

Alma Mater Studiorum Università di Bologna
Archivio istituzionale della ricerca

Order reduction methods for solving large-scale differential matrix riccati equations

This is the final peer-reviewed author's accepted manuscript (postprint) of the following publication:

Published Version:

Kirsten G., Simoncini V. (2020). Order reduction methods for solving large-scale differential matrix riccati equations. SIAM JOURNAL ON SCIENTIFIC COMPUTING, 42(4), A2182-A2205 [10.1137/19M1264217].

Availability:

This version is available at: <https://hdl.handle.net/11585/784040> since: 2021-03-01

Published:

DOI: <http://doi.org/10.1137/19M1264217>

Terms of use:

Some rights reserved. The terms and conditions for the reuse of this version of the manuscript are specified in the publishing policy. For all terms of use and more information see the publisher's website.

This item was downloaded from IRIS Università di Bologna (<https://cris.unibo.it/>).
When citing, please refer to the published version.

(Article begins on next page)

1 **ORDER REDUCTION METHODS FOR SOLVING LARGE-SCALE**
2 **DIFFERENTIAL MATRIX RICCATI EQUATIONS** *

3 GERHARD KIRSTEN[†] AND VALERIA SIMONCINI[‡]

4 **Abstract.** We consider the numerical solution of large-scale symmetric differential matrix Riccati equations. Under certain hypotheses on the data, reduced order methods have recently arisen as a promising class of solution strategies, by forming low-rank approximations to the sought after solution at selected timesteps. We show that great computational and memory savings are obtained by a reduction process onto rational Krylov subspaces, as opposed to current approaches. By specifically addressing the solution of the reduced differential equation and reliable stopping criteria, we are able to obtain accurate final approximations at low memory and computational requirements. This is obtained by employing a two-phase strategy that separately enhances the accuracy of the algebraic approximation and the time integration. The new method allows us to numerically solve much larger problems than in the current literature. Numerical experiments on benchmark problems illustrate the effectiveness of the procedure with respect to existing solvers.

15 **Key words.** Differential Matrix Riccati, Rational Krylov, Extended Krylov, Linear Quadratic Regulator, Low-rank, BDF

17 **1. Introduction.** We consider the solution of the continuous-time differential matrix Riccati equation (DRE in short) of the form

19 (1.1) $\dot{X}(t) = A^T X(t) + X(t)A - X(t)BB^T X(t) + C^T C, \quad X(0) = X_0,$

20 in the unknown matrix $X(t) \in \mathbb{R}^{n \times n}$, where $X_0 = ZZ^T$ and $t \in [0, t_f]$. Here, $A \in \mathbb{R}^{n \times n}$, $B \in \mathbb{R}^{n \times s}$, $C \in \mathbb{R}^{p \times n}$ and $Z \in \mathbb{R}^{n \times q}$ are time invariant, and $s, p, q \ll n$. The matrix A is assumed to be large, sparse and nonsingular, whereas B , C and Z have full rank. In particular, we consider low-rank DREs, where both matrices $C^T C$ and X_0 have very low rank compared to n . Even though the matrix A is sparse, the solution $X(t)$ is typically dense and impossible to store when n is large. Under the considered hypotheses, numerical evidence seems to indicate that $X(t)$ usually has rapidly decaying singular values, hence a low-rank approximation to $X(t)$ may be considered, see e.g., [48]. For completeness, we also refer the reader to [21, 20] for results on the existence of low-rank solutions for the algebraic Sylvester and Lyapunov equations.

31 The DRE plays a fundamental role in optimal control theory, filter design theory, model reduction problems, as well as in differential games [2, 7, 11, 13, 38]. Equations of the form (1.1) are crucial in the numerical treatment of the linear quadratic regulator (LQR) problem [2, 13, 29]: given the state equation

35 (1.2) $\dot{x}(t) = Ax(t) + Bu(t), \quad y(t) = Cx(t), \quad x(0) = x_0$

36 consider the finite horizon case, where the finite time cost integral has the form

37 (1.3) $J(u) = x(t_f)^T P_f x(t_f) + \int_0^{t_f} (x(t)^T C^T C x(t) + u(t)^T u(t)) dt.$

38 The matrix P_f is assumed to be symmetric and nonnegative definite. Assuming that the pair (A, B) is stabilizable and the pair (C, A) is detectable, the optimal input

* This version dated April 21, 2020

[†]Dipartimento di Matematica, Università di Bologna, Piazza di Porta S. Donato, 5, I-40127 Bologna, Italy, (gerhard.kirsten2@unibo.it)

[‡]Dipartimento di Matematica and AM², Università di Bologna, Piazza di Porta S. Donato, 5, I-40127 Bologna, Italy; and IMATI-CNR, Pavia Italy (valeria.simoncini@unibo.it).

40 $\tilde{u}(t)$, minimizing (1.3), can be determined as $\tilde{u}(t) = -B^T P(t)\tilde{x}(t)$, and the optimal
 41 trajectory is subject to $\dot{\tilde{x}} = (A - BB^T P(t))x(t)$. The matrix $P(t)$ is the solution to
 42 the DRE

$$43 \quad (1.4) \quad \dot{P}(t) = A^T P(t) + P(t)A - P(t)BB^T P(t) + C^T C, \quad P(t_f) = P_f.$$

44 Using a common practice, we can transform (1.4) into the initial value problem (1.1)
 45 via the change of variables $X(t_f - t) = P(t)$.

46 Under certain assumptions, the exact solution of (1.1) can be expressed in integral
 47 form as (see e.g., [28, Theorem 8])

$$48 \quad (1.5) \quad X(t) = e^{tA^T} Z Z^T e^{tA} + \int_0^t e^{(t-s)A^T} (C^T C - X(s)BB^T X(s)) e^{(t-s)A} ds,$$

49 so that when $t \rightarrow \infty$ the DRE reaches a steady state solution satisfying the algebraic
 50 Riccati equation (ARE)

$$51 \quad (1.6) \quad 0 = A^T X_\infty + X_\infty A - X_\infty BB^T X_\infty + C^T C.$$

52 In the framework of differential equations, the DRE is characterized by both fast
 53 and slow varying modes, hence it is classified as a stiff ordinary differential equa-
 54 tion (ODE). The stiffness and the nonlinearity of the DRE are responsible for the
 55 difficulties in its numerical solution even on a small scale ($n < 10^3$). Several stiff
 56 integrators have been investigated, including the matrix generalizations of implicit
 57 ODE solvers [15, 12], linearization methods [14] and more recently matrix versions of
 58 splitting methods [35, 46, 47]. These methods are feasible on a small scale but fail
 59 to be efficient when n is large. In [41], iterative methods are implemented within the
 60 matrix generalization of standard implicit methods allowing for the computation of
 61 an approximate solution to the DRE when $n \gg 10^3$. These algorithms require the
 62 solution of a large *algebraic* Riccati equation at each timestep, which again raises big
 63 concerns as of storage and computational efforts.

64 A promising idea is to rely on a model order reduction strategy typically used
 65 in linear and nonlinear dynamical systems. In this setting, the original system is
 66 replaced with

$$67 \quad (1.7) \quad \dot{\hat{x}}(t) = A_m \hat{x}(t) + B_m u(t), \quad y(t) = C_m \hat{x}(t), \quad \hat{x}(0) = \hat{x}_0$$

68 where A_m, B_m and C_m are projections and restrictions of the original matrices onto
 69 a subspace of small dimension. The differential Riccati equation associated with this
 70 reduced order problem is solved, yielding an optimal corresponding cost function.
 71 This strategy allows for a natural low-rank approximation to the sought after DRE
 72 solution $X(t)$, obtained by interpolating the reduced order solution at selected time
 73 instances. One main feature is that a single space is used for all time snapshots,
 74 so that the approximate solutions can be kept in factored form with few memory
 75 allocations. We refer the reader to, e.g., [4] for a general presentation of algebraic
 76 reduction methods for linear dynamical systems, and to [43] for a detailed discussion
 77 motivating the reduction approach in the context of the algebraic Riccati equation.

78 A key ingredient for the success of the reduction methodology is the choice of the
 79 approximation space onto which the algebraic reduction is performed; [4] presents a
 80 comprehensive description of various space selections in the dynamical system setting.
 81 Following strategies already successfully adopted for the algebraic Riccati equations,
 82 the authors of [27] and [23] have independently used polynomial and extended Krylov

83 subspaces as approximation space, respectively, in the differential setting. A major
 84 characteristic of these spaces is that their dimension can be expanded iteratively, so
 85 that if the determined approximate solution is not sufficiently accurate, the Krylov
 86 space can be enlarged and the process continued. Several questions remain open in
 87 the methods proposed in [27],[23]. On the one hand, it is well known that polynomial
 88 Krylov subspaces require a very large dimension to satisfactorily solve real application
 89 problems, thus destroying the reduction advantages. On the other hand, the multiple
 90 timestepping proposed in the method in [23] only provides an accurate approximation
 91 at $t = t_f$, except when $X_0 = 0$. For $X_0 = ZZ^T \neq 0$ of low rank, memory require-
 92 ments of the extended method grow significantly. These problems can be satisfactorily
 93 solved by using a general *rational* Krylov subspace, which is shown in various appli-
 94 cations to be able to supply good spectral information on the involved matrices with
 95 much smaller dimensions than the polynomial and extended versions. Such gain has
 96 been experimentally reported in the literature in the solution of the *algebraic* Riccati
 97 equation. We show that great computational and memory savings can be obtained
 98 when projecting onto the fully rational Krylov subspace, and that with an appropriate
 99 implementation the extended Krylov subspace may also be competitive with certain
 100 data.

101 A related issue that has somehow been overlooked in the available literature is the
 102 expected final accuracy and thus the stopping criterion. Time dependence of the DRE
 103 makes the reduced problem trickier to handle than in the purely algebraic case; in
 104 particular, two intertwined issues arise: i) The accuracy of the approximate solution
 105 may vary considerably within the time interval $[0, t_f]$; ii) Throughout the reduction
 106 process the reduced ODE cannot be solved with high accuracy and, quite the opposite,
 107 low-order methods should be used to make the overall cost feasible. We analyze these
 108 difficulties in detail, and by exploiting the inherent structure of the reduced order
 109 model, we derive a two-phase strategy that first focuses on the reduction and then on
 110 the integration, in a way that is efficient for memory and CPU time usage, but also
 111 in terms of final expected accuracy.

112 We also discuss several algebraic properties of the approximate solution and its
 113 relation both with the solution $X(t)$ for $t \in [0, t_f]$, and with the steady state solution
 114 X_∞ . These results continue a matrix analysis started in [27], where positivity and
 115 monotonicity properties of the approximate solution obtained by certain reduction
 116 methods are explored.

117 The paper is organized as follows. In [section 2](#) we introduce reduction methods
 118 and discuss the use of Krylov subspace based strategies. Matrix-oriented BDF meth-
 119 ods are recalled for the solution of the projected problem in [section 3](#). In [section 4](#)
 120 we devise a stopping criterion for the order reduction methods and illustrate its key
 121 role in the implementation. [Section 5](#) is devoted to the analysis of matrix properties
 122 of the solution, as well as the reduced model, from a control theory perspective. Sev-
 123 eral numerical experiments are reported in [section 6](#), where the new methods are also
 124 compared with state-of-the-art procedures. Our conclusions are discussed in [section 7](#).
 125 Finally, in [Appendix A](#) and [Appendix B](#) we review some properties of the extended
 126 and rational Krylov subspaces.

127 *Notation and definitions.* Throughout the paper, the matrix I_n will denote the
 128 $n \times n$ identity matrix. In terms of norms, $\|\cdot\|$ refers to any induced matrix norm,
 129 where in particular the Frobenius norm is denoted by $\|\cdot\|_F$. A matrix A is stable
 130 (sometimes also called Hurwitz) if all its *eigenvalues* are contained in the left half
 131 open complex plane. A linear dynamical system, $\dot{x} = Ax$, is called dissipative if the
 132 real matrix A has its *field of values* contained in the left half open complex plane.

133 All reported experiments were performed using MATLAB 9.4 (R2018b) ([33]) on
 134 a MacBook Pro with 8-GB memory and a 2.3-GHz Intel core i5 processor.

