Hermann Mena Peter Benner

Rosenbrock methods for solving differential Riccati equations

Max Planck Institute Magdeburg Preprints

MPIMD/11-06 **September 27, 2011**

Impressum:

Max Planck Institute for Dynamics of Complex Technical Systems, Magdeburg

Publisher:
Max Planck Institute for Dynamics of Complex Technical Systems

Address:
Max Planck Institute for Dynamics of Complex Technical Systems
Sandtorstr. 1
39106 Magdeburg

www.mpi-magdeburg.mpg.de/preprints

Rosenbrock methods for solving differential Riccati equations

Peter Benner Hermann Mena

Abstract

The differential Riccati equation (DRE) arises in several fields like optimal control, optimal filtering, $\mathbf{H}_\infty$ control of linear-time varying systems, differential games, etc. In the literature there is a large variety of approaches to compute its solution. Particularly for stiff DREs, matrix-valued versions of the standard multi-step methods for solving ordinary differential equations have given good results. In this paper we discuss a particular class of one-step methods. These are the linear-implicit Runge-Kuta methods, i.e, the so called Rosenbrock methods. We show that they offer a practical alternative for solving stiff DREs. They can be implemented with good stability properties and they allow a cheap way to control the step size. The matrix valued version of the Rosenbrock methods for DREs requires the solution of one Sylvester equation in each stage of the method. For the case in which the coefficient matrices of the Sylvester equation are dense, the Bartels-Stewart method can be efficiently applied for solving the equations. The computational cost (computing time and memory requirements) is smaller than for the multi step methods.

Index Terms

DRE, Rosenbrock methods, linear-implicit Runge-Kuta, Sylvester equation

I. Introduction

We consider time-varying differential Riccati equations (DREs) of the form

$$\begin{aligned} \dot{X}(t) &= Q(t) + X(t)A(t) + B(t)X(t) - X(t)R(t)X(t) \equiv F(t, X(t)), \\ X(t_0) &= X_0, \end{aligned} \tag{1}$$

where $t \in [t_0, t_f]$, $A(t) \in \mathbb{R}^{n\times n}$, $B(t) \in \mathbb{R}^{m\times m}$, $Q(t) \in \mathbb{R}^{m\times n}$, $R(t) \in \mathbb{R}^{n\times m}$, $X(t) \in \mathbb{R}^{m\times n}$. We assume that the coefficient matrices are piecewise continuous locally bounded matrix-valued functions, which ensure existence of the solution and uniqueness of (1), see, e.g., [2, Thm. 4.1.6]. The DRE is called symmetric, if $Q(t), R(t)$ are square, symmetric matrices and $B(t) = A(t)^T$ for all $t \in [t_0, t_f]$.

Symmetric DREs arise from linear-quadratic optimal control problems like LQR and LQG design with finite-time horizon, in H_∞ control of linear-time varying systems as well as in differential games; see, e.g., [2], [15], [20]. Unfortunately, in most control problems fast and slow modes are present. This implies that the associated DRE will be fairly stiff, which in turn demands for implicit methods to solve such DREs numerically. Therefore, we will focus here on the stiff case. In this context, several approaches to solve DREs have been proposed, [11], [16], [18]. Particularly, matrix-valued algorithms for solving DREs based on generalizations of the BDF methods have been considered, see [3], [10], [12]. These methods are also suitable for large scale differential Riccati equations (DREs) arising in optimal control problems for parabolic partial differential equations [5], [19].

