rank approximations to large-scale matrix differential equations was proposed in [9]. It is the so-called dynamical low-rank approximation. Its strength relies on the fact that a low-rank approximation to the solution is computed by solving differential equations only for its low-rank factors, see also [10–12].

The performance of low-rank based methods relies on the decay of the singular values of the solution. This phenomenon has been deeply studied and is frequent in applications, see, e.g., [13,14] and references therein.

In this paper, we focus on a new approach for solving DREs and DLEs that arise in PDE based models. Following the ideas of [11,15] we develop an efficient algorithm based on a splitting integrator. We split the vector field into a linear stiff and a non-linear non-stiff part. The stiff problem is efficiently integrated by an exponential integrator. We employ here the Leja method which proves to be very competitive, see [15]. We combine this integrator with the dynamical low-rank approximation for the non-stiff part. In particular, we employ the projector-splitting scheme, proposed in [11]. This choice makes our low-rank integrator very robust with respect to small singular values. In this work we keep the rank fixed during time integration. Changing dynamically the rank would require a strategy for adapting the rank, based on a control of the corresponding error. Such a rank-adaptive approach is feasible but beyond the scope of our work. However, we illustrate the behaviour of our integrator with respect to different approximation ranks by numerical experiments.

The paper is organized as follows. In Section 2 we introduce the class of matrix differential equations considered and we propose our low-rank numerical integrator. In Sections 3 and 4, we specialize our integrator for DLEs and DREs, and we discuss a possible extension to generalized DREs. We give details on the implementation and present numerical results that support our chosen approach. Finally, some conclusions are given.

2. Matrix differential equations and numerical integrators

In this work we are interested in the numerical solution of the following class of matrix differential equations

$$\dot{X}(t) = AX(t) + X(t)A^{\mathsf{T}} + G(t, X(t)), \quad X(t_0) = X_0, \quad t \in [t_0, T], \tag{1}$$

where $X(t), A \in \mathbb{R}^{d\times d}$, $G : \mathbb{R} \times \mathbb{R}^{d\times d} \to \mathbb{R}^{d\times d}$, and $(\cdot)^{\mathsf{T}}$ denotes the transpose. This class includes, among others, differential Lyapunov and Riccati equations. In particular, we consider here problems stemming from PDEs. Therefore, the matrix A is typically the spatial discretization of a differential operator. The function G on the other hand is assumed to be non-stiff. By means of the variation-of-constants formula, the solution of (1) can be written as

$$X(t) = \mathrm{e}^{(t-t_0)A}X(t_0)\mathrm{e}^{(t-t_0)A^{\mathsf{T}}} + \int_{t_0}^{t} \mathrm{e}^{(t-s)A}G(s, X(s))\mathrm{e}^{(t-s)A^{\mathsf{T}}}\,\mathrm{d}s \tag{2}$$

for $t_0 \le t \le T$.

The stiffness of (1) is a limitation for most of the standard numerical integrators. Explicit methods have to use tiny step sizes whereas the use of implicit methods can result in high computational cost. Both approaches are thus not efficient. As a remedy, we design here a numerical integrator which is able to handle the stiff part of the solution (2) in an exact and efficient way. The key idea is the use of splitting methods. For an introduction to this class of numerical integrators we refer to [16]. The structure of (1) suggests a splitting into two terms, where we split the stiff linear part from the non-stiff non-linearity. Following this approach, two subproblems arise:

$$\dot{M}(t) = AM(t) + M(t)A^{\mathsf{T}}, \tag{3a}$$
$$\dot{N}(t) = G(t, N(t)), \tag{3b}$$

where $t \in [t_0, T]$. A clear advantage of this splitting approach is that the stiffness is only present in the first subproblem (3a), whereas the second equation (3b) is a non-stiff ordinary differential equation. Therefore, we have more freedom for the numerical integration of (3b). Depending on the way of recombining the partial flows of (3), we obtain splitting methods with different orders of convergence.

Let us denote with $\Phi_\tau^A(Z)$ the exact solution of (3a) at $t_0 + \tau$ with initial value $M(t_0) = Z$, i.e.,

$$\Phi_\tau^A(Z) = M(t_0 + \tau) = \mathrm{e}^{\tau A}Z\mathrm{e}^{\tau A^{\mathsf{T}}}. \tag{4}$$

Further, let $\Phi_\tau^G(Z)$ denote the exact solution of (3b) with initial value $N(t_0) = Z$, i.e., $\Phi_\tau^G(Z) = N(t_0+\tau)$. The simplest splitting integrator for solving (1) is then given by

$$\mathcal{L}_\tau Z = \Phi_\tau^G \circ \Phi_\tau^A(Z), \tag{5}$$

where $Z = X_0$. The result $\mathcal{L}_\tau X_0$ is the numerical approximation to $X(t_0 + \tau)$. This is a first-order method and called Lie splitting in the literature. A second-order method, the so-called Strang splitting, is given by the symmetric formula

$$\mathcal{S}_\tau Z = \Phi_{\tau/2}^A \circ \Phi_\tau^G \circ \Phi_{\tau/2}^A(Z). \tag{6}$$

The numerical evaluation of (4) can be computationally expensive if the dimension d is large. In this case we propose to approximate the action of the flow Φ_τ^A by means of a low-rank approximation. Note that the differential equation (3a) is rank preserving, i.e, the rank of the solution $M(t)$ does not depend on time (see [17, Lemma 1.22]). A rank-r decomposition ($r \le d$) of the initial data Z is typically given by a truncated singular value decomposition (SVD) of the form USV^{T}, where $U, V \in \mathbb{R}^{d\times r}$ have orthonormal columns and $S \in \mathbb{R}^{r\times r}$ is diagonal. Using this approximation for the initial data Z, we obtain a rank-r approximation M_1 to the exact solution $M(t_0+\tau) = \Phi_\tau^A(Z)$ by

$$M_1 = \mathrm{e}^{\tau A} USV^{\mathsf{T}} \mathrm{e}^{\tau A^{\mathsf{T}}}. \tag{7}$$

Note that the action of the matrix exponential $\mathrm{e}^{\tau A}$ to the skinny matrices U and V can be efficiently computed, for instance, by Taylor interpolation [18], interpolation at Leja points [15], and Krylov subspace methods [19,20]. We will compute these actions here with the Leja method. The resulting matrices $\mathrm{e}^{\tau A}U$ and $\mathrm{e}^{\tau A}V$ can be orthogonalized by means of a QR decomposition such that $\mathrm{e}^{\tau A}U = \widetilde{U}R$, where $\widetilde{U} \in \mathbb{R}^{d\times r}$ has orthonormal columns and $R \in \mathbb{R}^{r\times r}$. Similarly we have $\mathrm{e}^{\tau A}V = \widetilde{V}P$. After defining $\widetilde{S} = RSP^{\mathsf{T}}$, a rank-$r$ solution of (3a) is given as $M_1 = \widetilde{U}\widetilde{S}\widetilde{V}^{\mathsf{T}}$.

For the solution of (3b) we make use of the dynamical low-rank approach proposed in [9] (see also [10–12]). This is a differential equation based approach to efficiently compute low-rank approximations to time-dependent large matrices or to solutions of large-scale matrix differential equations. Let us denote by $\mathcal{M}_r = \mathcal{M}_r^{d\times d}$ the manifold of rank-r matrices of dimension $d \times d$. At any time t, a rank-r approximation to the solution $N(t) \in \mathbb{R}^{d\times d}$ of (3b) is a matrix $Y(t) \in \mathcal{M}_r$ defined by the following condition: for every t, the matrix $\dot{Y}(t) \in \mathcal{T}_{Y(t)}\mathcal{M}_r$ is such that

$$\|\dot{Y}(t) - G(t, Y(t))\|_F = \min, \tag{8}$$

where $\mathcal{T}_{Y(t)}\mathcal{M}_r$ is the tangent space of $\mathcal{M}_r$ at the current state $Y(t)$ and $\|\cdot\|_F$ denotes the Frobenius norm. The minimization condition leads to a differential equation for Y on the low-rank manifold, which has to be solved numerically. Standard integrators fail due to the presence of small singular values. Therefore, a particular integrator, the so-called projector-splitting integrator was developed in [11]. Its convergence for non-stiff G and its robustness with respect to the small singular values was proven in [10].