135 **2. Order reduction with Krylov-based subspaces.** In this section, we re-
 136 view Krylov-based order reduction methods and show how they are applied to the
 137 DRE. Krylov subspaces that have been explored in the past years have the form

$$\begin{aligned}
 138 \quad \mathcal{K}_m(A, N) &= \text{range} \{ [N, AN, A^2N, \dots, A^{m-1}N] \} && \text{polynomial} \\
 139 \quad \mathcal{EK}_m(A, N) &= \mathcal{K}_m(A, N) + \mathcal{K}_m(A^{-1}, A^{-1}N) && \text{extended} \\
 140 \quad \mathcal{RK}_m(A, N, \mathbf{s}) &= \text{range} \left\{ [N, (A - s_2I)^{-1}N, \dots, \prod_{i=2}^m (A - s_iI)^{-1}N] \right\} && \text{rational.}
 \end{aligned}$$

141 where N is a tall matrix associated with the given problem. In the rational subspace,
 142 $\mathbf{s} = \{s_2, \dots, s_m\}$ is a set of properly chosen real or complex shifts, whose computation
 143 can be performed a priori or dynamically during the generation of the subspace; we
 144 refer the reader to [44, 18] for more complete descriptions.

145 Krylov-based projection methods (in short generically denoted as \mathcal{K}_m) were first
 146 applied to ARE's in [25] (polynomial spaces) and later improved in [24] (extended
 147 space) and [45] (rational spaces). The two rational spaces prove to be far superior
 148 to the polynomial Krylov space in most reduction strategies where they are applied
 149 in the literature, as long as solving linear systems at each iteration is feasible. The
 150 differential Riccati equation has been attacked in [23] with the extended space, and in
 151 [27] with the polynomial space; here we close the gap, as far as Krylov subspaces are
 152 concerned. In addition, we address several implementation issues to make the final
 153 method computationally reliable and, to the best of our knowledge, a great competitor
 154 among the available methods for large-scale DRE problems.

155 While for the algebraic Riccati equation $N = C^T$, in the differential context the
 156 starting matrix for generating these spaces is given by $N = [C^T, Z]$, where $X_0 = ZZ^T$.
 157 Both matrices C and Z play a crucial role in the closed-form DRE solution matrix and
 158 are thus included to generate the projection space. The idea of reduction methods is
 159 to first project the large DRE onto the smaller subspace \mathcal{K}_m , then solve the projected
 160 equation, and finally expand the solution back to the original space.

161 Let the columns of $\mathcal{V}_m \in \mathbb{R}^{n \times d}$ span the considered Krylov subspace. Then the
 162 following Arnoldi-type relation holds,

$$163 \quad (2.1) \quad A^T \mathcal{V}_m = \mathcal{V}_m \mathcal{T}_m^T + \nu_{m+1} \tau_m^T,$$

164 where the actual values of $\nu_{m+1} \in \mathbb{R}^n$ and τ_m^T depend on the chosen subspace. More-
 165 over, setting $\mathcal{V}_{m+1} = [\mathcal{V}_m, \nu_{m+1}]$ we have that $\mathcal{K}_{m+1} = \text{range}(\mathcal{V}_{m+1})$, which shows
 166 that Krylov subspaces are nested, that is $\mathcal{K}_m \subseteq \mathcal{K}_{m+1}$, resulting in a dimension in-
 167 crease after each iteration. Matrix relations leading to (2.1) for the extended and
 168 rational Krylov subspaces are recalled in Appendix A.

169 Assume that \mathcal{V}_m has orthonormal columns. Following similar reduction methods
 170 in the dynamical system contexts, see, e.g., [4], the reduction process consists of first
 171 projecting and restricting the original data onto the approximation space as

$$172 \quad \mathcal{T}_m = \mathcal{V}_m^T A \mathcal{V}_m, \quad B_m = \mathcal{V}_m^T B, \quad Z_m = \mathcal{V}_m^T Z \quad \text{and} \quad C_m = C \mathcal{V}_m.$$

173 Then the following low order differential Riccati equation needs to be solved,

$$\begin{aligned}
 174 \quad (2.2) \quad \dot{Y}_m(t) &= \mathcal{T}_m^T Y_m(t) + Y_m(t) \mathcal{T}_m - Y_m(t) B_m B_m^T Y_m(t) + C_m^T C_m \\
 Y_m(0) &= Z_m Z_m^T,
 \end{aligned}$$

175 for $t \in [0, t_f]$. This low-dimensional DRE admits a unique solution for $t_f < \infty$,
 176 see e.g., [28]. Restrictions on the data to allow for positive, stabilizing solutions are
 177 discussed in more detail in section 5.1. An approximation to the sought after solution
 178 is then written as

$$179 \quad (2.3) \quad X_m(t) = \mathcal{V}_m Y_m(t) \mathcal{V}_m^T \approx X(t), \quad t \in [0, t_f].$$

180 We stress that $X_m(t)$ is never explicitly computed, but always referred to via the
 181 matrix \mathcal{V}_m and the set of matrices $Y_m(t)$ at given time instances. In fact, the matrices
 182 $Y_m(t)$ may also be numerically low rank, so that at the end of the whole process a
 183 further reduction can be performed by truncating the eigendecomposition of $Y_m(t)$
 184 for each t .

185 **REMARK 2.1.** *The approach we have derived is solely based on the order reduction*
 186 *of the dynamical system (1.2). Nonetheless, and with some abuse of notation, the*
 187 *reduced DRE could have been formally obtained by means of a Galerkin condition on*
 188 *the differential equation. For $t \in [0, t_f]$ let*

$$189 \quad \mathcal{R}_m(t) := \dot{X}_m(t) - A^T X_m(t) - X_m(t)A + X_m(t)BB^T X_m(t) - C^T C$$

190 *be the residual matrix for $X_m(t) = \mathcal{V}_m Y_m \mathcal{V}_m^T$. The matrix $Y_m(t)$ is thus determined*
 191 *by imposing that the residual satisfies the following Galerkin condition*

$$192 \quad (2.4) \quad \mathcal{V}_m^T \mathcal{R}_m(t) \mathcal{V}_m = 0, \quad t \in [0, t_f],$$

193 *that is, $\mathcal{R}_m(t) \perp \mathcal{K}_m$ in a matrix sense, so that the residual is forced to belong to a*
 194 *smaller and smaller subspace as \mathcal{K}_m grows. Substituting $X_m(t) = \mathcal{V}_m Y_m(t) \mathcal{V}_m^T$ into*
 195 *the residual matrix, the application of the Galerkin condition results in the projected*
 196 *system*

$$\mathcal{V}_m^T (\mathcal{V}_m \dot{Y}_m(t) \mathcal{V}_m^T - A^T \mathcal{V}_m Y_m(t) \mathcal{V}_m^T - \mathcal{V}_m Y_m(t) \mathcal{V}_m^T A + \mathcal{V}_m Y_m(t) \mathcal{V}_m^T BB^T \mathcal{V}_m Y_m(t) \mathcal{V}_m^T - C^T C) \mathcal{V}_m = 0,$$

197 *which corresponds to (2.2). This is rigorous as long as $\dot{X}_m = \mathcal{V}_m \dot{Y}_m \mathcal{V}_m^T$ holds. \square*

198 It is crucial to realize that, as opposed to some available methods in the literature
 199 (such as, for instance, [41],[47] and the time-invariant algorithms in [30]), the approx-
 200 imation space is independent of the time stepping, that is a single space $\text{range}(\mathcal{V}_m)$
 201 is used for all time steps. This provides enormous memory savings whenever the
 202 approximate solution is required at different time instances in $[0, t_f]$. Theoretical
 203 motivation for keeping the approximation space independent of the time-stepping is
 204 contained in [5], where it is shown that the solution of the DRE lives in an invariant
 205 Krylov-subspace¹.

206 The class of numerical methods we used for solving the reduced DRE is described
 207 in the next section. In the rest of this paper, we specialize the generic derivation
 208 above to the extended and rational Krylov subspaces, which greatly outperformed
 209 polynomial spaces both in terms of CPU time and memory requirements. More infor-
 210 mation on these spaces and their properties are given in Appendix A; in particular,
 211 we discuss the generation of a real rational Krylov basis in the presence of non-real
 212 shifts.

¹ We also refer the reader to the recent manuscript [6], which appeared on-line briefly before the first round of our revision.

213 **3. BDF methods for the DRE.** The numerical solution of the small-scale
 214 DRE is a well-studied topic, see, e.g., [34, 14, 35, 46]. Among the explored methods
 215 are matrix generalizations of the BDF methods [34, 15], which are computationally
 216 appealing only for small problems. Due to the reduction strength of rational Krylov
 217 subspaces, we expect the reduced DRE in (2.2) to be small enough to allow for efficient
 218 use of matrix-based BDF methods, which we are going to summarize next. For
 219 simplicity of exposition, in the rest of this section we omit the subscript in Y_m , and
 220 denote $Y^{(k+1)} = Y(t_{k+1})$. If we define

$$221 \quad (3.1) \quad \mathbf{F}\left(Y^{(k+1)}\right) = \mathcal{T}_m^T Y^{(k+1)} + Y^{(k+1)} \mathcal{T}_m - Y^{(k+1)} B_m B_m^T Y^{(k+1)} + C_m^T C_m,$$

222 the approximation of $Y^{(k+1)}$ is given by the implicit relation

$$223 \quad (3.2) \quad Y^{(k+1)} = \sum_{i=0}^{b-1} \alpha_i Y^{(k-i)} + h\beta \mathbf{F}(Y^{(k+1)}),$$

224 where $h = t_{k+1} - t_k$ is the stepsize and the respective α_i 's and β are the coefficients
 225 of the b -step BDF method for $b \leq 3$ and are given below.

p	β	α_0	α_1	α_2
1	1	1		
2	2/3	4/3	-1/3	
3	6/11	18/11	-9/11	2/11

226 Substituting (3.1) into (3.2) results in the following nonlinear matrix equation

$$228 \quad -Y^{(k+1)} + h\beta \left(\mathcal{T}_m^T Y^{(k+1)} + Y^{(k+1)} \mathcal{T}_m - Y^{(k+1)} B_m B_m^T Y^{(k+1)} + C_m^T C_m \right) + \sum_{i=0}^{b-1} \alpha_i Y^{(k-i)} = 0,$$

229 which can be reformulated as the following continuous-time ARE

$$230 \quad (3.3) \quad \widehat{\mathcal{T}}_m^T Y^{(k+1)} + Y^{(k+1)} \widehat{\mathcal{T}}_m - Y^{(k+1)} \widehat{B}_m \widehat{B}_m^T Y^{(k+1)} + \widehat{Q}_m = 0.$$

231 The coefficient matrices are given by

$$232 \quad \widehat{\mathcal{T}}_m = h\beta \mathcal{T}_m - \frac{1}{2} I_m, \quad \widehat{B}_m = \sqrt{h\beta} B_m, \quad \widehat{Q}_m = h\beta C_m^T C_m + \sum_{i=0}^{b-1} \alpha_i Y^{(k-i)}.$$

233 The Riccati equation (3.3) can be solved using “direct” methods; see, e.g., [9]. In our
 234 experiments we used the MATLAB solver `care` from the control systems toolbox. A
 235 brief sketch of the b -step BDF method is reported in Algorithm 3.1; other approaches
 236 are discussed, e.g., in [34, 30].

237 We conclude the section by depicting the typical convergence behavior of the BDF
 238 methods in our context. We consider an example from [35], where the $n \times n$ matrix
 239 A stems from the spatial finite difference discretization of the following advection-
 240 diffusion equation

$$241 \quad \partial_t w = \Delta w - 10xw_x - 100yw_y, \quad w|_{\partial\Omega} = 0$$

242 on $\Omega = (0, 1)^2$ with homogeneous Dirichlet boundary conditions. The choices of
 243 $B \in \mathbb{R}^{n \times 1}$ and $C \in \mathbb{R}^{1 \times n}$ are given binomially as described in [35]. The initial
 244 condition is taken to be the zero matrix, that is $Z = \mathbf{0}_{n \times 1}$. We compare the obtained

Algorithm 3.1 b -step BDF method – BDF(b, ℓ)

Require: $\mathcal{T}_m \in \mathbb{R}^{d \times d}$, $B_m \in \mathbb{R}^{d \times s}$, $C_m \in \mathbb{R}^{p \times d}$, $Z_m \in \mathbb{R}^{d \times q}$, final time t_f , number of timesteps ℓ , initial approximations $Y^{(0)}, \dots, Y^{(b-1)}$.