In general, the BDF methods require fewer function evaluations per step than one step methods, and they allow a simpler streamlined method design from the point of view of order and error estimation. However, the associated overhead is higher for changing the step size compared with one step methods. Diagonally implicit Runge-Kutta methods or collocation methods offer an alternative to the BDF methods for solving stiff problems. Especially, linearly implicit one-step methods (better known as Rosenbrock methods) give

Max Planck Institute for Dynamics of Complex Technical Systems, 39106 Magdeburg, and TU Chemnitz, Germany benner@mpi-magdeburg.mpg.de

Departamento de Matemática, Escuela Politécnica Nacional, Ladrón de Guevara E11-253, Quito-Ecuador, hermann.mena@epn.edu.ec

satisfactory results see, e.g., [7], [14]. The idea of these methods can be interpreted as the application of one Newton iteration to each stage of an implicit Runge-Kutta method and the derivation of stable formulae by working with the Jacobian matrix directly within the integration formulae. Here, we derive a matrix valued version of the Rosenbrock methods for solving DREs.
In the literature, variants of the Rosenrock method are discussed, in which the Jacobian matrix is retained over several steps or even replaced by an approximation which renders the linear system cheaper. Methods constructed in this way were first studied by T. Steihaug and A. Wolfbrand in 1979. Since they denoted the inexact Jacobi matrix by "W", these methods are often called W-methods. In general, they are a practical alternative when low accuracy is prescribed. The application of these ideas to the matrix-valued Rosenbrock methods discussed is straight forward.
This paper is organized as follows. First, we will review matrix-valued versions of numerical methods for DREs. Then, in Section II we describe the Rosenbrock methods, discuss stability properties and show some particular schemes. The application of this method to DREs is shown in Section III, as well as implementation details. In Section IV numerical experiments are discussed. A brief summary and outlook on future work concludes the paper.

II. Rosenbrock methods

As described in [10], the numerical methods for solving DREs of the form (1) can essentially be distinguished into five classes: unroll the matrices into vectors and integrate the resulting system, linearize the DRE and solve the resulting Hamiltonian type system, transform the DRE into two coupled nonlinear equations (the so-called Chandrasekhar system), methods based on the superposition property of the Riccati solutions and matrix versions of the ODE methods. The latter is in general the most efficient method, we refer the reader to [10], [12] and references therein for a detailed explanation. Moreover, as we consider stiff DREs the BDF methods are the natural choice. These methods solve the DRE using matrix-valued algorithms based on standard BDF methods for solving ODEs [10], [12].The application of the BDF methods yields an algebraic Riccati equation (ARE) to be solved in each time step. Since the ARE is a nonlinear matrix equation, it is natural to apply Newton's method or variants of it. The application of Newton's method yields a Sylvester equation to be solved in every step, details and further references can be found in [17].
Other matrix-valued algorithms yields an ARE to be solved in every step also, e.g., the Midpoint or Trapezodial rules. In fact, as stated in [12], mainly every implicit scheme yields an ARE to be solved in every step. For stiff DREs the BDF methods are a good choice among multistep methods [10]. In the following we will see that Rosenbrock methods offer a practical alternative to the BDFs methods.

Let us consider a non-autonomous ODE system

$$\dot{x} = f(t,x), \quad x(t_0) = x_0, \tag{2}$$

where $f : [t_0, t_f] \times \mathbb{R}^n \to \mathbb{R}^n$ is bounded in the neighborhood of the initial point and Lipschitz continuous in x. Then, the general s-stage Rosenbrock method for (2) is given by

$$\begin{aligned} k_i &= hf\big(t_k + \alpha_i h, x_k + \textstyle\sum_{j=1}^{i-1} \alpha_{ij} k_j\big) + \gamma_i h^2 \tfrac{\partial f}{\partial t}(t_k, x_k) \\ &\quad + h \tfrac{\partial f}{\partial x}(t_k, x_k) \textstyle\sum_{j=1}^{i} \gamma_{ij} k_j, \qquad\qquad i = 1, \dots, s, \\ x_{k+1} &= x_k + \textstyle\sum_{j=1}^{s} b_j k_j, \end{aligned} \tag{3}$$

where α_{ij}, γ_{ij}, b_j are the determining coefficients and h is the step size. The additional coefficients are given by

$$\alpha_i = \sum_{j=1}^{i-1} \alpha_{ij}, \qquad \gamma_i = \sum_{j=1}^{i} \gamma_{ij}. \tag{4}$$

s	A-stability	L-stability
1	$1/2 \leq \gamma \leq \infty$	
2	$1/4 \leq \gamma \leq \infty$	$(2-\sqrt{2})/2 \leq \gamma \leq (2+\sqrt{2})/2$
3	$1/3 \leq \gamma \leq 1.06857902$	$0.18042531 \leq \gamma \leq 2.18560010$
4	$0.39433757 \leq \gamma \leq 1.28057976$	$0.22364780 \leq \gamma \leq 0.57281606$
5	$0.42078251 \leq \gamma \leq 0.47326839$	$0.24799464 \leq \gamma \leq 0.67604239$

TABLE I
A AND L-STABILITY FOR ROSENBROCK METHODS [14].