Condition (8) states that $\dot{Y}(t)$ is obtained by an orthogonal projection of $G(t, Y(t))$ onto the tangent space $\mathcal{T}_{Y(t)}\mathcal{M}_r$, i.e.

$$\dot{Y}(t) = P(Y(t))G(t, Y(t)),$$

where $P(Y)$ is this orthogonal projection. The matrix $Y(t)$ is represented as $U(t)S(t)V(t)^{\mathsf{T}}$, where $U, V \in \mathbb{R}^{d\times r}$ with orthonormal columns and $S \in \mathbb{R}^{r\times r}$. Note that S is not assumed to be diagonal. Using the explicit form of the projector, which was computed in [9], the evolution equation for the approximation Y is seen to be

$$\dot{Y} = G(t, Y)VV^{\mathsf{T}} - UU^{\mathsf{T}}G(t, Y)VV^{\mathsf{T}} + UU^{\mathsf{T}}G(t, Y). \tag{9}$$

The projector-splitting integrator itself makes use of splitting methods. The right-hand side of (9) is split up into three subproblems. The integrator only deals with differential equations for the low-rank factors U, S and V. The practical algorithm for a general non-linearity $G(t, Y)$ is given in [10, Sect. 2.2]. Starting from $U_0S_0V_0^{\mathsf{T}}$, a rank-r approximation to the initial data $N(t_0)$, one step of the integrator is as follows.

a. Solve $\dot{K}(t) = G(t, K(t)V_0^{\mathsf{T}})V_0$ with initial value $K(t_0) = U_0S_0$. Then orthonormalize $K(t_1)$ by a QR decomposition to get $K(t_1) = U_1\widehat{S}_1$, where $U_1 \in \mathbb{R}^{d\times r}$ has orthonormal columns and $\widehat{S}_1 \in \mathbb{R}^{r\times r}$.
b. Solve $\dot{S}(t) = -U_1^{\mathsf{T}}G(t, U_1S(t)V_0^{\mathsf{T}})V_0$ with initial value $S(t_0) = \widehat{S}_1$. Set $\widetilde{S}_0 = S(t_1)$.
c. Solve $\dot{L}(t) = G(t, U_1L(t)^{\mathsf{T}})^{\mathsf{T}}U_1$ with initial value $L(t_0) = V_0\widetilde{S}_0^{\mathsf{T}}$. Then orthonormalize $L(t_1)$ by a QR decomposition to get $L(t_1) = V_1S_1^{\mathsf{T}}$, where $V_1 \in \mathbb{R}^{d\times r}$ has orthonormal columns and $S_1 \in \mathbb{R}^{r\times r}$.

The above scheme can be tailored to the particular form of G. More details are given in the sections below. A detailed convergence analysis of the proposed integrator will be presented elsewhere.

3. Differential Lyapunov equations

Differential Lyapunov equations arise for $G(t, X) = Q$ with Q being a constant matrix. Thus, we consider the following initial value problem

$$\begin{aligned} \dot{X}(t) &= AX(t) + X(t)A^{\mathsf{T}} + Q, \\ X(t_0) &= X_0, \end{aligned} \tag{10}$$

where $X(t)$, A, $Q \in \mathbb{R}^{d\times d}$, and $t \in [t_0, T]$. The matrix A typically arises from the discretization of a differential operator. Further, Q and the initial data X_0 are symmetric and positive semidefinite. Since (10) is a linear differential equation with constant coefficients the solution exists for all times. Moreover, the solution is also symmetric and positive semidefinite. This is a straightforward consequence of (2), see also [1].

3.1. A low-rank split-step integrator

As explained in Section 2 we split (10) into the following two subproblems:

$$\dot{M}(t) = AM(t) + M(t)A^{\mathsf{T}}, \quad M(t_0) = M_0, \tag{11a}$$

$$\dot{N}(t) = Q, \quad N(t_0) = N_0, \tag{11b}$$

where $t \in [t_0, T]$. Note that the exact solution of (11b) simply is $N(t) = N_0 + tQ$. Employing this representation directly in our splitting method, however, would require one additional SVD per time step. To avoid this computational overhead, we follow the approach chosen in [10] and use the projector-splitting method.

Since the solution of (10) is symmetric and positive semidefinite we represent it as a rank-r matrix of the form USU^{T}, where $U \in \mathbb{R}^{d\times r}$ has orthonormal columns and $S \in \mathbb{R}^{r\times r}$ is symmetric. This is an SVD-like decomposition with the additional constraint of symmetry.

The linear problem (11a) can be treated as explained in Section 2. The projector-splitting integrator for the solution of (11b) in its standard formulation, however, does not preserve the symmetry and positive semidefiniteness of the solution. Let $U_0S_0U_0^{\mathsf{T}}$ be a low-rank decomposition of the initial data N_0. In order to preserve the symmetry of the problem we propose a modification of the algorithm given at the end of Section 2, see also [10, Sect. 2.2]. The modified projector-splitting integrator is described in step 5 of Algorithm 1. In the first substep we update the value of the left-sided orthonormal factor U_0 to U_1, whereas we keep the right-sided one till the last step. There we get $L(t_1) = S_1\widetilde{U}_1^{\mathsf{T}}$. Imposing $\widetilde{U}_1 = U_1$ and taking advantage of the orthonormality of U_1 we obtain $S_1 = L(t_1)U_1$. The low-rank approximation N_1 at $t_0 + \tau$ is then simply $U_1S_1U_1^{\mathsf{T}}$.

In Algorithm 1 we summarize the above proposed Lie splitting for the solution of (10). Note that the flows of all these substeps can be computed exactly. Moreover, the whole time integration is performed by working only with the low-rank factors of the solution, which has several computational advantages such as less computing time and less memory requirements. By using the explicit structure of the subproblems, it is easy to show that S_1 and consequently N_1 are symmetric and positive semidefinite.

Algorithm 1 The first-order low-rank split-step integrator for DLEs

```
1: Compute a symmetric rank-$r$ approximation $M_0$ to the given initial data $X_0$: $M_0 = U_0S_0U_0^{\mathsf{T}}$.
2: Let $t_k = t_0 + k\tau$ for $k \in \mathbb{N}$ and set $n = 0$.
3: Compute $e^{\tau A}U_n$ and orthogonalize it by a QR decomposition to get $e^{\tau A}U_n = U_n^AR$, where $U_n^A \in \mathbb{R}^{d\times r}$ and $R \in \mathbb{R}^{r\times r}$.
4: Define $S_n^A = RS_nR^{\mathsf{T}}$.
5: Given $U_n^A$ and $S_n^A$ perform one integration step:
    a. Solve $\dot{K}(t) = QU_n^A$ with initial value $K(t_n) = U_n^AS_n^A$. Then orthogonalize $K(t_{n+1})$ by a QR decomposition and set
       $U_{n+1}\widehat{S}_{n+1} = K(t_{n+1})$, where $U_{n+1} \in \mathbb{R}^{d\times r}$ has orthonormal columns and $\widehat{S}_{n+1} \in \mathbb{R}^{r\times r}$.
    b. Solve $\dot{S}(t) = -U_{n+1}^{\mathsf{T}}QU_n^A$ with initial value $S(t_n) = \widehat{S}_{n+1}$. Then set $\widetilde{S}_n = S(t_{n+1})$.
    c. Solve $\dot{L}(t) = U_{n+1}^{\mathsf{T}}Q$ with initial value $L(t_n) = \widetilde{S}_n(U_n^A)^{\mathsf{T}}$. Then set $S_{n+1} = L(t_{n+1})U_{n+1}$.
6: Increase $n$ by 1.
7: Go to 3 and repeat as long as $t_n < T$.
```

Lemma 1. *The matrix S_n arising in step 5c of Algorithm 1 is symmetric and positive semidefinite.*