- 1: $h = t_f/\ell$, $\hat{\mathcal{T}}_m = h\beta\mathcal{T}_m - \frac{1}{2}I_m$, $\hat{B}_m = \sqrt{h\beta}B_m$
- 2: **for** $k = 0$ **to** ℓ **do**
- 3: $\hat{Q}_m = h\beta C_m^T C_m + \sum_{i=0}^{b-1} \alpha_i Y^{(k-i)}$
- 4: Solve $\hat{\mathcal{T}}_m^T Y^{(k+1)} + Y^{(k+1)} \hat{\mathcal{T}}_m - Y^{(k+1)} \hat{B}_m \hat{B}_m^T Y^{(k+1)} + \hat{Q}_m = 0$
- 5: **end for**
- 6: **return** $Y^{(k)} \approx Y(t_k)$, $t_k = 0, h, \dots, t_f$

245 solution with a “reference” numerical solution $Y_{ref}(t)$ computed by an accurate but
 246 expensive method (the MATLAB function `ode23s` in our experiments), so that n is
 247 kept small, $n = 49$. The convergence behavior for $b = 1, 2, 3$ and ℓ timesteps, with $\ell =$
 248 10, 100, 1000 is displayed in Figure 3.1. The left plot shows the error $\|Y(t) - Y_{ref}(t)\|$
 249 as a function of t , for different values of ℓ . The right plot shows the evolution of
 250 the (1,1) component of the solution throughout the time span for the most accurate
 choice of BDF method, compared with that of the reference solution. These plots

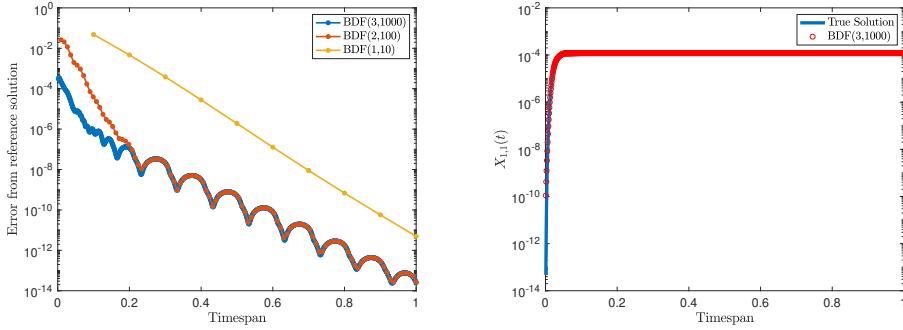


FIG. 3.1. Typical convergence behavior of BDF methods (left) and evolution of the $X_{1,1}$ component of the reference and BDF(3,1000) solution (right).

251 illustrate that we cannot expect an overall high accuracy of the projection method as
 252 long as the reduced differential equation is not solved with sufficiently good accuracy.
 253 The importance of this is discussed in more detail in the following section.
 254

255 **4. Stopping criterion and the complete algorithm.** To complete the re-
 256 duction algorithm of section 2, we need to introduce a stopping criterion. We found
 257 that it is crucial to take into account the accuracy of the numerical method employed
 258 to solve the reduced DRE, as discussed in section 3.

259 To derive our stopping criterion we were inspired by those in [27, 23], however,
 260 we made some important modifications. In both cited references, the authors assume
 261 that the inner problem (2.2) is solved exactly, which is not true in general. We thus
 262 consider that the numerical method solves the reduced problem with residual matrix
 263 $R_m^{(I)}(t) := \dot{Y}_m(t) - \mathbf{F}(Y_m(t))$, so that the final DRE residual can be split into two
 264 components.

265 **PROPOSITION 4.1.** Let $X_m(t) = V_m Y_m(t) V_m^T$ be the Krylov-based approximate
 266 solution after m iterations, where $Y_m(t)$ approximately solves the reduced problem

267 (2.2). With the previous notation, the residual matrix $\mathcal{R}_m(t) = \dot{X}_m(t) - \mathbf{F}(X_m(t))$
 268 satisfies

$$269 \quad (4.1) \quad \|\mathcal{R}_m(t)\|_F^2 = \|R_m^{(I)}(t)\|_F^2 + 2\|R_m^{(O)}(t)\|_F^2,$$

270 where $R_m^{(I)}(t) = \dot{Y}_m(t) - \mathbf{F}(Y_m(t))$ and $R_m^{(O)}(t) = \tau_m^T Y_m(t)$ with τ_m as in (2.1).

271 *Proof.* Substituting (2.3) into the residual $\mathcal{R}_m(t)$ we obtain

$$272 \quad (4.2) \quad \begin{aligned} \mathcal{R}_m(t) &= \mathcal{V}_m \dot{Y}_m(t) \mathcal{V}_m^T - A^T \mathcal{V}_m Y_m(t) \mathcal{V}_m^T - \mathcal{V}_m Y_m(t) \mathcal{V}_m^T A \\ &\quad + \mathcal{V}_m Y_m(t) \mathcal{V}_m^T B B^T \mathcal{V}_m Y_m(t) \mathcal{V}_m^T - C^T C. \end{aligned}$$

273 Since C^T belongs to $\text{range}(\mathcal{V}_m)$, we can write $C^T = \mathcal{V}_m C_m^T$. Using (2.1), we get

$$274 \quad \begin{aligned} \mathcal{R}_m(t) &= \mathcal{V}_m \dot{Y}_m(t) \mathcal{V}_m^T - (\mathcal{V}_m \mathcal{T}_m^T + \nu_{m+1} \tau_m^T) Y_m(t) \mathcal{V}_m^T - \mathcal{V}_m Y_m(t) (\mathcal{T}_m \mathcal{V}_m^T + \tau_m \nu_{m+1}^T) \\ &\quad + \mathcal{V}_m Y_m(t) \mathcal{V}_m^T B B^T \mathcal{V}_m Y_m(t) \mathcal{V}_m^T - \mathcal{V}_m C_m^T C_m \mathcal{V}_m^T. \end{aligned}$$

275 Since $\mathcal{V}_{m+1} = [\mathcal{V}_m, \nu_{m+1}]$, we can write $\mathcal{R}_m(t) = \mathcal{V}_{m+1} \mathcal{J}_m(t) \mathcal{V}_{m+1}^T$, where

$$276 \quad \mathcal{J}_m(t) = \left[\begin{array}{c|c} \dot{Y}_m(t) - \mathcal{T}_m^T Y_m(t) - Y_m(t) \mathcal{T}_m + Y_m(t) B_m B_m^T Y_m(t) - C_m^T C_m & Y_m(t) \tau_m \\ \hline \tau_m^T Y_m(t) & \mathbf{0} \end{array} \right].$$

277 Let $R_m^{(I)}(t)$ be the residual of the numerical ODE inner solver. Then

$$278 \quad \mathcal{J}_m(t) = \left[\begin{array}{c|c} R_m^{(I)}(t) & Y_m(t) \tau_m \\ \hline \tau_m^T Y_m(t) & \mathbf{0} \end{array} \right].$$

279 Since the columns of \mathcal{V}_{m+1} are orthonormal,

$$280 \quad \begin{aligned} \|\mathcal{R}_m(t)\|_F^2 &= \|\mathcal{V}_{m+1} \mathcal{J}_m(t) \mathcal{V}_{m+1}^T\|_F^2 = \|\mathcal{J}_m(t)\|_F^2 \\ &= \text{Tr} \left(R_m^{(I)}(t)^T R_m^{(I)}(t) + 2(Y_m(t) \tau_m)(\tau_m^T Y_m(t)) \right), \end{aligned}$$

281 that is, $\|\mathcal{R}_m(t)\|_F^2 = \|R_m^{(I)}(t)\|_F^2 + 2\|\tau_m^T Y_m(t)\|_F^2$, and the result follows. \square

The expression for $\mathcal{J}_m(t)$ emphasizes that at each iteration m the matrix $Y_m(t)$ is the exact solution of

$$\dot{Y}_m(t) - \mathcal{T}_m^T Y_m(t) - Y_m(t) \mathcal{T}_m + Y_m(t) B_m B_m^T Y_m(t) - C_m C_m^T - R_m^{(I)}(t) = 0.$$

282 Hence, as long as $\|R_m^{(I)}(t)\|_F$ is not very small, the increase of m aims at more and more
 283 accurately approximating a “nearby” differential problem to the truly projected one,
 284 with a term $R_m^{(I)}(t)$ that varies with m . Hence, $X_m(t) = \mathcal{V}_m Y_m \mathcal{V}_m^T$ is an approximation
 285 not to $X(t)$, but to the solution of a differential problem with an additional term whose
 286 projection onto the space is $R_m^{(I)}(t)$.

287 Proposition 4.1 also implies that we cannot expect an overall small residual norm
 288 if either of the two partial residual norms $\|R_m^{(I)}(t)\|_F$, $\|R_m^{(O)}(t)\|_F$ is not small. In par-
 289 ticular, we observe that the two residuals can be made small independently. Therefore
 290 we propose the following practical strategy:

291 (i) Run the algorithm as presented, with a low-order cheap ODE inner solver (i.e.,
 292 BDF(1, ℓ) with ℓ relatively small) and use $R_m^{(O)}(t)$ in the stopping criterion;

293 (ii) Once completed step (i) after \widehat{m} iterations, use the matrices $T_{\widehat{m}}, C_{\widehat{m}}, B_{\widehat{m}}$ and
 294 $Z_{\widehat{m}}$ to refine the ODE inner solution by using a higher-order ODE solver for the
 295 projected system.

296 The final matrix $Y_{\widehat{m}}(t)$ obtained in step (ii) will provide a more accurate solution
 297 matrix than what would have been obtained at the end of step (i). We emphasize
 298 that *any* ODE method for small and medium scale DREs could be used at steps (i)
 299 and (ii). Our choice of BDF(1, ℓ) is due to its good trade-off between accuracy and
 300 computational effort; other approaches could be considered.

301 To complete the description of the stopping criterion, we recall that $R_m^{(O)}(t)$ de-
 302 pends on t , so that we need to estimate the integral over the whole time interval by
 303 means of a quadrature formula, that is

$$304 \quad (4.3) \quad \left\| \int_0^{t_f} R_m^{(O)}(\gamma) d\gamma \right\|_F = \left\| \tau_m^T \int_0^{t_f} Y_m(\gamma) d\gamma \right\|_F \approx \left\| \tau_m^T \sum_{j=1}^{\ell} \frac{t_f}{\ell} Y_m(t_j) \right\|_F =: \rho_m.$$

305 where the interval $[0, t_f]$ has been divided into ℓ intervals with nodes t_j .

306 The overall algorithm² based on the rational Krylov subspace method is reported
 307 in Algorithm 4.1, while the algorithm based on the extended method is postponed
 308 to Appendix B. Several implementation issues of Algorithm 4.1 are also described in
 309 Appendix A, such as the use of a real basis in case of complex shifts s_j in the basis
 310 construction.

Algorithm 4.1 RKSM-DRE

Require: $A \in \mathbb{R}^{n \times n}$, $B \in \mathbb{R}^{n \times s}$, $C \in \mathbb{R}^{p \times n}$, $Z \in \mathbb{R}^{n \times q}$, tol , t_f , ℓ , $\mathbf{s}_0 = \{s_0^{(1)}, s_0^{(2)}\}$

(i) Perform reduced QR: $[C^T, Z] = V_1 \Lambda_1$

Set $\mathcal{V}_1 \equiv V_1$

for $m = 2, 3 \dots$

 Compute the next shift and add it to \mathbf{s}_0

 Compute the next *real* basis block V_m

 Set $\mathcal{V}_m = [\mathcal{V}_{m-1}, V_m]$

 Update $\mathcal{T}_m = \mathcal{V}_m^T A \mathcal{V}_m$ and $B_m = \mathcal{V}_m^T B$, $Z_m = \mathcal{V}_m^T Z$ and $C_m = C \mathcal{V}_m$

 Integrate (2.2) from 0 to t_f using BDF(1, ℓ)

 Compute ρ_m using (4.3) where $\tau_m^T = G_m^T$

if $\rho_m < tol$

go to (ii)

end if

end for

(ii) Refinement: solve (2.2) with a more accurate integrator

Compute $Y_m(t_j) = \widehat{Y}_m(t_j) \widehat{Y}_m(t_j)^T$, $j = 1, \dots, \ell$ using the truncated SVD

return $\mathcal{V}_m \in \mathbb{R}^{n \times m(p+q)}$ and ℓ factors $\widehat{Y}_m(t_j) \in \mathbb{R}^{m(p+q) \times r}$, $j = 1, \dots, \ell$

311 **5. Stability analysis and error bounds.** In this section, we provide a few
 312 results on the spectral and convergence properties of the obtained approximate solu-
 313 tion. We first inspect some properties of the asymptotic matrix solution, which solves

²A Matlab implementation of Algorithm 4.1 can be downloaded from either <https://github.com/Gerhard-Kirsten/Differential-Riccati-RKSM> or <http://www.dm.unibo.it/~simoncin/software>.