Each stage of this method consists of a system of linear equations with unknowns k_i. Note that for the autonomous case, i.e., $f(t,x) \equiv f(x)$, the coefficients from (4) do not appear and therefore the iteration is simplified. Also of special interest are the methods for which $\gamma_{11} = \cdots = \gamma_{ss} = \gamma$, then A-stability and L-stability can be achieved, see Table I, the order of the resulting method is s or $s+1$ for specific values of γ. That is because in general Rosenbrock methods have the same stability function as diagonally implicit Runge-Kutta methods, for details we refer the reader to [14].
Rosenbrock methods are very attractive for several reasons like: excellent stability properties, easy to implement, cheap step size control, suitability for parallelization [8], [22], applicability to implicit systems of the form $M\dot{y} = f(y)$, among others. In order to avoid matrix-vector multiplications, in the implementation of the method new variables are introduced:

$$u_i = \sum_{j=1}^{i} \gamma_{ij} k_j.$$

Thus, the s-stage Rosenbrock method (3) can be written as

$$\begin{array}{rcl} (\frac{1}{h\gamma_{ii}} I - \frac{\partial f}{\partial x}(t_k, x_k))u_i &=& f\big(t_k + \alpha_i h, x_k + \sum_{j=1}^{i-1} a_{ij} u_j\big) \\ & & + \sum_{j=1}^{i-1} \frac{c_{ij}}{h} u_j + \gamma_i h \frac{\partial f}{\partial t}(t_k, x_k), \\ x_{k+1} &=& x_k + \sum_{j=1}^{s} m_j u_j, \end{array} \tag{5}$$

where $i = 1, \ldots, s$, and

$$\begin{array}{ll} \Gamma = (\gamma_{ij}), & C = \text{diag}(\gamma_{11}^{-1}, \ldots, \gamma_{ss}^{-1}) - \Gamma^{-1}, \\ (a_{ij}) = (\alpha_{ij})\Gamma^{-1}, & (m_1, \ldots, m_s) = (b_1, \ldots, b_s)\Gamma^{-1}. \end{array} \tag{6}$$

The Rosenbrock methods have been deeply studied, e.g., in [14].

A. Some popular Rosenbrock schemes

a) Linearly implicit Euler method: The one stage Rosenbrock method, also known as the linearly implicit Euler method, for non-autonomous systems becomes

$$\begin{array}{rcl} (I - \frac{\partial f}{\partial x}(t_k, x_k)h)k_1 &=& f(x_k) + h\frac{\partial f}{\partial t}(t_k, x_k), \\ x_{k+1} &=& x_k + hk_1. \end{array} \tag{7}$$

The coefficients are chosen as

$$b_1 = 1, \qquad \gamma = 1, \qquad \alpha_{11} = 0.$$

The method is of order $p = 1$ and the stability function is the same as the one for the implicit Euler method.

b) A second order method: In [6] a second order method (for autonomous ODE systems) is described for application to atmospheric dispersion problems describing photochemistry, advective, and turbulent diffusive transport. For non-autonomous systems the scheme can be written as

$$\begin{aligned}(I-\gamma h\tfrac{\partial f}{\partial x}(t_k,x_k))k_1 &= f(t_k,x_k)+\gamma h\tfrac{\partial f}{\partial t}(t_k,x_k),\\ (I-\gamma h\tfrac{\partial f}{\partial x}(t_k,x_k))k_2 &= f(t_k+h,x_k+hk_1)-2k_1-\gamma h\tfrac{\partial f}{\partial t}(t_k,x_k),\\ x_{k+1} &= x_k+\tfrac{h}{2}(3k_1+k_2).\end{aligned} \tag{8}$$