Proof. Without loss of generality we prove the lemma for $n = 0$. For simplicity, we will also omit the upper index A and write U_0 instead of U_0^A, etc. Starting from the rank-r approximation $U_0S_0U_0^{\mathsf{T}}$ of N_0, the first step of the projector-splitting integrator gives

$$U_1\widehat{S}_1 = U_0S_0 + \tau QU_0. \tag{12}$$

The solution of the second substep is

$$\widetilde{S}_0 = \widehat{S}_1 - \tau U_1^{\mathsf{T}}QU_0.$$

From step 5c we get

$$S_1 = \widetilde{S}_0U_0^{\mathsf{T}}U_1 + \tau U_1^{\mathsf{T}}QU_1. \tag{13}$$

Using the orthonormality of U_1, we compute $\widehat{S}_1 = U_1^{\mathsf{T}}U_0S_0 + \tau U_1^{\mathsf{T}}QU_0$ from (12). Inserting this formula into (13) we get the symmetric formula

$$S_1 = U_1^{\mathsf{T}}U_0S_0U_0^{\mathsf{T}}U_1 + \tau U_1^{\mathsf{T}}QU_1$$

which proves the assertion. □

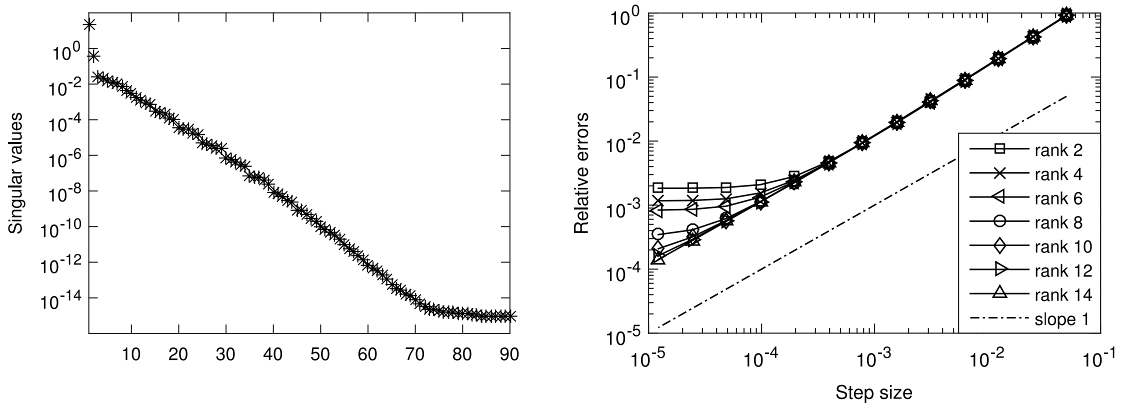

Fig. 1. Results for the DLE of Section 3.2 for $d = 400$. Left: First 90 singular values of the reference solution computed with DOPRI5 at $T = 0.1$. Right: Errors of Lie splitting described in Algorithm 1 in the Frobenius norm (14) as a function of step size and rank at $T = 0.1$.

3.2. Numerical results for a parabolic problem

In order to illustrate the behaviour of the integrator, we take a simple test problem. We consider the heat equation

$$\partial_t w = \Delta w, \qquad w|_{\partial\Omega} = 0$$

on $\Omega = (0, 1)^2$ with homogeneous Dirichlet boundary conditions. The associated DLE is of the form (10), where A arises from the spatial discretization of the Laplacian. Thus, we have $A = \widetilde{A} \otimes I + I \otimes \widetilde{A}$, where $\widetilde{A}$ is the matrix obtained from standard centred finite differences in 1D with $\widetilde{d}$ uniformly spaced grid points in the interior. In the following we take $\widetilde{d} = 20$ and obtain $d = \widetilde{d}^2 = 400$. The matrix Q in our example has random coefficients and is of rank 5. The initial value X_0 is chosen to be a random matrix of rank 10. This choice ensures the low-rank behaviour of the solution of the DLE. This can be observed in Fig. 1, left where we plot the first 90 singular values of a numerical solution computed with DOPRI5 [21] at the final integration time $T = 0.1$. This solution is taken as the reference solution for all tests in this section. The code DOPRI5 is an explicit Runge–Kutta method of order 5 with adaptive step size strategy.

In Fig. 1, right we show the error behaviour of the Lie splitting (5) as described in Algorithm 1. The errors are measured in an appropriately scaled Frobenius norm

$$\|Z\|_F = \frac{1}{d}\sqrt{\sum_{i,j=1}^{d} Z_{ij}^2}, \qquad Z \in \mathbb{R}^{d\times d}. \tag{14}$$

We observe that the error of our method is composed by two different contributions, the error due to the outer splitting into (11a) and (11b) and the error due to the low-rank approximation. As long as the error due to the low-rank approximation is not dominant, we observe the expected order of convergence one for Lie splitting. On the other hand, if the low-rank approximation is poor, decreasing the step size will not improve the quality of the solution. We observe a stagnation of the error around certain values depending on the rank. These errors can be related with the magnitude of the first singular value discarded for each choice of the approximation rank, as can be observed by comparing Fig. 1, left and right. In Fig. 2 we show the corresponding results for Strang splitting for $d = 400$ and $d = 3600$, respectively. On the left we observe the expected order of convergence two for the outer splitting. Again, when the approximation rank is too low, the outer error becomes independent of any step size refinement. An order reduction shows up in the figure on the right. It is due to the fact that the inhomogeneity Q is not in the domain of the Laplacian. Therefore, AQ cannot be bounded independently of d, see [22].

In Table 1 we list the defects in symmetry

$$d_{\text{sym}} = \frac{\|Y - Y^{\mathsf{T}}\|_F}{\|Y_{\text{ref}}\|_F}, \tag{15}$$

where Y is the solution obtained with Lie splitting as defined in Algorithm 1 and Y_{ref} is the reference solution. We observe that the symmetry is numerically preserved, as expected from Lemma 1. In Table 2 we show that the standard non-symmetric integrator as presented in Section 2 is not symmetry-preserving. It is therefore necessary to modify the projector-splitting as explained in detail in Algorithm 1.

Due to Lemma 1, our integrator also preserves positive semidefiniteness of the solution. To show this, we denote with $\widehat{Y}$ the nearest symmetric positive semidefinite matrix to Y obtained by the method proposed in [23]. In Table 3 we list the

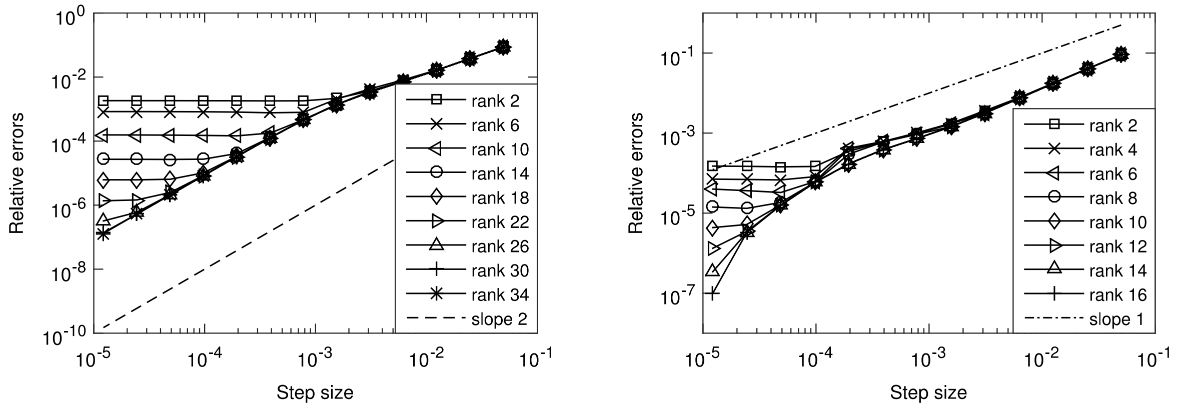

Fig. 2. Errors of Strang splitting for the solution of the DLE of Section 3.2. The errors are measured in the Frobenius norm (14) as a function of step size and rank at $T = 0.1$. Left: $d = 400$, right: $d = 3600$.