314 the algebraic Riccati equation. Then we propose a bound for the error matrix, in an
315 appropriate functional norm.

316 **5.1. Properties of the (steady state) algebraic Riccati equation.** Prop-
317 erties associated with the algebraic Riccati equation – as asymptotic solution to the
318 DRE – are well known in linear quadratic optimal control, see, e.g., [29, 13]. In par-
319 ticular, classical uniqueness and stabilization properties of the solution (see, e.g., [13,
320 Lemma 12.7.2]), can directly be extended to the reduced DRE (2.2).

321 **COROLLARY 5.1.** *Let $(\mathcal{T}_m, B_m, C_m)$ be stabilizable and detectable system. Let*
322 *$Y_m(t)$ be the solution of (2.2) at time t and let $Y_m^\infty = \lim_{t \rightarrow \infty} Y_m(t)$. Then Y_m^∞*
323 *is the unique symmetric nonnegative definite solution and the only stabilizing solution*
324 *to the (reduced) algebraic Riccati equation*

$$325 \quad (5.1) \quad 0 = \mathcal{T}_m^T Y_m^\infty + Y_m^\infty \mathcal{T}_m - Y_m^\infty B_m B_m^T Y_m^\infty + C_m^T C_m.$$

326 *Moreover, if the pair (C_m, \mathcal{T}_m) is observable, Y_m^∞ is strictly positive definite.*

327 We notice that the stabilizability and detectability properties of $(\mathcal{T}_m, B_m, C_m)$
328 are not necessarily implied by those on (A, B, C) . Nevertheless, it is shown in [43]
329 that if there exists a feedback matrix K , such that the linear dynamical system $\dot{x} =$
330 $(A - BK)x$ is dissipative, then the pair (\mathcal{T}_m, B_m) is stabilizable. A similar result can
331 be formulated for the detectability of (C_m, \mathcal{T}_m) , since by duality reasoning, (C_m, \mathcal{T}_m)
332 is detectable if (\mathcal{T}_m^T, C_m^T) is stabilizable. The question regarding the existence of such
333 a feedback matrix, with respect to A and B (A^T and C^T), is addressed in [22].

334 With these results, we can relate the asymptotic solution of the original and
335 projected problems. Let $X_m(t) = \mathcal{V}_m Y_m(t) \mathcal{V}_m^T$ and $X_m^a = \mathcal{V}_m Y_m^\infty \mathcal{V}_m^T$ respectively be
336 approximate solutions to (1.1) and (1.6) by a projection onto $\text{range}(\mathcal{V}_m)$. If there
337 exist matrices K and L such that the systems $\dot{x} = (A - BK)x$ and $\dot{x} = (A^T - C^T L)x$
338 are dissipative, then

$$339 \quad (5.2) \quad \lim_{t \rightarrow \infty} X_m(t) = \mathcal{V}_m \lim_{t \rightarrow \infty} Y_m(t) \mathcal{V}_m^T = \mathcal{V}_m Y_m^\infty \mathcal{V}_m^T = X_m^a,$$

340 that is, X_m^a is the steady state solution of $X_m(t)$ when projected onto the same basis.

341 Under the hypotheses that (A, B, C) is a stabilizable and detectable system, there
342 exists a unique non-negative and stabilizing solution X_∞ to (1.6) (see, e.g., [28, The-
343 orem 5]). In [43] a bound was derived for the error $X_\infty - X_m^a$ in terms of the matrix
344 residual norm. Here we complete the argument by stating that in exact arithmetic
345 and if the whole space can be spanned, the obtained approximate solution equals X_∞ .

346 **PROPOSITION 5.2.** *Suppose (A, B, C) is stabilizable and detectable. Assume it is*
347 *possible to determine m_* such that $\dim(\text{range}(\mathcal{V}_{m_*})) = n$, and let $X_{m_*}^a = \mathcal{V}_{m_*} Y_{m_*}^\infty \mathcal{V}_{m_*}^T$*
348 *be the obtained approximate solution of (1.6) after m_* iterations. Then, $X_{m_*}^a = X_\infty$.*

Proof. Since \mathcal{V}_{m_*} is square and orthogonal the projected ARE is given by

$$0 = \mathcal{V}_{m_*}^T A^T \mathcal{V}_{m_*} Y_{m_*}^\infty + Y_{m_*}^\infty \mathcal{V}_{m_*}^T A \mathcal{V}_{m_*} - Y_{m_*}^\infty \mathcal{V}_{m_*}^T B B^T \mathcal{V}_{m_*} Y_{m_*}^\infty + \mathcal{V}_{m_*}^T C^T C \mathcal{V}_{m_*}.$$

349 From (A, B, C) stabilizable and detectable it follows that $(\mathcal{V}_{m_*}^T A \mathcal{V}_{m_*}, \mathcal{V}_{m_*}^T B, C \mathcal{V}_{m_*})$ is
350 also stabilizable and detectable, so that $Y_{m_*}^\infty \geq 0$ and stabilizing. Multiplying by \mathcal{V}_{m_*}
351 (by $\mathcal{V}_{m_*}^T$) from the left (right), we obtain $0 = A^T X_{m_*}^a + X_{m_*}^a A - X_{m_*}^a B B^T X_{m_*}^a + C^T C$,
352 that is, $X_{m_*}^a \geq 0$ is a solution to the original ARE. Since X_∞ is the unique nonnegative
353 definite solution, it must be $X_{m_*}^a = X_\infty$. \square

354 **5.2. Error bound for the differential Riccati equation.** In this section we
 355 derive a bound for the maximum error obtained by the reduction process, in terms of
 356 the residual

$$357 \quad (5.3) \quad R_m(t) = A^T X_m(t) + X_m(t)A - X_m(t)BB^T X_m(t) + C^T C - \dot{X}_m(t).$$

358 Note that $R_m(t)$ is the residual matrix with respect to the exact solution of the reduced
 359 differential problem, that is, it also includes the discretization error. A similar bound
 360 on the error has been derived for the nonsymmetric DRE in [3], which used matrix
 361 perturbation techniques from [26].

PROPOSITION 5.3. *For $t \in [0, t_f]$ let $\mathcal{E}_m(t) = X(t) - X_m(t)$ and assume that $\mathcal{A}(t) := A - BB^T X(t)$ is stable for all $t \in [0, t_f]$. Denote*

$$\nu := \max_{t \in [0, t_f]} \left\{ \int_0^t \|\Phi_{\mathcal{A}^T}(t, s)\| \|\Phi_{\mathcal{A}}(t, s)\| ds \right\},$$

362 where $\Phi_{\mathcal{A}}$ is the state-transition matrix satisfying $\frac{\partial \Phi_{\mathcal{A}}(t, s)}{\partial t} = \mathcal{A}(t)\Phi_{\mathcal{A}}(t, s)$, $\Phi_{\mathcal{A}}(s, s) = I$.
 If $4\nu^2 \|B\|^2 \|R_m\|_{\infty_t} < 1$, then

$$\|\mathcal{E}_m\|_{\infty_t} \leq 2\nu \|R_m\|_{\infty_t},$$

363 where $\|L\|_{\infty_t} = \max_{t \in [0, t_f]} \|L(t)\|$ for any continuous matrix function $L(t)$.

Proof. By subtracting (5.3) from (1.1) and manipulating terms we obtain

$$\dot{\mathcal{E}}_m(t) = (A - BB^T X(t))^T \mathcal{E}_m(t) + \mathcal{E}_m(t)(A - BB^T X(t)) + \mathcal{E}_m(t)BB^T \mathcal{E}_m(t) + R_m(t),$$

with $\mathcal{E}_m(0) = 0$. Therefore, by the variation of constants formula (see, e.g., [28])

$$\mathcal{E}_m(t) = \int_0^t \Phi_{\mathcal{A}^T}(t, s) (R_m(s) + \mathcal{E}_m(s)BB^T \mathcal{E}_m(s)) \Phi_{\mathcal{A}}(t, s) ds.$$

Taking norms yields

$$\|\mathcal{E}_m(t)\|_{\infty_t} \leq \max_{t \in [0, t_f]} \int_0^t \|\Phi_{\mathcal{A}^T}(t, s)\| \|\Phi_{\mathcal{A}}(t, s)\| (\|R_m(s)\| + \|\mathcal{E}_m(s)\|^2 \|B\|^2) ds,$$

so that $\|\mathcal{E}_m(t)\|_{\infty_t} \leq \nu (\|R_m(t)\|_{\infty_t} + \|\mathcal{E}_m(t)\|_{\infty_t}^2 \|B\|^2)$. Solving this quadratic inequality yields

$$\|\mathcal{E}_m(t)\|_{\infty_t} \leq \frac{1 - \sqrt{1 - 4\nu^2 \|B\|^2 \|R_m\|_{\infty_t}}}{2\nu \|B\|^2}.$$

364 The result follows from multiplying and dividing by $(1 + \sqrt{1 - 4\nu^2 \|B\|^2 \|R_m\|_{\infty_t}})$ and
 365 noticing that at the denominator this quantity can be bounded from below by 1. \square

366 We conclude with a remark on the intuitive fact that if the approximation space
 367 spans the whole space, the obtained solution by projection necessarily coincides with
 368 the sought after solution of the DRE.

REMARK 5.4. *If it is possible to determine m_* such that $\dim(\mathcal{V}_{m_*}) = n$, then the approximate solution $X_{m_*}(t)$ coincides with $X(t)$ for all $t \geq 0$. Indeed, let us write $X_{m_*}(t) = \mathcal{V}_{m_*} Y_{m_*}(t) \mathcal{V}_{m_*}^T$, where \mathcal{V}_{m_*} is square and orthogonal. The reduced DRE is given by*

$$\dot{Y}_{m_*} = \mathcal{V}_{m_*}^T A^T \mathcal{V}_{m_*} Y_{m_*} + Y_{m_*} \mathcal{V}_{m_*}^T A \mathcal{V}_{m_*} - Y_{m_*} \mathcal{V}_{m_*}^T BB^T \mathcal{V}_{m_*} Y_{m_*} + \mathcal{V}_{m_*}^T C^T C \mathcal{V}_{m_*}$$

369 with $Y_{m_*} = Y_{m_*}(t)$. Multiplying by \mathcal{V}_{m_*} (by $\mathcal{V}_{m_*}^T$) from the left (right), we obtain

$$370 \quad \dot{X}_{m_*}(t) = A^T X_{m_*}(t) + X_{m_*}(t)A - X_{m_*}(t)BB^T X_{m_*}(t) + C^T C,$$

371 hence, $X_{m_*}(t) \geq 0$ is a solution of (1.1). Since $X(t)$ is the unique nonnegative definite
372 solution of (1.1) for any $X_0 \geq 0$ (see, e.g., [28]), then $X_{m_*}(t) = X(t)$ for $t \geq 0$. \square

373 **6. Numerical experiments.** In this section we report on our numerical ex-
374 perience with the developed techniques. We consider two artificial symmetric and
375 nonsymmetric model problems, as well as three (of which two are nonsymmetric)
376 standard benchmark problems. Information about the considered data is contained
377 in Table 1. For the first two datasets displayed in Table 1, the matrix A stems from
378 the finite difference discretization with homogenous Dirichlet boundary conditions on
379 the unit square and unit cube, respectively. The first matrix (SYM2D) comes from
380 the finite difference discretization of the two-dimensional Laplace operator in the unit
381 square with homogeneous boundary conditions, while the second matrix (NSYM3D)
382 stems from the finite difference discretization of the three-dimensional differential
383 operator

$$384 \quad \mathcal{L}(u) = e^{xy}(u_x)_x + e^{xy}(u_y)_y + (u_z)_z + (1+x)e^{-x}u_x + y^2u_y + 10(x+y)u_z,$$

in the unit cube, with homogeneous boundary conditions. For both datasets, the
matrices B, C and Z are selected randomly with normally distributed entries. The
realizations of the random matrices are fixed for both examples using the MATLAB
command `rng`: for B, C and Z we use `rng(7)`, `rng(2)` and `rng(3)`, respectively. The
following two datasets (CHIP and FLOW) are taken from [1], and all coefficient matrices
(\hat{A} , \hat{B} , \hat{C} and \hat{E}) are contained in the datasets, which stem from the dynamical system

$$\hat{E}\dot{\hat{x}} = \hat{A}\hat{x} + \hat{B}u, \quad \hat{y} = \hat{C}\hat{x}.$$

385 Since \hat{E} is diagonal and nonsingular, it is incorporated as $A = \hat{E}^{-\frac{1}{2}}\hat{A}\hat{E}^{-\frac{1}{2}}$, while
386 \hat{B} and \hat{C} are updated accordingly to form B and C .