It is pointed out by the authors that the method is capable of integrating with large a priori described step sizes for the problem that they are dealing with [6].

c) A fourth order method: The most popular Rosenbrock methods are of fourth order: Kaps-Rentrop (1979), Shampine (1982) and Veldhuizen (1984). The scheme proposed by Shampine can be written as

$$\begin{aligned}Ek_1 &= f(t_k,x_k)+\tfrac{1}{2}h\tfrac{\partial f}{\partial t}(t_k,x_k),\\ Ek_2 &= f(t_k+h,x_k+hk_1)-\tfrac{3}{2}h\tfrac{\partial f}{\partial t}(t_k,x_k)-4k_1,\\ Ek_3 &= f\Big(t_k+\tfrac{3}{5}h,x_k+\tfrac{24}{25}hk_1+\tfrac{3}{25}hk_2\Big)+\tfrac{121}{50}h\tfrac{\partial f}{\partial t}(t_k,x_k)\\ &\quad +\tfrac{185}{25}k_1+\tfrac{6}{5}k_2,\\ Ek_4 &= f\Big(t_k+\tfrac{3}{5}h,x_k+\tfrac{24}{25}hk_1+\tfrac{3}{25}hk_2\Big)+\tfrac{29}{250}h\tfrac{\partial f}{\partial t}(t_k,x_k)\\ &\quad -\tfrac{56}{125}k_1-\tfrac{27}{125}k_2-\tfrac{1}{5}k3,\\ x_{k+1} &= x_k+h\Big(\tfrac{19}{18}k_1+\tfrac{1}{4}k_2+\tfrac{25}{216}k_3+\tfrac{125}{216}k_4\Big),\end{aligned} \tag{9}$$

where $E=I-\frac{1}{2}h\frac{\partial f}{\partial x}(t_k,x_k)$, see [21] for details.

d) A stiff-nonstiff method: Most codes for solving initial value problems are intended to deal with stiff or nonstiff problems but not both. Rosenbrock methods allow to deal with this issue choosing the coefficients of the method common to the ones of an explicit Runge-Kutta method. In [13] the authors investigated this idea for a fourth order method. The result is a Rosenbrock integrator of order four that contains an explicit Runge-Kutta method embedded. It switches from one to the other solver, when the solution leaves a stiff domain and enters a nonstiff domain or vice versa. Even though it is not a simple task to determine when the solution leaves or enters in a stiff domain, this idea can be efficiently exploited to integrate some type of problems [13].

All the methods described above have already proven to be very effective in many applications, e.g. see [6], [13], [22]. Most of the computational cost is related with the evaluation of the partial derivatives of f at every step. In general the latter is considered expensive. In the next section we explain how to efficiently implement this for solving DREs.

III. Application to DREs

We consider time varying DREs of the form (1). The application of the Rosenbrock method (5), as a matrix-valued algorithm, to the DRE (1) yields

$$\begin{aligned}(\tfrac{1}{h\gamma_{ii}}I-\tfrac{\partial F}{\partial X}(t_k,X_k))K_i &= F\big(t_k+\alpha_i h,X_k+\textstyle\sum_{j=1}^{i-1}a_{ij}K_j\big)\\ &\quad +\textstyle\sum_{j=1}^{i-1}\frac{c_{ij}}{h}K_j+\gamma_i hF_{t_k},\\ X_{k+1} &= X_k+\textstyle\sum_{j=1}^{s}m_jK_j,\end{aligned} \tag{10}$$

where $i=1,\dots,s$, and the coefficients of the method are defined as in (4) and (6). We use K_i instead of u_i, note that K_i represent $n\times n$ matrix. We also note $F_{t_k}=\frac{\partial F}{\partial t}(t_k,X(t_k))$.