Table 1
Defects in symmetry for Algorithm 1 applied to the DLE of Section 3.2 for $d = 400$. The defect (15) is displayed as a function of the approximation rank (rk) and the number of time steps (ns) at $T = 0.1$.

	ns = 2	ns = 2^4	ns = 2^7	ns = 2^{10}	ns = 2^{13}
rk = 2	4.7294e−16	1.6989e−16	1.5067e−16	7.3221e−16	5.1379e−16
rk = 4	7.7743e−16	3.8092e−16	4.9242e−16	1.0861e−15	2.7226e−15
rk = 6	1.0498e−15	3.4863e−16	5.0365e−16	3.2494e−16	3.9935e−16
rk = 8	1.0781e−15	4.1657e−16	3.9016e−16	2.6837e−15	1.2762e−14
rk = 10	8.2787e−16	7.7903e−16	1.3127e−15	1.7855e−15	2.8277e−15
rk = 12	1.1677e−15	4.6731e−16	6.3586e−16	2.1097e−15	6.9804e−15
rk = 14	1.8577e−15	5.5461e−16	8.0892e−16	1.9972e−15	2.5059e−15

Table 2
Defects in symmetry for Lie splitting of Section 2 applied to the DLE of Section 3.2 for $d = 400$ without symmetry-preserving modification. The defect (15) is displayed as a function of the approximation rank (rk) and the number of time steps (ns) at $T = 0.1$.

	ns = 2	ns = 2^4	ns = 2^7	ns = 2^{10}	ns = 2^{13}
rk = 2	9.5178e−03	3.2545e−03	4.2299e−04	4.3455e−05	5.2796e−06
rk = 4	1.2097e−02	3.8332e−03	2.4587e−04	2.0065e−05	2.3394e−06
rk = 6	2.3350e−04	4.1849e−03	2.6199e−04	2.0735e−05	2.3876e−06
rk = 8	1.2875e−05	8.0839e−04	1.9463e−04	1.5981e−05	1.8315e−06
rk = 10	2.5457e−07	1.5020e−04	1.0291e−04	8.1713e−06	9.2641e−07
rk = 12	1.3272e−09	5.9891e−05	4.7831e−05	3.9946e−06	4.5438e−07
rk = 14	1.0354e−12	5.5624e−06	1.8540e−05	1.7506e−06	1.9895e−07

defects in positive semidefiniteness

$$d_{\text{psd}} = \frac{\|Y - \widehat{Y}\|_F}{\|Y_{\text{ref}}\|_F} \tag{16}$$

for different approximation ranks and step sizes.

3.3. Comparison with a standard approach

In order to give more insight into the performance of the low-rank splitting described in Algorithm 1 we carry out a comparison with a standard method. The basic idea is to discretize the DLE in time to convert the differential problem into an algebraic one. Since our aim is to compare the first-order low-rank splitting, we apply here the backward Euler method with time step τ to obtain the numerical approximation X_{n+1} to $X(t_{n+1})$. This gives

$$X_{n+1} = X_n + \tau\left(AX_{n+1} + X_{n+1}A^{\mathsf{T}} + Q\right).$$

After recombining the terms, we end up with the following algebraic Lyapunov equation (ALE)

$$0 = (X_n + \tau Q) + \left(\tau A - \frac{1}{2}I\right)X_{n+1} + X_{n+1}\left(\tau A^{\mathsf{T}} - \frac{1}{2}I\right). \tag{17}$$

Several approaches are available in the literature for solving this type of equations. We employ the one proposed in [24]. It consists of projecting the ALE onto an approximation space, generated as combination of Krylov subspaces in A and

Table 3
Defects in positive semidefiniteness for Algorithm 1 applied to the DLE of Section 3.2 for $d = 400$. The defect (16) is displayed as a function of the approximation rank (rk) and the number of time steps (ns) at $T = 0.1$.

	ns = 2	ns = 2^4	ns = 2^7	ns = 2^{10}	ns = 2^{13}
rk = 2	4.9230e−15	2.7584e−15	2.7871e−15	4.3101e−15	2.3554e−15
rk = 4	4.1822e−15	4.6147e−15	3.2650e−15	5.4743e−15	3.2069e−15
rk = 6	5.9181e−15	1.9928e−15	2.5106e−15	3.0357e−15	4.0661e−15
rk = 8	5.1317e−15	2.6077e−15	1.7991e−15	3.0388e−15	6.6778e−15
rk = 10	7.9000e−15	3.3707e−15	2.7210e−15	3.3223e−15	2.3704e−15
rk = 12	5.7540e−15	2.2482e−15	2.0631e−15	2.6968e−15	5.7000e−15
rk = 14	4.2693e−15	7.0395e−15	2.6659e−15	2.0693e−15	3.2132e−15

A^{-1}. Such an approach is also called extended Krylov method. The reduced equation is then solved by means of a direct solver.

In particular, consider an ALE of the form

$$0 = \widetilde{A}X + X\widetilde{A}^{\mathsf{T}} + \widetilde{B}\widetilde{B}^{\mathsf{T}}, \tag{18}$$

with $\widetilde{A} \in \mathbb{R}^{d\times d}$ and $\widetilde{B} \in \mathbb{R}^{d\times s}$, $s \ll d$. Let us denote with $\widetilde{V}$ the matrix whose columns span the approximation space and with Y the solution of the projected ALE. Then the solution of the original ALE is reconstructed as

$$X = \widetilde{V}Y\widetilde{V}^{\mathsf{T}}.$$

A low-rank solution is generated discarding all the eigenvalues of Y which are smaller than a certain tolerance `tolY`. Then the solution X is given in low-rank form as ZZ^{T}. A second user-supplied parameter, denoted by `tol`, is used for the stopping criterion of the algorithm. The iteration is stopped if

$$\frac{\|\widetilde{A}X^m + X^m\widetilde{A}^{\mathsf{T}} + \widetilde{B}\widetilde{B}^{\mathsf{T}}\|_2}{2\|\widetilde{A}\|_F\|Y^m\|_F + \|\widetilde{B}\|_F^2} \le \texttt{tol},$$

where the superscript m denotes the number of the iteration. The algorithm associated with these procedure is called K-PIK (Krylov-plus-inverted Krylov). For further details we refer to [24].

In the following we report some numerical experiments for the example we considered in the beginning of Section 3.2. We solve it until $T = 0.1$. An algebraic equation of the form (17) has to be solved for each time step. In order to use the K-PIK procedure we set

$$\widetilde{A} = \tau A - \frac{1}{2}I \quad \text{and} \quad \widetilde{B} = [\sqrt{\tau}R \;\; Z_n],$$

where Z_n is the low-rank factor of the solution X_n at time t_n, i.e., $X_n = Z_nZ_n^{\mathsf{T}}$ and R is the low-rank factor of Q, i.e., $Q = RR^{\mathsf{T}}$.

In Fig. 3 we illustrate the behaviour of this integrator for different values of `tolY`. The results are all obtained in Matlab with `tol` $= 10^{-12}$. The projected ALE is solved with Matlab's `lyap` function. Moreover, as a comparison, we compute the solution of (10) by directly solving (17) by the routine `lyap`. Since in both the cases the problem is discretized in time in the same way, this comparison reduces to a comparison between the K-PIK procedure and the `lyap` method. In Fig. 3, left we show the error behaviour of the two methods. The K-PIK procedure is tested for different values of `tolY`. The error we observe is basically just the first order error of the backward Euler method. Only when the time step becomes very small ($\tau \approx 10^{-6}$) we observe an additional error if `tolY` is not chosen sufficiently small. This is basically the error due to the neglect of some eigenvalues of the solution. In Fig. 3, right we observe how the choice of `tolY` influences the rank of the solution. The full rank given by the solution computed with `lyap` is recovered by the K-PIK procedure with `tolY`= 10^{-12}.