The final considered dataset (RAIL) stems from a semi-discretized heat transfer
problem for optimal cooling of steel profiles³ [8]. We consider the largest of the
four available discretizations (file `rail.79841.c60` containing $\hat{A}, \hat{B}, \hat{C}$ and \hat{E}) with
 $n = 79841$. The symmetric and positive definite mass matrix \hat{E} has a sparsity pattern
very similar to \hat{A} . Both matrices are therefore reordered by the same approximate
minimum degree (RKSM-DRE) or reverse Cuthill-McKee (EKSM-DRE) permutation to
limit fill-in. The state-space transformation is done using the Cholesky factorization
of \hat{E} . More precisely, let $\hat{E} = \hat{E}_L\hat{E}_L^T$ with \hat{E}_L lower triangular, and consider the
transformed state $x = \hat{E}_L^T\hat{x}$. Then

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

387 with $A = \hat{E}_L^{-1}\hat{A}\hat{E}_L^{-T}$, $B = \hat{E}_L^{-1}\hat{B}$ and $C = \hat{C}\hat{E}_L^{-T}$. These matrices are *never* explicitly
388 formed, rather they are commonly applied implicitly by solves with the factor \hat{E}_L at
389 each iteration; see, e.g., [18, 42].

390 The initial low-rank factors are selected as the zero vector for FLOW, $Z = \sin g$
391 for CHIP and $Z = \cos g$ for RAIL, where $g \in \mathbb{R}^{n \times 1}$ is a vector with entries in $[0, 2\pi]$.
392 Other sufficiently general choices were tried during our numerical investigation how-
393 ever results did not significantly differ from the ones we report.

³Data available at http://modelreduction.org/index.php/Steel_Profile

TABLE 1
Relevant information concerning the experimental data

Name	n	$p/s/q$	$\ A\ _F$	$\ B\ _F$	$\ C\ _F$	$\ Z\ _F$	$\ E\ _F$
SYM2D	640000	5/1/1	$3.6 \cdot 10^3$	$8.0 \cdot 10^2$	$1.8 \cdot 10^3$	$8.0 \cdot 10^2$	$8 \cdot 10^2$
NSYM3D	64000	6/1/3	$2.0 \cdot 10^3$	$2.5 \cdot 10^2$	$6.2 \cdot 10^2$	$2.8 \cdot 10^2$	$2.5 \cdot 10^2$
Name	n	$p/s/q$	$\ \widehat{A}\ _F$	$\ \widehat{B}\ _F$	$\ \widehat{C}\ _F$	$\ Z\ _F$	$\ \widehat{E}\ _F$
CHIP	20082	5/1/1	$2.2 \cdot 10^6$	$1.7 \cdot 10^2$	$3.3 \cdot 10^4$	$1.0 \cdot 10^2$	$2 \cdot 10^{-4}$
FLOW	9669	5/1/1	$4.5 \cdot 10^6$	$2.0 \cdot 10^4$	$1.2 \cdot 10^3$	–	$6.8 \cdot 10^0$
RAIL	79841	7/6/1	$7 \cdot 10^{-3}$	$1 \cdot 10^{-7}$	$6.2 \cdot 10^0$	$1.9 \cdot 10^2$	$8 \cdot 10^{-4}$

394 *Performance of the projection methods.* We first investigate the convergence be-
 395 havior of the outer solver. The quantity we monitor in our stopping criterion is the
 396 backward error in an integral norm given by

397 (6.1)
$$\frac{\rho_m}{t_f \|C\|_F^2 + 2\xi_m + \psi_m},$$

with ρ_m as in (4.3) and

$$\xi_m = \left\| A^T \mathcal{V}_m \int_0^{t_f} Y_m(\gamma) d\gamma \right\|_F \quad \text{and} \quad \psi_m = \left\| \int_0^{t_f} Y_m(\gamma) \mathcal{V}_m^T B B^T \mathcal{V}_m Y_m(\gamma) d\gamma \right\|_F.$$

398 The integrals are approximated by a quadrature formula in a similar fashion to (4.3),
 399 and we note that ξ_m can be cheaply computed by using the Arnoldi-type relation.

400 For all datasets, the stopping tolerance was chosen as 10^{-7} . For the first four
 401 datasets, $t_f = 1$ and BDF(1,10) is used as inner solver. For RAIL, $t_f = 4500$ (see e.g., [8]
 402 for further details about the setting) and BDF(1,45) is used as inner solver. Figures 6.1
 403 to 6.5 display the convergence of the rational Krylov subspace method (Algorithm 4.1,
 404 RKSM-DRE) and of the extended Krylov subspace method (Algorithm B.1, EKSM-DRE).
 405 The left plots report the history of the backward error as the approximation space
 406 dimension increases, while the right plots display the same history versus the total
 407 computational time (in seconds) as the iterations proceed. We notice that the cost of
 the refinement step is not taken into account in these first tests.

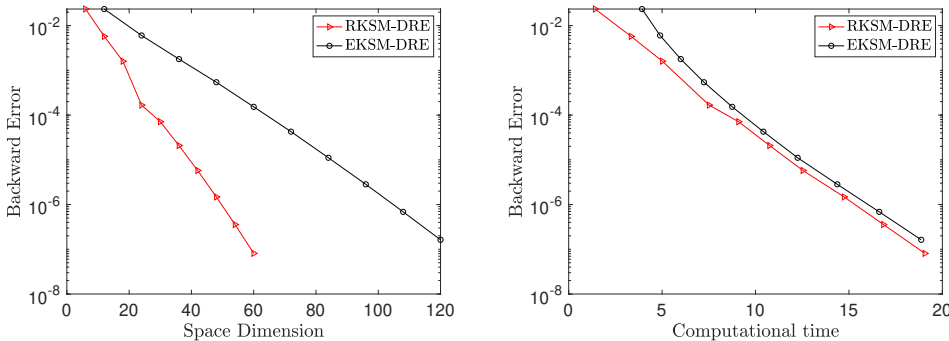


FIG. 6.1. SYM2D: *Convergence history for EKSM-DRE and RKSM-DRE. Left: backward error versus space dimension. Right: backward error versus computational time.*

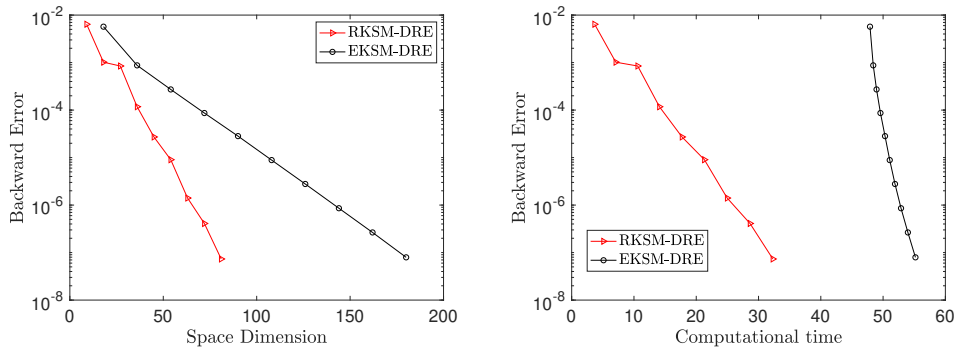


FIG. 6.2. NSYM3D: Convergence history for EKSM-DRE and RKSM-DRE. Left: backward error versus space dimension. Right: backward error versus computational time.

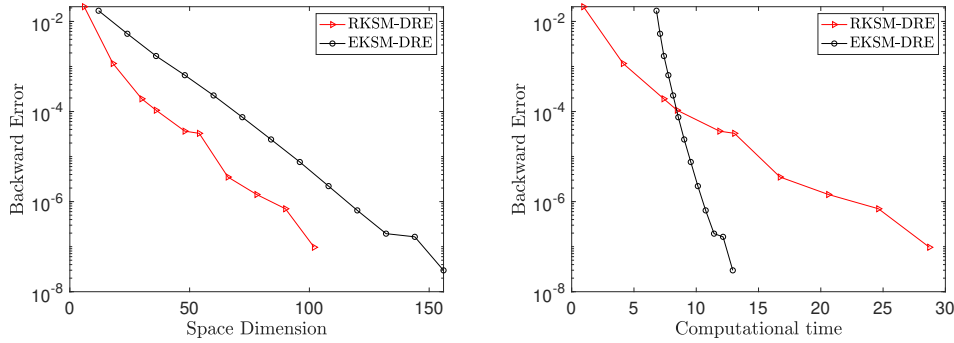


FIG. 6.3. CHIP: Convergence history for EKSM-DRE and RKSM-DRE. Left: backward error versus space dimension. Right: backward error versus computational time.

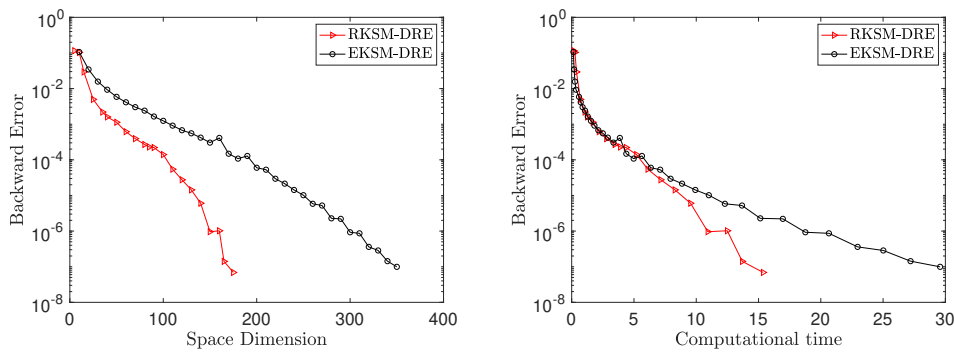


FIG. 6.4. FLOW: Convergence history for EKSM-DRE and RKSM-DRE. Left: backward error versus space dimension. Right: backward error versus computational time.

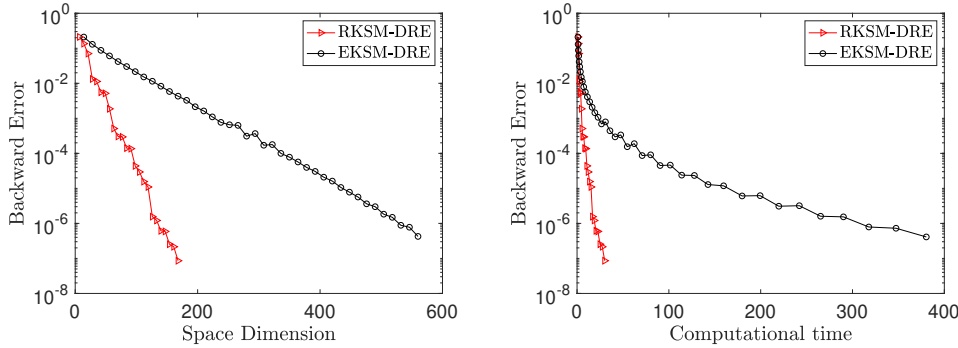


FIG. 6.5. RAIL: Convergence history for EKSM-DRE and RKSM-DRE. Left: backward error versus space dimension. Right: backward error versus computational time.

409 For the dataset SYM2D, the large algebraic linear system in RKSM-DRE was iter-
 410 atively solved by implementing a block conjugate gradient algorithm, with an inner
 411 tolerance of 10^{-10} , preconditioned with an incomplete Cholesky factorization with
 412 drop tolerance 10^{-4} . For all other datasets, the MATLAB built-in backslash operator
 413 was used. For EKSM-DRE the coefficient matrix A used to generate the Krylov space
 414 remains constant, hence a sparse reordered Cholesky (for SYM2D and RAIL) or LU
 415 (for all other datasets) factorization was performed once and for all at the start of
 416 the algorithm. Therefore, only sparse triangular solves are required at each iteration.
 417 Clearly, the cost of the initial factorization depends on the size and density of the
 418 coefficient matrix. These two cost stages are particularly noticeable in the right plots
 419 of Figure 6.2 and Figure 6.3, where the EKSM-DRE curve starts towards the right of
 420 the plot, while the rest of the computation throughout the iterations is significantly
 421 faster.