The derivative $\frac{\partial F}{\partial X}(t_k, X_k)$ in (10) is given by the (Frechét) derivative of F at X_k, represented by the Sylvester operator

$$\frac{\partial F}{\partial X}(t_k, X_k) : U \to (B_k^T - R_k X_k)^T U + U(A_k - R_k X_k),$$

where $X_k \approx X(t_k)$, $A_k \equiv A(t_k)$, $B_k \equiv B(t_k)$, $R_k \equiv R(t_k)$ and $U \in \mathbb{R}^{n\times n}$.
Replacing $\frac{\partial F}{\partial X}(t_k, X_k)$ in the left hand side of the first equation in (10) we obtain,

$$\frac{1}{h\gamma_{ii}} K_i - (B_k^T - R_k X_k)^T K_i - K_i(A_k - R_k X_k),$$

and re-arranging terms yields

$$-\left(\left(B_k^T - R_k X_k - \frac{1}{2h\gamma_{ii}} I\right)^T K_i + K_i\left(A_k - R_k X_k - \frac{1}{2h\gamma_{ii}} I\right)\right).$$

Defining

$$\bar{A}_k = A_k - R_k X_k - \frac{1}{2h\gamma_{ii}} I, \qquad \bar{B}_k = \left(B_k^T - R_k X_k - \frac{1}{2h\gamma_{ii}} I\right)^T,$$

we can write (10) as

$$\begin{array}{rcl} \bar{B}_k K_i + K_i \bar{A}_k & = & -F\big(t_k + \alpha_i h, X_k + \sum_{j=1}^{i-1} a_{ij} K_j\big) \\ & & - \sum_{j=1}^{i-1} \frac{c_{ij}}{h} K_j - \gamma_i h F_{t_k}, \\ X_{k+1} & = & X_k + \sum_{j=1}^{s} m_j K_j, \end{array} \tag{11}$$

where $i = 1, \ldots, s$. Hence, one Sylvester equation has to be solved in each stage of the method every time step. Note that for the matrix valued BDF methods applied to DREs one ARE has to be solved every step, which requires the solution of one Sylvester equation in every Newton iteration [17]. Therefore, the computational cost for solving DREs using Rosenbrock methods is drastically reduced compared with the one using the BDF methods.
For the case in which the coefficient matrices of Sylvester equations are dense, the Bartels-Stewart method [4] can be efficiently applied. We point out that if the number of stages of the method is bigger than one, i.e., $s > 1$, only one Schur decomposition is needed so the cost of solving s Sylvester equations is almost the same as for solving one equation. This can be exploited using, e.g., the Sylvester solver from SLICOT[1], which is accesible in MATLAB as function `slsylv` provided by the SLICOT Basic Systems and Control Toolbox[2]. This property can be used to efficiently implement the method, especially if the dimension of the problem is large, i.e., the size of the matrix X.
Note that working with an s-stage Rosenbrock method the intermediate approximation at stage $s-1$ can be used to cheaply estimate the local error for controling the step size. For instance, the intermediate approximation of the second order Rosenbrock method (8) corresponds to the application of the linearly implicit Euler method at t_{k+1}. Taking this into account a general implementation of the s-stage Rosenbrock method using step size control is sketched in Algorithm III.1. There, the growth of the step is limited by a factor $q > 1$ and the maximum step size allowed $h_{\max}$.

A. *Implementation details*

If $F\big(t_k + \alpha_i h, X_k + \sum_{j=1}^{i-1} a_{ij} K_j\big)$ in (11) is written as

$$\begin{array}{c} \big(B_{k_h}^T - R_{k_h}\bar{X}_k - \frac{1}{2h\gamma_{ii}} I\big)^T \bar{X}_k + \bar{X}_k\big(A_{k_h} - R_{k_h}\bar{X}_k - \frac{1}{2h\gamma_{ii}} I\big) \\ + Q_{k_h} + \bar{X}_{k_h} R_{k_h} \bar{X}_k + \frac{1}{h\gamma_{ii}} \bar{X}_k, \end{array} \tag{12}$$

[1]Subroutine Library in Control, available at www.slicot.org
[2]See http://www.slicot.org/index.php?site=slbasic

Algorithm III.1 s-stage Rosenbrock for DREs using step size control

Require: Let h_0 be the initial step size, $[a, b]$ the integration interval, X_0 the initial condition, $\rho < 1$ and $q > 1$ safety parameters, Tol desired integration error, and $h_{\max}$ maximum step size allowed.