In Fig. 4 we compare the results obtained with the Lie splitting approach described in Algorithm 1 with the ones obtained from the combination of the backward Euler method with the K-PIK procedure. In Fig. 4, left the relative errors for the two methods are shown. We observe that the first-order splitting method is more accurate than the backward Euler method for this example. This is due to the fact that we solve the linear subproblem exactly.

In Fig. 4, right the computing time at $T = 0.1$ is displayed. The solution of the ALE (17) is computed by the K-PIK procedure, where the Cholesky decomposition of the matrix $\widetilde{A}$ is computed once and for all. The computation of the matrix exponential required for the Lie splitting in Algorithm 1 was carried out by Leja interpolation [15] with single precision. We conclude that the splitting proposed here is considerably faster than the backward Euler method combined with the K-PIK approach.

3.4. Numerical results for the simulation of El Niño

The quasiperiodic weather phenomenon El Niño, which is characterized by an unusual warming of the sea surface in the Indo-Pacific ocean, has a huge impact on the climate worldwide and is also responsible for many natural disasters. Different models (both deterministic and stochastic) are used to describe the variation in the sea surface temperature. We consider in the following a stochastic advection equation driven by additive noise

$$\mathrm{d}X(t) = \mathcal{A}X(t) + F(t), \qquad t \in [t_0, T], \tag{19}$$

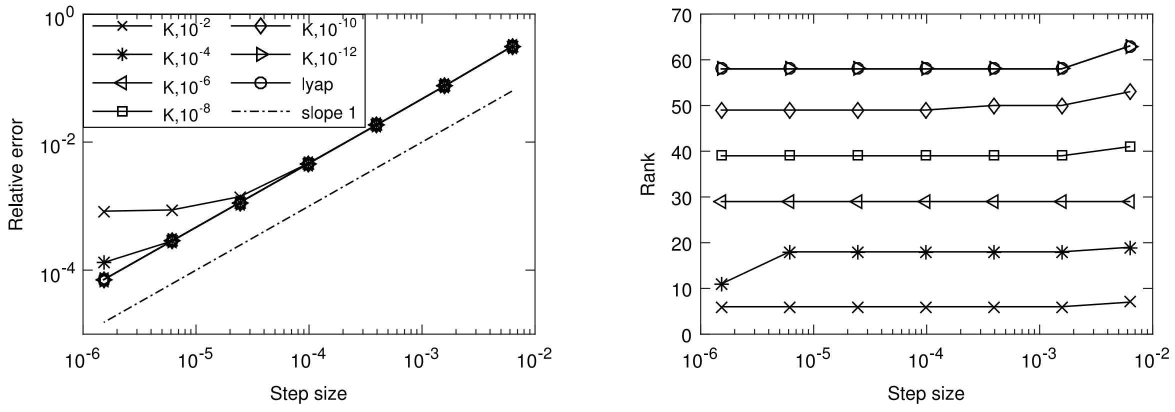

Fig. 3. Comparison between the K-PIK procedure for different values of tolY (K, tolY) and lyap. Left: Errors in Frobenius norm (14) as a function of step size at $T = 0.1$. Right: Rank as a function of the step size at $T = 0.1$.

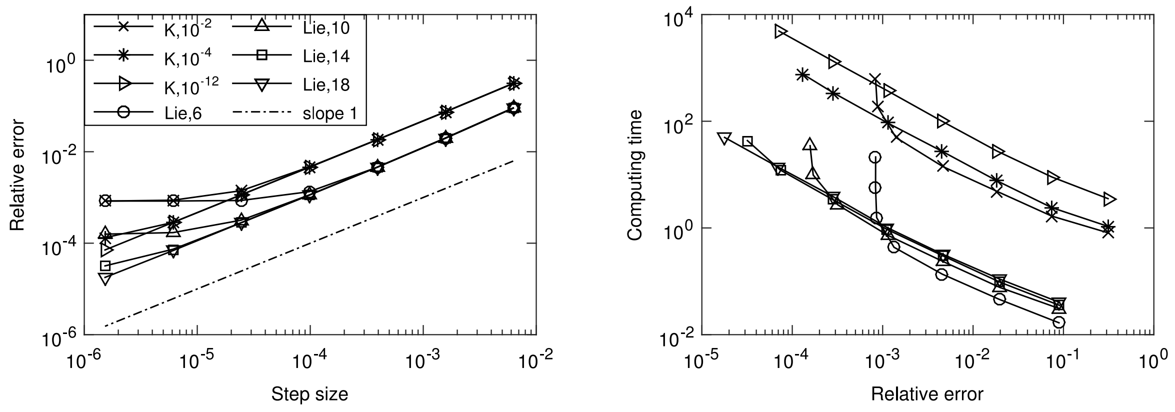

Fig. 4. Comparison between the Lie splitting described in Algorithm 1 for different ranks (Lie, rank) and the backward Euler method combined with the K-PIK procedure for different values of tolY (K, tolY). Left: Errors in the Frobenius norm (14) as a function of step size at $T = 0.1$. Right: Computing time as a function of the error at $T = 0.1$.

where the vector X contains the sea surface temperature (SST) anomalies, the operator $\mathcal{A} = u \cdot \nabla$ describes the advection by the ocean currents u, and F is a random vector which aggregates external forces like wind stress or evaporation [25,26]. Following [26] we construct the SST anomalies, using the dataset OISST [27], whereas for the ocean currents we use the dataset OSCAR [28], both from the National Oceanic and Atmospheric Administration (NOAA). We model the random force F by a Gaussian white noise process with zero mean and covariance operator $\mathcal{Q}$ and compute the first two moments by

$$\mathrm{d}\widehat{X}(t) = \mathcal{A}\widehat{X}(t), \tag{20}$$

$$\mathrm{d}\mathcal{P}(t) = \mathcal{A}\mathcal{P}(t) + \mathcal{P}(t)\mathcal{A}^{\mathsf{T}} + \mathcal{Q}, \tag{21}$$

where $t \in [t_0, T]$ and we denote by $\widehat{X}$ the expectation and by $\mathcal{P}$ the covariance of X [29]. As the solution X is a Gaussian random field, it is completely defined by its second-order statistics.

We discretize the operator $\mathcal{A}$ via centred finite differences and obtain the matrix A. The first subproblem (20) is solved by a method based on Leja interpolation. Discretizing the Indo-Pacific ocean we get 3900 grid points, hence solving the large-scale differential Lyapunov equation (21) with full rank is rather expensive. For such computations, our splitting algorithm has substantial advantages. It requires considerably less computing time and memory. In order to illustrate its accuracy, we make a comparison of the proposed algorithm with a second order scheme. Similarly as in Section 3.3 we apply the backward Euler method to the DLE (21) and solve the resulting ALE by the K-PIK procedure. In order to obtain a second order scheme we then use Richardson extrapolation. Given Z_n which is a low-rank factor of P_n, we make one step with step size τ and get an approximation $\tilde{Z}_{n+1}$ and two steps with step size $\frac{\tau}{2}$ to obtain $\hat{Z}_{n+1}$. The numerical approximation Z_{n+1} is then given by

$$Z_{n+1} = 2\hat{Z}_{n+1} - \tilde{Z}_{n+1}. \tag{22}$$

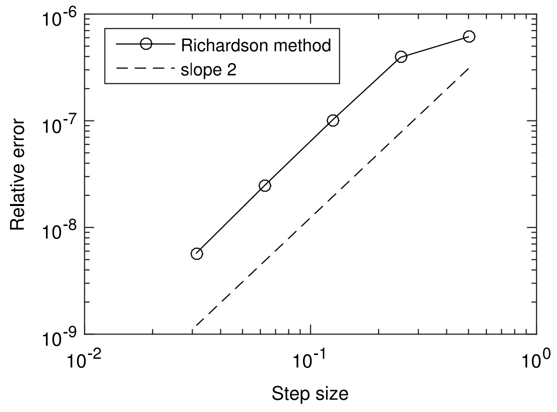

Fig. 5. Error at $T = 5$ of the method (22) for different step sizes and $\mathtt{tolY} = 10^{-12}$ and $\mathtt{tol} = 10^{-10}$.