422 In the implementation of RKSM-DRE it is possible to decide a priori whether to use
 423 only real or generically complex shifts. Our experiments showed that complex shifts
 424 were unnecessary for SYM2D and NSYM3D and, in fact, slowed down convergence when
 425 used. On the other hand, the use of general complex shifts proved to be crucial for
 426 the efficient convergence of RKSM-DRE for CHIP and FLOW. For the symmetric data
 427 in RAIL no complex shifts were used. We mention in passing that both algorithms
 428 are implemented so that the inner solves of (2.2) and the residual computations are
 429 performed at each iteration; for more demanding data we would advise a user to
 430 perform these computations only periodically to save on computational time.

431 Comparing performance, we observe that the two algorithms have alternating
 432 leadership in terms of computational time, but that RKSM-DRE almost consistently
 433 requires half the space dimension of EKSM-DRE. This is expected as the space di-
 434 mension of EKSM-DRE increases with twice the number of columns per iteration, in
 435 comparison to RKSM-DRE. This observation is crucial at the refinement step, where
 436 it could be considerably more expensive to accurately integrate a DRE of dimension
 437 $2m(p + q)$ in comparison to a DRE with approximately half the dimension.

438 To have a clearer picture of how the various steps influence the performance of
 439 the methods, Table 2 depicts the overall computational time for the system solves, the
 440 orthogonalization steps and the integration of the reduced systems for each algorithm.
 441 For EKSM-DRE the CPU time required for the Cholesky and LU factorizations are

442 included in the solving time, but indicated in brackets as well. It is particularly
 443 interesting to notice the small percentage of time required by RKSM-DRE in comparison
 444 to EKSM-DRE for integrating the reduced system, confirming the comment made in
 445 the previous paragraph.

TABLE 2

A breakdown of the computational time for the considered methods for the first two datasets.

Data	Method	System solves (s)	Orthogonalisation steps (s)	Integration steps (s)
SYM2D	RKSM-DRE	6.1	6.9	0.4
	EKSM-DRE	8.6 (2.7)	12.1	1.3
NSYM3D	RKSM-DRE	38.3	0.9	0.8
	EKSM-DRE	48.6 (43.5)	1.6	4.0

446

447 *Comparisons with other BDF based methods.* We compare the two projection
 448 methods RKSM-DRE and EKSM-DRE with low-rank methods that have been developed
 449 following different strategies. The package M.E.S.S. [41], for instance, can solve Lyapunov and Riccati equations, and perform model reduction of systems in state space
 450 and structured differential algebraic form, with time-variant and time-invariant data.
 451 For our purposes, the solvers in M.E.S.S. first discretize the time interval, and then
 452 solve the algebraic Riccati equation resulting from the ODE solver at each time step.
 453 Therefore, the approximation strategy employed at each time iteration to solve the
 454 algebraic problem is completely independent, and the obtained low-rank numerical
 455 solution needs to be stored separately. More precisely, if ℓ timesteps are performed,
 456 the procedure requires solving at least ℓ AREs of large dimensions, delivering the
 457 corresponding low-rank approximate solutions. Moreover, the rank of the constant
 458 term in the ARE increases with the time step, due to the way the ODE solver is
 459 structured, further increasing the complexity of the ARE numerical treatment. In
 460 our experiments with M.E.S.S. we only requested the approximate solution at the
 461 final stage. If the whole approximate solution matrix is requested at different time
 462 instances, the memory requirements will grow linearly with that. The overall strategy
 463 appears to be memory and computational time consuming, therefore we considered
 464 datasets of reduced size for our comparisons, as displayed in Table 3. The considered
 465 timespans were left unchanged.

TABLE 3

Data information for comparisons between projection-based methods and M.E.S.S.

Name	n	$p/s/q$	$\ A\ _F$	$\ B\ _F$	$\ C\ _F$	$\ Z\ _F$	$\ E\ _F$
SYM2D	40000	5/1/1	$1.3 \cdot 10^3$	$3.0 \cdot 10^2$	$6.7 \cdot 10^2$	$3.0 \cdot 10^2$	$2 \cdot 10^2$
NSYM3D	8000	6/1/3	$6.1 \cdot 10^2$	$7.7 \cdot 10^1$	$1.9 \cdot 10^2$	$8.3 \cdot 10^1$	$2.8 \cdot 10^2$
Name	n	$p/s/q$	$\ \widehat{A}\ _F$	$\ \widehat{B}\ _F$	$\ \widehat{C}\ _F$	$\ Z\ _F$	$\ \widehat{E}\ _F$
FLOW	9669	5/1/1	$4.5 \cdot 10^6$	$2.0 \cdot 10^4$	$1.2 \cdot 10^3$	0	$6.8 \cdot 10^0$
RAIL	20209	7/6/1	$4 \cdot 10^{-3}$	$2.1 \cdot 10^{-7}$	$6.2 \cdot 10^0$	$1.9 \cdot 10^2$	$2 \cdot 10^{-4}$

467 Our experimental results are displayed in Tables 4 to 7; we remark that now also
 468 the refinement cost is taken into account in the projection methods. In all tables, the
 469 code BDF(b, ℓ) refers to the BDF method implemented in the refinement procedure of
 470 the reduction methods and in the time discretization procedure of M.E.S.S.

TABLE 4

SYM2D: *Storage and computational time comparison of RKSM-DRE, EKSM-DRE and M.E.S.S.. Reduction phase performed with BDF(1,10), refinement phase with BDF(2,100). In M.E.S.S. only the approximate solution at the final time is stored, with no solutions at intermediate time instances returned.*

Method	# n -long Vecs	Min/Max rank	Reduction phase(s)	Refine phase(s)	Tot CPU time(s)
RKSM-DRE	54	23/43	1.4	0.15	1.6
EKSM-DRE	120	23/43	1.7	1.9	3.6
M.E.S.S.-BDF(1,10)	988	58/75			319.9
M.E.S.S.-BDF(2,100)	1032	58/86			4005.4

471

472

473

474

475

476

477

478

479

480

481

482

483

484

485

486

487

488

489

The tables show the storage requirements in terms of n -length vectors, the minimum and maximum approximate solution rank (with a truncation tolerance 10^{-8} for the projection methods) within the set of solutions, the CPU time break out of projection and refinement phases for the two projected methods, and finally the total CPU time. The stopping tolerance for all algebraic methods – that is the two projection methods and the Newton–Kleinmann-type method used in M.E.S.S. to solve each ARE – is set to 10^{-7} .

In the M.E.S.S. software the user can either select a stopping tolerance (to be used for all solvers within the Newton–Kleinmann strategy) or a maximum number of iterations. We have experimented with both cases, where the maximum number of iterations was detected (a-posteriori) as the maximum number of iterations required within M.E.S.S. to reach the tolerance of 10^{-7} . It was observed that, in the majority of cases, avoiding the residual computation may, in fact, slow down the computational procedure. This is due to the possibility of performing several unnecessary iterations at some timesteps after the desired accuracy has in fact been reached. We therefore only report the results of the more realistic, reliable case where a stopping tolerance is selected beforehand. Galerkin acceleration is used to boost the performance of Newton–Kleinmann.

TABLE 5

NSYM3D: *Storage and computational time comparison of RKSM-DRE, EKSM-DRE and M.E.S.S.. Reduction phase performed with BDF(1,10), refinement phase with BDF(2,100). In M.E.S.S. only the approximate solution at the final time is stored, with no solutions at intermediate time instances returned.*

Method	# n -long Vecs	Min/Max rank	Reduction phase(s)	Refine phase(s)	Tot CPU time(s)
RKSM-DRE	90	36/66	2.4	2.8	5.2
EKSM-DRE	180	36/66	2.6	5.4	8.0
M.E.S.S.-BDF(1,10)	1116	71/90			431.0
M.E.S.S.-BDF(2,100)	1152	67/94			4965.0

490

491

492

493

494

495

496

All numbers in the tables illustrate the large computational costs of M.E.S.S., as expected by the strategy “first time-discretize, then solve”, whereas both projection methods require just a few seconds of CPU in most cases.

The storage requirements of both reduction methods is independent of the number of timesteps where the solution is required. This is due to the fact that only a few n -long basis vectors need to be generated and stored, while only the reduced problem

497 solution $Y_m(t)$ changes at the timesteps t . The memory requirements of M.E.S.S. are
 498 measured as the dimensions of the low-rank factor returned by the Newton-Kleinmann
 499 procedure, before column compression, at the final timestep. The dimension decreases
 500 significantly with the column compression. In our experiments we only stored the
 501 approximate solution at the last time step, however memory will be correspondingly
 502 higher if the whole approximation matrix is required at more instances (memory will
 503 thus grow linearly with the number of time instances to be monitored).

504 Between the two projection methods, we observe that the extended space yields a
 505 significantly larger basis than the actual approximate solution rank it produces. This
 506 means that the approximate solution belongs to a much smaller space than the one
 507 constructed by EKSM-DRE. This is far less so with RKSM-DRE. The different behavior
 508 confirms what has been already observed for the two methods in the ARE case [45].

TABLE 6

FLOW: Storage and computational time comparison of RKSM-DRE, EKSM-DRE and M.E.S.S.. Reduction phase performed with BDF(1,10), refinement phase with BDF(2,100). In M.E.S.S. only the approximate solution at the final time is stored, with no solutions at intermediate time instances returned.

Method	# n -long Vecs	Min/Max rank	Reduction phase(s)	Refine phase(s)	Tot CPU time(s)
RKSM-DRE	175	95/100	11.8	4.5	16.3
EKSM-DRE	350	95/100	27.4	23.5	50.9
M.E.S.S.-BDF(1,10)	1280	87/106			431.7

509

TABLE 7

RAIL: Storage and computational time comparison of RKSM-DRE, EKSM-DRE and M.E.S.S.. Reduction phase performed with BDF(1,10), refinement phase with BDF(2,100). In M.E.S.S. only the approximate solution at the final time is stored, with no solutions at intermediate time instances returned.

Method	# n -long Vecs	Min/Max rank	Reduction phase(s)	Refine phase(s)	Tot CPU time(s)
RKSM-DRE	168	153/160	6.4	3.3	9.7
EKSM-DRE	462	153/160	39.2	5.7	44.9
M.E.S.S.-BDF(1,10)	6345	151/158			705.3
M.E.S.S.-BDF(2,100)	4023	124/158			3396.5

510

511 *Comparisons with splitting methods.* We next compare RKSM-DRE with the fourth
 512 order additive splitting method (SPLIT-ADD4(ℓ)) developed in [47]. The method is
 513 based on splitting the DRE into the linear and non-linear subproblems, for which
 514 respective closed form solutions exist and are explicitly approximated. The numerical
 515 solutions to the subproblems are then recombined to approximate the solution
 516 to the full problem, by means of an additive splitting scheme. The main computa-
 517 tional effort is due to the repeated evaluation of matrix exponentials, which has been
 518 resolved by using a Krylov-based matrix exponential approximation. Similar to the
 519 issue discussed with M.E.S.S. in the previous section, the ℓ (factored) solution ma-
 520 trices are independently calculated at each timestep, leading to significant memory
 521 requirements.

522 To ensure that we are comparing methods with similar approximation accuracies,
 523 we generate reference solutions $X_{ref}(t_j)$ for the selected time instances t_j . This is
 524 done by using RKSM-DRE with a stopping tolerance of 10^{-10} , plus a refinement process
 525 with BDF(4, 10^4) from [41]. To allow for such accurate approximations, we consider
 526 slightly smaller problem dimensions for the first two datasets, and we set $p = s = 1$
 527 and $X_0 = 0$.

The input parameters are tailored so that the approximate solutions from different methods have reliable accuracies. In particular, RKSM-DRE is solved with an outer stopping tolerance of 10^{-6} and with BDF(3,1000) in the refinement process. The number of timesteps utilized in SPLIT-ADD4 is selected as $\ell = 500$. The expected approximation errors relative to the reference solution, measured as

$$\frac{\|X_{approx}(t) - X_{ref}(t)\|_F}{\|X_{ref}(t)\|_F},$$

528 are illustrated in Figure 6.6 (dataset SYM2D in the left plot, dataset NSYM3D in the right plot). The figures indicate that we compare methods having approximation

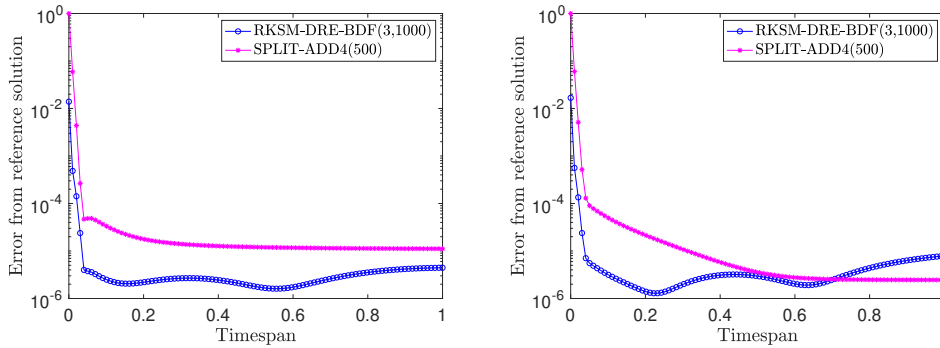


FIG. 6.6. Expected approximation error for RKSM-DRE and SPLIT-ADD4(500). Left: Dataset SYM2D. Right: Dataset NSYM3D.