Table 4
Error of the covariance matrix $\mathcal{P}(T)$ in (21) of size 3900×3900 in Frobenius norm (14) obtained by the low-rank integrator described in Algorithm 1 with step size $\tau = 0.1$ days. The reference solution is computed with step size $\tau = 0.001$ days.

2013			2015		
Weeks	Rank	Error	Weeks	Rank	Error
1	10	3.1117e−05	1	10	5.0599e−05
	100	8.8863e−07		100	1.4408e−06
2	10	3.1229e−05	2	10	5.3995e−05
	100	1.7762e−06		100	2.8798e−06
3	10	3.1687e−05	3	10	5.5525e−05
	100	2.6620e−06		100	4.3158e−06

As discussed in Section 3.3 one has to choose parameters `tol` and `tolY`, which can have an impact in the solution. In the following we use $\mathtt{tol} = 10^{-10}$ and $\mathtt{tolY} = 10^{-12}$. Fig. 5 shows the relative error of this approach. The reference solution is obtained by the same method but with step size 2^{-8}, which is 8 times smaller than the smallest step size used in Fig. 5.

In Table 4 we compare the error in Frobenius norm (14) of the low-rank integrator with respect to the second order method (22) for the years 2013 (no event) and 2015 (a strong event), respectively. In both experiments, we took as starting time June 15. In the table we display the absolute errors of the low-rank approximation after one, two and three weeks for two different approximation ranks. We observe that the error gets smaller for rank 100. For higher ranks, however, the splitting error dominates in this example.

We further make a realization of the stochastic random field. In Figs. 6 and 7 the SST anomalies are given in degree Celsius. We show a section of the Indo-Pacific ocean with Australia being on the lower left part and America on the right part. The black part indicates the land, bright colours indicate higher temperatures and dark ones lower temperatures than usual. In Fig. 6 one can see that the temperature is nearly uniformly distributed, whereas we observe from Fig. 7 the typical face of an El Niño event with the unusual warming of the sea surface.

4. Differential Riccati equations

We now consider the case where $G(t, X) = Q - XPX$ with constant matrices Q and P. The resulting differential equation

$$\begin{aligned} \dot{X}(t) &= AX(t) + X(t)A^{\mathsf{T}} + Q - X(t)PX(t), \\ X(t_0) &= X_0 \end{aligned} \tag{23}$$

is called differential Riccati equation. Again $X(t), A, Q, P \in \mathbb{R}^{d\times d}$, and $t \in [t_0, T]$. The matrices Q and P, and the initial value X_0 are symmetric and positive semidefinite. The global existence and positive semidefiniteness of the solution is guaranteed under these conditions, see [30].

4.1. A low-rank split-step integrator

Following the general idea of the paper, we split (23) into the following two subproblems:

$$\dot{M}(t) = AM(t) + M(t)A^{\mathsf{T}}, \quad M(t_0) = M_0, \tag{24a}$$

$$\dot{N}(t) = Q - N(t)PN(t), \quad N(t_0) = N_0, \tag{24b}$$

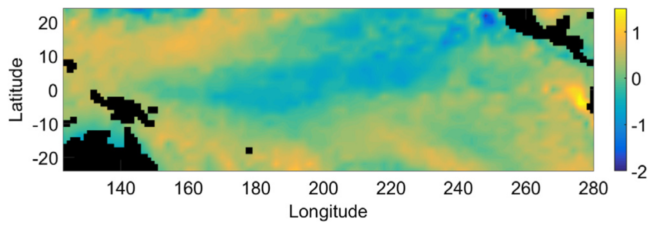

Fig. 6. Simulation of the SST anomalies for December 2013 obtained by the low-rank integrator described in Algorithm 1 with rank 10 and using 3900 grid points.

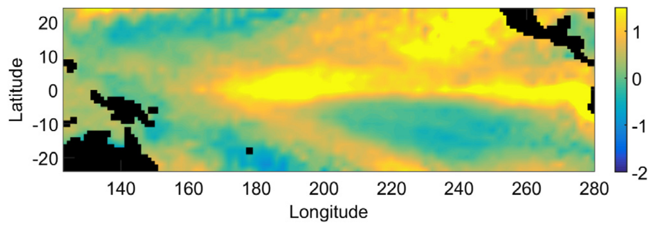

Fig. 7. Simulation of the SST anomalies for December 2015 obtained by the low-rank integrator described in Algorithm 1 with rank 10 and using 3900 grid points.

where $t \in [t_0, T]$. The linear problem (24a) is handled as in Section 3.1. The projector-splitting integrator for the solution of (24b) can be specified taking into account the form of the non-linearity. Moreover, we propose the same kind of modification as for DLEs in order to get a low-rank decomposition of the form USU^{T}. The resulting algorithm for the Lie splitting is given in Algorithm 2.

Algorithm 2 The first-order low-rank split-step integrator for DREs

```
1: As in Algorithm 1.
2: As in Algorithm 1.
3: As in Algorithm 1.
4: As in Algorithm 1.
5: Given $U_n^A$ and $S_n^A$ perform one integration step:
    a. Solve $\dot{K}(t) = QU_n^A - K(t)(U_n^A)^{\mathsf{T}}PK(t)$ with initial value $K(t_n) = U_n^A S_n^A$. Then orthogonalize $K(t_{n+1})$ by a QR decomposition and set $U_{n+1}\widehat{S}_{n+1} = K(t_{n+1})$, where $U_{n+1} \in \mathbb{R}^{d\times r}$ has orthonormal columns and $\widehat{S}_{n+1} \in \mathbb{R}^{r\times r}$.
    b. Solve $\dot{S}(t) = -U_{n+1}^{\mathsf{T}} QU_n^A + S(t)(U_n^A)^{\mathsf{T}} PU_{n+1} S(t)$ with initial value $S(t_n) = \widehat{S}_{n+1}$. Then set $\widetilde{S}_n = S(t_{n+1})$.
    c. Solve $\dot{L}(t) = U_{n+1}^{\mathsf{T}} Q - L(t)PU_{n+1}L(t)$ with initial value $L(t_n) = \widetilde{S}_n (U_n^A)^{\mathsf{T}}$. Then set $S_{n+1} = L(t_{n+1})U_{n+1}$.
6: As in Algorithm 1.
7: As in Algorithm 1.
```

4.2. Numerical results for an optimal control problem

DREs arise, e.g., in optimal control for linear quadratic regulator problems with finite time horizon T for parabolic partial differential equations. Thus we consider the linear control system

$$\dot{x} = Ax + Bu, \quad x(t_0) = x_0,$$

where $A \in \mathbb{R}^{d\times d}$ and $B \in \mathbb{R}^{d\times m}$ are the system matrices, $x \in \mathbb{R}^d$ are the state variables and $u \in \mathbb{R}^m$ is the control. The output $y \in \mathbb{R}^q$ is defined as $y = Cx$, where $C \in \mathbb{R}^{q\times d}$. Both m and q are much smaller than the number d of degrees of freedom. The functional that has to be minimized is

$$\mathcal{J}(u, x) = \frac{1}{2}\int_{t_0}^{T} \Big(x(t)^{\mathsf{T}} C^{\mathsf{T}} QCx(t) + u(t)^{\mathsf{T}} Ru(t)\Big)\, \mathrm{d}t,$$

where $Q \in \mathbb{R}^{q\times q}$ is symmetric and positive semidefinite, and $R \in \mathbb{R}^{m\times m}$ is symmetric and positive definite. The optimal control is given in feedback form by $u_{\mathrm{opt}}(t) = -R^{-1}B^{\mathsf{T}}X(t)x(t)$, where $X(t)$ is the solution of the following DRE

$$\dot{X}(t) = A^{\mathsf{T}}X(t) + X(t)A + C^{\mathsf{T}}QC - X(t)BR^{-1}B^{\mathsf{T}}X(t). \tag{25}$$