529 errors of similar order. The performance results are contained in Table 8 for two
 530 different discretizations of SYM2D and NSYM3D.
 531

532 All numbers indicate the competitiveness of RKSM-DRE in terms of storage and com-
 533 putational time. The memory requirements for SPLIT-ADD4 is measured as the di-
 534 mension of the solution factor at the final timestep, before column compression. If
 535 the solution is required at more time instances, then these memory requirements will
 536 increase accordingly.
 537

538 We also mention that we have experimented with the dynamic splitting methods
 539 introduced in [35], however the algorithms proposed by the authors⁴ in [35] appeared
 540 to be better suited for small to medium size problems.

541 *Discussion on the refinement step.* In previous sections, we have stressed that
 542 the two approximation stages of the projection method are independent, and we have
 543 focused on determining an effective approximation space. Here we linger over the

⁴We thank Chiara Piazzola for providing us with her Matlab implementation of the method.

TABLE 8

Storage and computational time comparison of RKSM-DRE and SPLIT-ADD4(500). Reduction phase performed with BDF(1,10), refinement phase with BDF(3,1000). In SPLIT-ADD4 only the approximate solution at the final time is stored.

Data (n)	Method	# n -long Vecs	Min/Max rank	Tot CPU time (s)
SYM2D (10^4)	RKSM-DRE	8	3/6	0.6
	SPLIT-ADD4(500)	28	3/7	34.9
NSYM3D ($8 \cdot 10^3$)	RKSM-DRE	10	4/7	2.2
	SPLIT-ADD4(500)	36	3/9	37.9
SYM2D ($9 \cdot 10^4$)	RKSM-DRE	6	3/4	1.2
	SPLIT-ADD4(500)	28	3/7	330.0
NSYM3D ($2.7 \cdot 10^4$)	RKSM-DRE	10	4/8	10.1
	SPLIT-ADD4(500)	36	3/9	127.8

TABLE 9

SYM2D (of size 10^6): Results with RKSM-DRE, using different refinement strategies. Reduction phase performed with BDF(1,10) and tolerance 10^{-8} .

Refinement Method	# n -long Vecs	Soln. rank	Reduction phase(s)	Refinement phase(s)	Tot CPU time(s)
BDF(2,100)	72	55	37.7	1.1	38.8
BDF(3,1000)	72	55	37.7	9.6	47.3
BDF(4,10000)	72	55	37.7	95.5	133.2
SPLIT-ADD4(500)	72	55	37.7	1.9	39.6
SPLIT-ADD8(500)	72	55	37.7	5.1	42.8
SPLIT-ADAPT8	72	55	37.7	23.1	60.8

544 accuracy of the second stage, the refinement step. Exploiting the far smaller problem
545 size of the reduced problem, it is possible to allow for a much more accurate integration
546 phase than what was done during the iteration of the reduction step. This crucial
547 fact is already illustrated in the time break down of Tables 4 to 6, where especially
548 for RKSM-DRE the refinement phase employs a fraction of the overall computational
549 time, while still allowing for a rather accurate final solution.

550 We next explore in more detail these advantages with RKSM-DRE on SYM2D,
551 where the discretization was further refined to get a coefficient matrix of dimension
552 10^6 . The dimensions of the other corresponding matrices remain as presented in
553 Table 1. We investigate the time taken by DRE solvers with different accuracies to
554 emphasize the advantages and flexibility of the refinement procedure. Table 9 reports
555 the timings for a refinement step performed by three different BDF methods and three
556 splitting methods. The 8th order adaptive splitting method (SPLIT-ADAPT8) also
557 comes from [47] and is performed with a tolerance of 10^{-7} . We emphasize that in
558 the refinement phase we have utilized some of the most accurate integrators available,
559 and nevertheless the high-dimensional ($n = 10^6$) problem is approximated in less than
560 150 seconds for all integrators.

561

562 **7. Conclusions and open problems.** We have devised a rational Krylov sub-
563 space based order reduction method for solving the symmetric differential Riccati
564 equation, providing a low-rank approximate solution matrix at selected time steps. A

565 single projection space is generated for all time instances, and the space is expanded
 566 until the solution is sufficiently accurate. We stress that our approach is very general,
 567 and that it could be applied to subspaces other than Krylov-based ones, as long as the
 568 spaces are nested, so that they keep growing as the iterations proceed. This methodol-
 569 ogy could then be employed for more complex settings, such as parameter dependent
 570 problems, where the involved approximation space may require the inclusion of some
 571 parameter sampling.

572 Like in typical model order reduction strategies, in our methodology time step-
 573 ping is only performed at the reduced level, so that the integration cost is drastically
 574 lower than what one would have by applying the time stepping on the original large
 575 dimensional problem. We have derived a new stopping criterion that takes into ac-
 576 count the different approximation behavior of the algebraic and differential portions
 577 of the problem, together with a refinement procedure that is able to improve the final
 578 approximate solution by using a high-order integrator. These enhancement strategies
 579 have also been applied to the extended Krylov subspace approach. We have ana-
 580 lyzed the asymptotic behavior of the reduced order solution, so as to ensure that the
 581 generated approximation behaves like the sought after time-dependent solution.

582 Although our numerical results are promising, there are still several open issues
 583 associated with the reduced order solution of the DRE. In particular, while stability
 584 and other matrix properties associated with the solutions $X(t)$ have been thoroughly
 585 studied [10, 19, 37, 16], the analysis of corresponding properties for the approximate
 586 solution $X_m(t) = \mathcal{V}_m Y_m(t) \mathcal{V}_m^T$ for $t \in [0, t_f]$ is still a largely open problem. In [27] some
 587 interesting monotonicity properties have been shown when the polynomial Krylov
 588 subspace is used together with particular ODE solvers; a complete analysis for $X_m(t)$
 589 in a more general setting would be desirable.

590 **Acknowledgements.** We thank Khalide Jbilou for helpful explanations on [3],
 591 and Jens Saak for assistance in using the software M.E.S.S. v.1.0.1 [41]. We are
 592 also grateful to Chiara Piazzola for making her code from [35] available and to Tony
 593 Stillfjord for pointing us towards his codes from [47]. We thank Maximilian Behr
 594 for some helpful criticism on a previous version of this manuscript. The first author
 595 acknowledges the insightful comments and kind hospitality of the group of Peter
 596 Benner at the Max Planck Institute in Magdeburg (D), during a short visit in February
 597 2019. Furthermore, we thank two anonymous referees for their careful reading and
 598 insightful comments.

599 This research is supported in part by the Alma Idea grant of the Alma Mater
 600 Studiorum Università di Bologna, and by INdAM-GNCS under the 2018 Project
 601 “Metodi numerici per equazioni lineari, non lineari e matriciali con applicazioni”.
 602 The second author is a member of INdAM-GNCS.

603 **Appendix A. Krylov subspace properties.** In this Appendix we review
 604 some properties of extended and rational Krylov subspaces. As in section 2 we denote
 605 $N = [C^T, Z]$.

606 *Extended Krylov subspace.* The extended Krylov subspace $\mathcal{EK}_m(A^T, N)$ takes the
 607 form discussed in section 2. The orthonormal basis $\mathcal{V}_m \in \mathbb{R}^{n \times 2m(p+q)}$ spanning the
 608 subspace is formed using the extended Arnoldi algorithm [17]. Let

$$609 \quad (A.1) \quad \tilde{\mathcal{T}}_m^T = \mathcal{V}_{m+1}^T A^T \mathcal{V}_m = \begin{bmatrix} \mathcal{T}_m^T \\ t_{m+1,m} E_{2m}^T \end{bmatrix} \in \mathbb{R}^{2(m+1)(p+q) \times 2m(p+q)},$$

610 where $\mathcal{V}_{m+1} = [\mathcal{V}_m \ V_{m+1}] \in \mathbb{R}^{n \times 2(m+1)(p+q)}$ and E_{2m} is the last $2(p+q)$ columns of

611 $I_{2m(p+q)}$. The extended Arnoldi algorithm produces the Arnoldi-type relation

$$612 \quad (\text{A.2}) \quad A^T \mathcal{V}_m = \mathcal{V}_{m+1} \tilde{\mathcal{T}}_m^T = \mathcal{V}_m \mathcal{T}_m^T + V_{m+1} t_{m+1,m} E_{2m}^T.$$

613 *Rational Krylov subspace.* The rational Krylov subspace was originally proposed
614 in the eigenvalue context in [39]. Its use in our context is motivated by [45] and later
615 [43], where its effectiveness in the solution of the algebraic Riccati equation is amply
616 discussed.

617 Assume that A is Hurwitz. Given $\mathbf{s} = \{s_1, s_2, \dots\}$, with $s_j \in \mathbb{C}^+$, the rational
618 Krylov subspace is given by $\mathcal{RK}_m(A, N, \mathbf{s})$ as defined in section 2. The approximation
619 effectiveness of this subspace depends on the choice of shifts \mathbf{s} , and this issue has been
620 investigated in the literature; see, e.g., [36], [18]. The adaptive choice of shifts was
621 tailored to the ARE in [32] by the inclusion of information of the term BB^T during
622 the shift selection; see also [43] for a more detailed discussion⁵. In our numerical
623 experiments we used this last adaptive strategy, where the approximate solution at
624 timestep t_f is used.

625 The algorithm presented in [18] forms a complex basis, when the shifts are not all
626 real. In short, when $s_j \in \mathbb{C}^+$, the original approach would be to use the shift s_j to form
627 the next block V_j and to then let the following shift be given by $s_{j+1} = \bar{s}_j$, where \bar{s}_j
628 denotes the complex conjugate of s_j . This results in both V_j and V_{j+1} being complex.
629 As a consequence, the reduced DRE has complex coefficient matrices, although the
630 final resulting approximations $X_m(t)$ will be real. Standard ODE solvers do not
631 handle complex arithmetic well, hence we implemented an all-real basis using the
632 method introduced in [40], which works as follows. If the shift s_j is complex then the
633 block $W_j = (A - s_j I)^{-1} V_{j-1}$ is also complex, hence we split it into its real and complex
634 parts, that is $W_j = W_j^{(r)} + W_j^{(c)} i$. The block V_j is then formed by orthogonalizing $W_j^{(r)}$
635 with respect to all vectors in the already computed basis, after which V_{j+1} is formed by
636 orthogonalizing $W_j^{(c)}$ with respect to all previous vectors in the computed basis, and
637 in V_j . This determines the same space, since $\text{span}\{W_j, \bar{W}_j\} = \text{span}\{V_j, V_{j+1}\}$. The
638 resulting *real* basis of the rational Krylov subspace is given by $\mathcal{V}_m = [V_1, \dots, V_m] \in$
639 $\mathbb{R}^{n \times m(p+q)}$. We also define the matrices $\mathcal{V}_{m+1} = [\mathcal{V}_m, V_{m+1}] \in \mathbb{R}^{n \times (m+1)(p+q)}$ and the
640 matrix

$$641 \quad (\text{A.3}) \quad \tilde{\mathcal{H}}_m = \begin{bmatrix} \mathcal{H}_m \\ r_{m+1,m} E_m^T \end{bmatrix} \in \mathbb{R}^{(m+1)(p+q) \times m(p+q)},$$

642 where $r_{m+1,m} \in \mathbb{R}^{(p+q) \times (p+q)}$ and E_m holds the last $(p+q)$ columns of $I_{m(p+q)}$. The
643 matrix $\tilde{\mathcal{H}}_m$ contains the orthogonalization coefficients obtained during the rational
644 Arnoldi algorithm.

645 Let $\mathcal{T}_m^T = \mathcal{V}_m^T A^T \mathcal{V}_m \in \mathbb{R}^{m(p+q) \times m(p+q)}$. The rational Krylov basis satisfies the
646 Arnoldi-type relation

$$647 \quad (\text{A.4}) \quad A^T \mathcal{V}_m = \mathcal{V}_m \mathcal{T}_m^T + \hat{V}_{m+1} G_m^T,$$

648 where $G_m^T = \gamma r_{m+1,m} E_m^T \mathcal{H}_m^{-1}$ and the matrix \hat{V}_{m+1} is an orthonormal matrix such
649 that

$$650 \quad (\text{A.5}) \quad \hat{V}_{m+1} \gamma = V_{m+1} s_m - (I_n - \mathcal{V}_m \mathcal{V}_m^T) A^T V_{m+1}$$

⁵The Matlab code of the rational Krylov subspace method for ARE is available at
<http://www.dm.unibo.it/~simoncin/software>

651 is the QR decomposition of the matrix on the right (see [18, 31]). The rational Krylov
 652 procedure requires as an extra input the (usually real) values $s_0^{(1)}, s_0^{(2)}$, which form a
 653 rough approximation of a spectral region used to compute the next shift. The reader
 654 is referred to [18, 43] for implementation details. Further, for the computation of
 655 the term $G_m^T Y_m(t)$ contained in the residual computation of RKSM-DRE, we follow an
 656 accelerated computation technique presented in [18].

657 **Appendix B. Extended Krylov subspace based method.** The extended
 658 Krylov (EKSM-DRE) subspace method for solving (1.1) is presented in Algorithm B.1.

Algorithm B.1 EKSM-DRE

Require: $A \in \mathbb{R}^{n \times n}$, $B \in \mathbb{R}^{n \times s}$, $C \in \mathbb{R}^{p \times n}$, $Z \in \mathbb{R}^{n \times q}$, tol , t_f , ℓ

(i) Perform reduced QR : $([C^T, Z], A^{-1}[C^T, Z]) = V_1 \Lambda_1$

Set $\mathcal{V}_1 \equiv V_1$

for $m = 2, 3 \dots$

 Compute the next basis block V_m

 Set $\mathcal{V}_m = [\mathcal{V}_{m-1}, V_m]$

 Update \mathcal{T}_m as in [42] and $B_m = \mathcal{V}_m^T B$, $Z_m = \mathcal{V}_m^T Z$ and $C_m = C \mathcal{V}_m$

 Integrate (2.2) from 0 to t_f using BDF(1, ℓ)

 Compute ρ_m using (4.3) where $\tau_m^T = t_{m+1, m} E_{2m}^T$

if $\rho_m < tol$

go to (ii)

end if

end for

(ii) Refinement: solve (2.2) with a more accurate integrator

Compute $Y_m(t_j) = \hat{Y}_m(t_j) \hat{Y}_m(t_j)^T$, $j = 1, \dots, \ell$ using the truncated SVD

return $\mathcal{V}_m \in \mathbb{R}^{n \times 2m(p+q)}$ and ℓ factors $\hat{Y}_m(t_j) \in \mathbb{R}^{2m(p+q) \times r}$, $j = 1, \dots, \ell$.

659

660

REFERENCES

- 661 [1] *Oberwolfach model reduction benchmark collection.*, 2003.
 662 [2] H. ABOU-KANDIL, G. FREILING, V. IONESCU, AND G. JANK, *Matrix Riccati Equations in Control*
 663 *and Systems Theory*, Birkhauser, 2003.
 664 [3] V. ANGELOVA, M. HACHED, AND K. JBILOU, *Approximate solutions to large nonsymmetric*
 665 *differential Riccati problems with applications to transport theory*, *Numer. Linear Algebra*
 666 *Appl.*, 27 (2020), p. e2272.
 667 [4] A. C. ANTOULAS, *Approximation of large-scale dynamical systems*, vol. 6, SIAM, 2005.
 668 [5] M. BEHR, P. BENNER, AND J. HEILAND, *On an invariance principle for the solution space of*
 669 *the differential Riccati equation*, *Proc. Appl. Math. Mech.*, 18 (2018), p. e201800031.
 670 [6] ———, *Invariant Galerkin ansatz spaces and Davison-Maki methods for the numerical solution*
 671 *of differential Riccati equations*, arXiv preprint arXiv:1910.13362, (2019).
 672 [7] P. BENNER, A. COHEN, M. OHLBERGER, AND K. WILLCOX, *Model reduction and approximation*
 673 *theory and algorithms*, SIAM, Philadelphia, 2017.
 674 [8] P. BENNER, V. MEHRMANN, AND D. SORENSEN, *Dimension Reduction of Large-Scale Systems*,
 675 Springer-Verlag, Berlin/Heidelberg, Germany, 2005.
 676 [9] D. A. BINI, B. LANNAZZO, AND B. MEINI, *Numerical solution of algebraic Riccati equations*,
 677 vol. 9, SIAM, 2012.
 678 [10] R. BITMEAD, M. GEVERS, I. PETERSEN, AND J. KAYE, *Monotonicity and stabilizability-*
 679 *properties of solutions of the Riccati difference equation: Propositions, lemmas, theorems,*
 680 *fallacious conjectures and counterexamples*, *Systems & Control Letters*, 5 (1985), pp. 309–
 681 315.

- 682 [11] S. BLANES, *High order structure preserving explicit methods for solving linear-quadratic optimal*
683 *control problems and differential games*, Numerical Algorithms, 69 (2015), pp. 271—290.
- 684 [12] C. CHOI AND A. LAUB, *Efficient matrix-valued algorithms for solving stiff Riccati differential*
685 *equations*, IEEE Trans. Automat. Control, 35 (1990), p. 770–776.
- 686 [13] M. J. CORLESS AND A. E. FRAZHO, *Linear systems and control - an operator perspective*,
687 Marcel Dekker, New York Basel, 2003.
- 688 [14] E. DAVISON AND M. MAKI, *The numerical solution of the matrix Riccati differential equation*,
689 IEEE Trans Autom Control, (1973), pp. 71–73.
- 690 [15] L. DIECI, *Numerical integration of the differential Riccati equation and some related issues*,
691 SIAM J. Numer. Anal., 29 (1992), pp. 781–815.
- 692 [16] L. DIECI AND T. EIROLA, *Preserving monotonicity in the numerical solution of Riccati differ-*
693 *ential equations*, Numer Math, 74 (1996), pp. 35–47.
- 694 [17] V. DRUSKIN AND L. KNIZHNERMAN, *Extended Krylov subspaces: approximation of the matrix*
695 *square root and related functions*, SIAM J Matrix Anal Appl, 19 (1998), pp. 755–771.
- 696 [18] V. DRUSKIN AND V. SIMONCINI, *Adaptive rational Krylov subspaces for large-scale dynamical*
697 *systems*, Systems & Control Letters, 60 (2011), pp. 546–560.
- 698 [19] M. GEVERS, R. BITMEAD, I. PETERSEN, AND J. KAYE, *When is the solution of the Riccati*
699 *equation stabilizing at every instant?*, in Frequency Domain and State Space Methods for
700 Linear Systems, North-Holland, 1986, pp. 531–540.
- 701 [20] L. GRASEDYCK, *Existence and computation of low Kronecker-rank approximations for large*
702 *linear systems of tensor product structure*, Computing., 72 (2004), pp. 247–265.
- 703 [21] ———, *Existence of a low rank or \mathcal{H} -matrix approximant to the solution of a Sylvester equation*,
704 Numer. Linear Algebra Appl., 11 (2004), pp. 371–389.
- 705 [22] N. GUGLIELMI AND V. SIMONCINI, *On the existence and approximation of a dissipating feedback*,
706 arXiv preprint arXiv:1811.00069, (2019).
- 707 [23] Y. GULDOGAN, M. HACHED, K. JBILOU, AND M. KURULAYA, *Low rank approximate solutions to*
708 *large-scale differential matrix Riccati equations*, Applicationes Mathematicae, 45 (2018),
709 pp. 233–254.
- 710 [24] M. HEYOUNI AND K. JBILOU, *An extended block arnoldi algorithm for large-scale solutions*
711 *of the continuoustimealgebraic riccati equation*, Electron Trans Numer Anal, 33 (2009),
712 pp. 53–62.
- 713 [25] I. M. JAIMOUKHA AND E. M. KASENALLY, *Krylov subspace methods for solving large Lyapunov*
714 *equations*, SIAM J. Numer. Anal., 31 (1994), pp. 227–251.
- 715 [26] M. KONSTANTINOV AND G. PELOVA, *Sensitivity of the solutions to differential matrix Riccati*
716 *equations*, IEEE Trans. Autom. Control, 36 (1991), pp. 213–215.
- 717 [27] A. KOSKELA AND H. MENA, *Analysis of Krylov subspace approximation to large scale differen-*
718 *tial Riccati equations*, arXiv preprint arXiv:1705.07507, (2018).
- 719 [28] V. KUCERA, *A review of the matrix Riccati equation*, Kibernetika, 9 (1973), pp. 42–61.
- 720 [29] H. KWAKERNAAK AND R. SIVAN, *Linear optimal control systems*, vol. 1, Wiley-interscience New
721 York, 1972.
- 722 [30] N. LANG, *Numerical methods for large-scale linear time-varying control systems and related*
723 *differential matrix equations*, PhD thesis, Technische Universitaet Chemnitz, 2017.
- 724 [31] Y. LIN AND V. SIMONCINI, *Minimal residual methods for large scale Lyapunov equations*, Ap-
725 *plied Num Math*, 72 (2013), pp. 52–71.
- 726 [32] ———, *A new subspace iteration method for the algebraic riccati equation*, Numerical Linear
727 Algebra w/App., 22 (2015), pp. 26–47.
- 728 [33] THE MATHWORKS, *MATLAB 7*, r2013b ed., 2013.
- 729 [34] H. MENA, *Numerical Solution of Differential Riccati Equations Arising in Optimal Control*
730 *Problems for Parabolic Partial Differential Equations*, PhD thesis, Escuela Politécnica
731 Nacional, 2007.
- 732 [35] H. MENA, A. OSTERMANN, L.-M. PFURTSCHELLER, AND C. PIAZZOLA, *Numerical low-rank*
733 *approximation of matrix differential equations*, J. Comput. Appl. Math., 340 (2018), pp. 602–
734 614.
- 735 [36] T. PENZL, *A cyclic low-rank smith method for large sparse lyapunov equations*, SIAM J Sci
736 Comput, 21 (2000), pp. 1401–1418.
- 737 [37] M.-A. POUBELLE, R. BITMEAD, AND M. GEVERS, *Fake algebraic Riccati techniques and stabil-*
738 *ity*, IEEE Trans. Autom. Control, 33 (1988), pp. 379–381.
- 739 [38] W. REID, *Riccati differential equations*, Academic Press, New York, 1972.
- 740 [39] A. RUHE, *Rational Krylov sequence methods for eigenvalue computation*, Lin. Alg. Appl., 58
741 (1984), pp. 391–405.
- 742 [40] ———, *The rational Krylov algorithm for nonsymmetric eigenvalue problems. III:Complex*
743 *shifts for real matrices*, BIT, 34 (1994), pp. 165–176.

- 744 [41] J. SAAK, M. KÖHLER, AND P. BENNER, *M-m.e.s.s.-1.0.1 – the matrix equations sparse solvers*
745 *library*. DOI:10.5281/zenodo.50575, Apr. 2016. see also:[www.mpi-magdeburg.mpg.de/](http://www.mpi-magdeburg.mpg.de/projects/mess)
746 [projects/mess](http://www.mpi-magdeburg.mpg.de/projects/mess).
- 747 [42] V. SIMONCINI, *A new iterative method for solving large-scale lyapunov matrix equations*, SIAM
748 *J Sci Comput*, 29 (2007), pp. 1268–1288.
- 749 [43] ———, *Analysis of the rational Krylov subspace projection method for large-scale algebraic*
750 *Riccati equations*, SIAM J. Matrix Anal. Appl., 37 (2016), pp. 1655–1674.
- 751 [44] ———, *Computational methods for linear matrix equations*, SIAM Rev, 58 (2016), pp. 377–441.
- 752 [45] V. SIMONCINI, D. B. SZYLD, AND M. MONSALVE, *On two numerical methods for the solution*
753 *of large-scale algebraic Riccati equations*, IMA Journal of Numerical Analysis, 34 (2014),
754 pp. 904–920.
- 755 [46] T. STILLFJORD, *Low-rank second-order splitting of large-scale differential Riccati equations*,
756 *IEEE Trans Automat Control*, 60 (2015), pp. 2791–2796.
- 757 [47] ———, *Adaptive high-order splitting schemes for large-scale differential Riccati equations*, *Nu-*
758 *mer Algor*, 78 (2018), pp. 1129–1151.
- 759 [48] ———, *Singular value decay of operator-valued differential Lyapunov and Riccati equations*,
760 *SIAM J. Control Optim.*, 56 (2018), pp. 3598–3618.