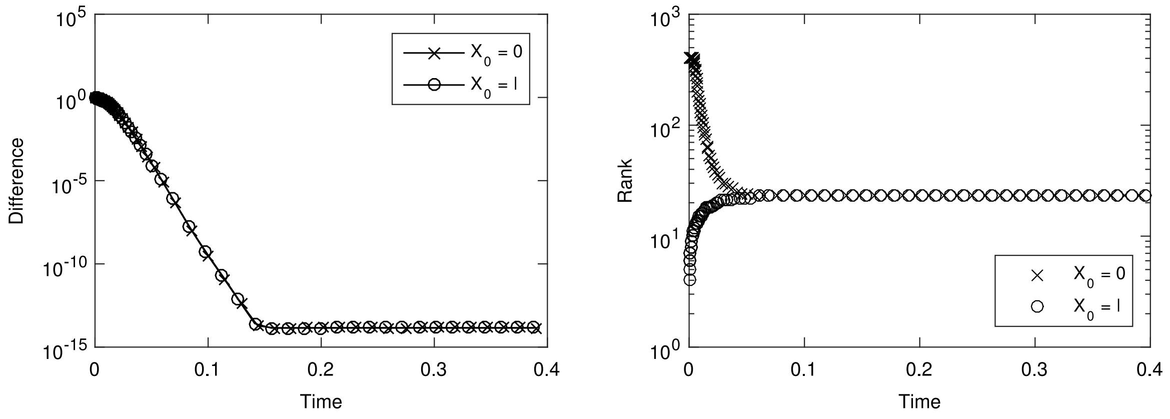

Fig. 8. Results for the DRE (25) for $d = 400$ and initial values $X_0 = 0$ and $X_0 = I$, respectively. Left: Difference in Frobenius norm (14) between (26) and (25) as a function of time. Right: Rank of the reference solution computed with DOPRI5 as a function of time.

Note that the solution of (25) converges for $T \to \infty$ to a steady state. This limit is given as the solution of the algebraic equation

$$0 = A^{\mathsf{T}}X(t) + X(t)A + C^{\mathsf{T}}QC - X(t)BR^{-1}B^{\mathsf{T}}X(t). \tag{26}$$

In order to illustrate the behaviour of Algorithm 2, we consider a test example proposed in [31]. We consider the following diffusion–advection equation

$$\partial_t w = \Delta w - 10x\partial_x w - 100y\partial_y w, \qquad w|_{\partial\Omega} = 0 \tag{27}$$

on $\Omega = (0, 1)^2$ with homogeneous Dirichlet boundary conditions. The matrix A arises from the spatial discretization of (27) using standard centred finite differences, with $\tilde{d}$ uniformly spaced grid points in each dimension. We denote the discretization points in the interior of Ω in x direction with $x_i = i\delta$, $\delta = (\tilde{d}+1)^{-1}$ for $i = 1, \dots, \tilde{d}$. We take $B \in \mathbb{R}^{d\times 1}$ with

$$B_i = \begin{cases} 1 & \text{if} \quad 0.1 < x_i \le 0.3, \\ 0 & \text{else}, \end{cases}$$

and $C \in \mathbb{R}^{1\times d}$ with

$$C_i = \begin{cases} 1 & \text{if} \quad 0.7 < x_i \le 0.9, \\ 0 & \text{else}. \end{cases}$$

Further, we choose $R = I$ and $Q = 100I$.

In Fig. 8 we show the behaviour of a reference solution of (25) computed with DOPRI5. The left figure shows the convergence of the solution of (25) to the solution of (26) for two different initial values $X_0 = 0$ and $X_0 = I$, respectively. As expected this limit is independent of the choice of the initial data. The relative difference between (25) and (26) in Frobenius norm is shown as function of time. Further, in Fig. 8, right we display the rank of the reference solution as a function of time.

In the following numerical example we take the initial value $X_0 = 0$, the final time $T = 0.1$ and the above reference solution computed with high accuracy. We show the results for $d = 400$. The equations in step 5 of Algorithm 2 are quadratic matrix differential equations. We solve them by means of the classical explicit Runge–Kutta method of order 4 with the same step size.

In Fig. 9, left we show the error behaviour of the Lie splitting given in Algorithm 2 . As pointed out above for DLEs, we observe that the error is composed by two different contributions. The choice of a small approximation rank results in stagnation of the error around certain rank-dependent values. On the other hand, if the approximation error becomes small enough, one observes the usual order of convergence one for the outer Lie splitting. In Fig. 10, left we present the corresponding results for Strang splitting. It shows the expected order of convergence two for sufficiently high rank.

In Figs. 9, right and 10, right we show the defects in symmetry (15) and positive semidefiniteness (16) in continuous and dotted lines, respectively. Although our method does not preserve these two features of the solutions, it is remarkable that the defects are always negligible with respect to the overall error of the method.

4.3. Generalized differential Riccati equations

In a stochastic version of the linear quadratic regulator problem, a generalized differential Riccati equation (GDRE) arises

$$\begin{aligned} \dot{X}(t) &= A^{\mathsf{T}}X(t) + X(t)A + Q + C^{\mathsf{T}}X(t)C - X(t)BR^{-1}B^{\mathsf{T}}X(t), \quad t \in [t_0, T], \\ X(0) &= H. \end{aligned} \tag{28}$$

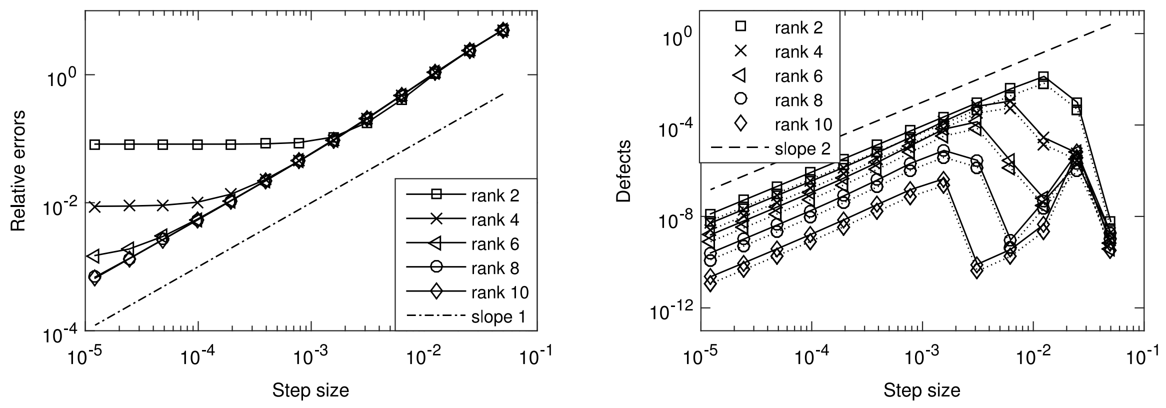

Fig. 9. Results for the DRE (25) for $d = 400$. Left: Errors of Lie splitting described in Algorithm 2 in the Frobenius norm (14) as a function of step size and rank at $T = 0.1$. Right: Defect in symmetry (15) (continuous line) and in positive semidefiniteness (16) (dotted line) for Algorithm 2 for different ranks as a function of step size at $T = 0.1$.

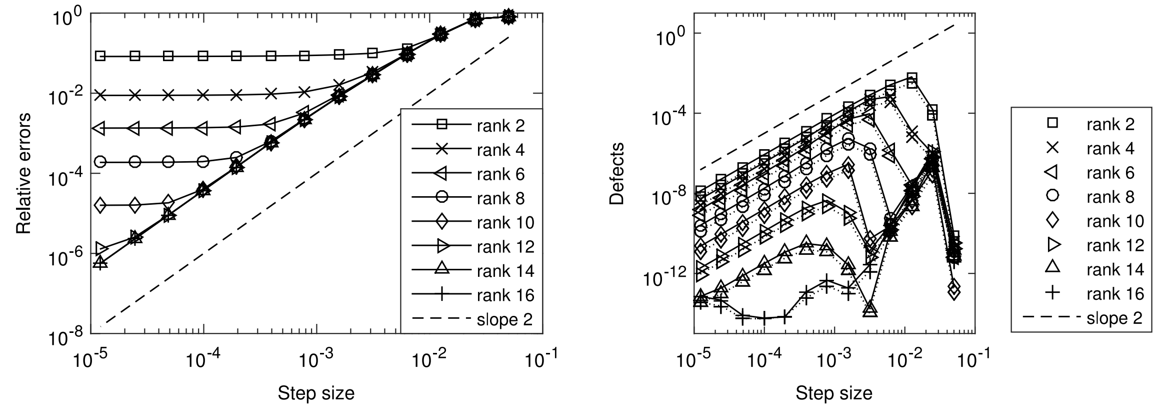

Fig. 10. Results for the DRE (25) for $d = 400$. Left: Errors of Strang splitting in Frobenius norm (14) as a function of step size and rank at $T = 0.1$. Right: Defect in symmetry (15) (continuous line) and in positive semidefiniteness (16) (dotted line) for Strang splitting for different ranks as a function of step size at $T = 0.1$.

We point out that (28) has the same structure as the DRE (23) except for the term $C^{\mathsf{T}}XC$. The matrix C is a weighting term in the noise perturbation, see [32]. Recently, a numerical method for GRDEs of the form (28) was proposed in [33].

The above GDRE shares the structure of (1) with $G = Q + C^{\mathsf{T}}XC - XBR^{-1}B^{\mathsf{T}}X$. It is therefore possible to employ the approach explained in Section 2. Two subproblems of the form (3a) and (3b) arise and they can be solved in low-rank form as presented in this work. The projector-splitting integrator can be adapted to the form of the non-linearity, however, a modification to preserve symmetry and positive semidefiniteness might be required.

5. Conclusions

We proposed a new low-rank integrator for a class of matrix differential equations which includes, among others, differential Lyapunov and differential Riccati equations. A low-rank approximation of the solution is computed in a dynamical way, working only with the low-rank factors of the solution. This approach gives substantial advantages in terms of computing time and memory requirements. Moreover, the integrator can handle stiffness in an efficient way. Splitting methods form the core ingredient of the new method. They make it possible to treat the stiff part of the equation separately from the non-stiff one. Numerical results for differential Lyapunov and differential Riccati equations are discussed, and a simulation of the weather phenomenon El Niño is presented.

Acknowledgement

This work was supported by the Austrian Science Fund (FWF) — project id: P27926.

References

[1] H. Abou-Kandil, G. Freiling, V. Ionescu, G. Jank, Matrix Riccati Equations in Control and Systems Theory, Birkhäuser, Basel, 2003.
[2] A.C. Antoulas, Approximation of Large-Scale Dynamical Systems, SIAM, Philadelphia, 2005.
[3] I.R. Petersen, V.A. Ugrinovskii, A.V. Savkin, Robust Control Design Using H^∞ Methods, Springer, London, 2000.
[4] C. Choi, A.J. Laub, Efficient matrix-valued algorithms for solving stiff Riccati differential equations, IEEE Trans. Automat. Control 35 (1990) 770–776.
[5] L. Dieci, Numerical integration of the differential Riccati equation and some related issues, SIAM J. Numer. Anal. 29 (1992) 781–815.
[6] P. Benner, H. Mena, Rosenbrock methods for solving differential Riccati equations, IEEE Trans. Automat. Control 58 (2013) 2950–2957.
[7] P. Benner, H. Mena, Numerical solution of the infinite-dimensional LQR problem and the associated Riccati differential equations, J. Numer. Math. 26 (2018) 1–20.
[8] T. Stillfjord, Low-rank second-order splitting of large-scale differential Riccati equations, IEEE Trans. Automat. Control 60 (2015) 2791–2796.
[9] O. Koch, C. Lubich, Dynamical low-rank approximation, SIAM J. Matrix Anal. Appl. 29 (2007) 434–454.
[10] E. Kieri, C. Lubich, H. Walach, Discretized dynamical low-rank approximation in the presence of small singular values, SIAM J. Numer. Anal. 54 (2016) 1020–1038.
[11] C. Lubich, I. Oseledets, A projector-splitting integrator for dynamical low-rank approximation, BIT 54 (2014) 171–188.
[12] A. Nonnenmacher, C. Lubich, Dynamical low-rank approximation: applications and numerical experiments, Math. Comput. Simulation 79 (2008) 1346–1357.
[13] A.C. Antoulas, D.C. Sorensen, Y. Zhou, On the decay rate of Hankel singular values and related issues, Systems Control Lett. 46 (2000) 323–342.
[14] J. Sabino, Solution of Large-Scale Lyapunov Equations via the Block Modified Smith Method (Ph.D. thesis), Rice University, Houston, Texas, 2007.
[15] M. Caliari, P. Kandolf, A. Ostermann, S. Rainer, The Leja method revisited: backward error analysis for the matrix exponential, SIAM J. Sci. Comput. 38 (2016) A1639–A1661.
[16] E. Hairer, C. Lubich, G. Wanner, Geometric Numerical Integration. Structure-Preserving Algorithms for Ordinary Differential Equations, second ed., Springer, Berlin, Heidelberg, 2000.
[17] U. Helmke, J.B. Moore, Optimization and Dynamical Systems, Springer, London, 1996.
[18] A.H. Al-Mohy, N.J. Higham, Computing the action of the matrix exponential, with an application to exponential integrators, SIAM J. Sci. Comput. 33 (2011) 488–511.
[19] S. Güttel, Rational Krylov approximation of matrix functions: Numerical methods and optimal pole selection, GAMM-Mitt 36 (2013) 8–31.
[20] Y. Saad, Analysis of some Krylov subspace approximations to the matrix exponential operator, SIAM J. Numer. Anal. 29 (1992) 209–228.
[21] E. Hairer, S.P. Nørsett, G. Wanner, Solving Ordinary Differential Equations I: Nonstiff Problems, second ed., Springer, Berlin, Heidelberg, 1993.
[22] L. Einkemmer, A. Ostermann, Overcoming order reduction in diffusion-reaction splitting. Part 1: Dirichlet boundary conditions, SIAM J. Sci. Comput. 37 (2015) A1577–A1592.
[23] N.J. Higham, Computing a nearest symmetric positive semidefinite matrix, Linear Algebra Appl. 103 (1988) 103–118.
[24] V. Simoncini, A new iterative method for solving large-scale Lyapunov matrix equations, SIAM J. Sci. Comput. 29 (2007) 1268–1288.
[25] J. Case, A simple predictive model for El Niño, SIAM News 42 (2009).
[26] C. Penland, P.D. Sardeshmukh, The optimal growth of tropical sea surface temperature anomalies, J. Clim. 8 (1995) 1999–2024.
[27] National Climatic Data Center, NESDIS, NOAA, U.S. Department of Commerce, NOAA Optimum Interpolation 1/4 Degree Daily Sea Surface Temperature Analysis, Version 2, http://rda.ucar.edu/datasets/ds277.7/. (Accessed 16 February 2016).
[28] ESR, 2009. OSCAR third degree resolution ocean surface currents. Ver. 1. PO.DAAC, CA, USA, http://dx.doi.org/10.5067/OSCAR-03D01. (Accessed 30 May 2016).
[29] H. Mena, L. Pfurtscheller, An efficient SPDE approach for El Niño, arXiv, 2017, https://arxiv.org/abs/1708.04144.
[30] L. Dieci, T. Eirola, Positive definiteness in the numerical solution of Riccati differential equations, Numer. Math. 67 (1994) 303–313.
[31] T. Penzl, LYAPACK Users Guide, Technical Report SFB393/00-33, TU Chemnitz, 2000. https://www.tu-chemnitz.de/sfb393/Files/PDF/sfb00-33.pdf.
[32] J. Yong, X.Y. Zhou, Stochastic Controls. Hamiltonian Systems and HJB Equations, Springer, New York, 1999.
[33] T. Damm, H. Mena, T. Stillfjord, Numerical solution of the finite horizon stochastic linear quadratic control problem, Numer. Linear Algebra Appl. 24 (2017) e2091. http://dx.doi.org/10.1002/nla.2091.