Ensure: (X_k, t_k) such that $X_k \approx X(t_k)$.

$k = 0$.
$t_0 = a$.
while $t_k < b$ **do**
  $t = t_k + h_k$
  for $i = 1$ to s **do**
    $\bar{A}_k = A_k - R_k X_k - \frac{1}{2h\gamma_{ii}} I$
    $\bar{B}_k = \left(B_k^T - R_k X_k - \frac{1}{2h\gamma_{ii}} I \right)^T$
    Solve the Sylvester equation

$$\begin{aligned} \bar{B}_k K_i + K_i \bar{A}_k &= -F\big(t_k + \alpha_i h, X_k + \textstyle\sum_{j=1}^{i-1} a_{ij} K_j\big) \\ &\quad - \textstyle\sum_{j=1}^{i-1} \frac{c_{ij}}{h} K_j - \gamma_i h F_{t_k} \end{aligned}$$

  end for
  $\hat{Y}_{k+1} = X_k + \sum_{j=1}^{s-1} m_j K_j$
  $Y_{k+1} = X_k + \sum_{j=1}^{s} m_j K_j$
  $\epsilon_k = \|Y_{k+1} - \hat{Y}_{k+1}\|$
  $h = \min(q h_k, h_{\max}, \sqrt[s]{\frac{\rho \cdot Tol}{\epsilon_k}} h_k)$
  if $\epsilon_k < Tol$ **then**
    $t_{k+1} = t$
    $X_{k+1} = Y_{k+1}$
    $h_{k+1} = \min(h, b - t_{k+1})$
    $k = k + 1$
  else
    $h_k = h$
  end if
end while

where $A_{k_h} = A(t_k + \alpha_i h)$, $B_{k_h} = B(t_k + \alpha_i h)$, $R_{k_h} = R(t_k + \alpha_i h)$ and $\bar{X}_k = X_k + \sum_{j=1}^{i-1} a_{ij} K_j$, then, in some cases we can simplify the implementation of the method.

Let us consider the linearly implicit Euler method applied to DREs

$$\begin{aligned} \bar{B}_k K_1 + K_1 \bar{A}_k &= -F(X_k) - h F_{t_k}, \\ X_{k+1} &= X_k + K_1, \end{aligned} \tag{13}$$

where $\bar{A}_k = A_k - R_k X_k - \frac{1}{2h} I$ and $\bar{B}_k = \left(B_k^T - R_k X_k - \frac{1}{2h} I \right)^T$. If we write $F(X_k)$ analogously to (12), then we can re-write (13) such that the next iterate is computed directly from the Sylvester equation,

$$\bar{B}_k X_{k+1} + X_{k+1} \bar{A}_k = -Q_k - X_k R_k X_k - \frac{1}{h} X_k - h F_{t_k}. \tag{14}$$

The right hand side of (14) is simpler to evaluate than the one in (13), so the implementation of (14) is more efficient.

Let us now consider the second-order Rosenbrock method (8), as a matrix-valued algorithm, applied to

the DRE (1)

$$\begin{aligned} \bar{B}_k K_1 + K_1 \bar{A}_k &= -F(t_{k+1}, X_k) - h\gamma F_{t_k}, \\ \bar{B}_k K_2 + K_2 \bar{A}_k &= -F(t_{k+1}, X_k + hK_1) + 2K_1 + h\gamma F_{t_k}, \\ X_{k+1} &= X_k + \tfrac{3}{2}K_1 + \tfrac{1}{2}K_2, \end{aligned} \tag{15}$$

where $\bar{A}_k = A_k - R_k X_k - \frac{1}{2h\gamma} I$, $\bar{B}_k = \left(B_k^T - R_k X_k - \frac{1}{2h\gamma} I \right)^T$ and $t_{k+1} = t_k + h$.
Rewriting the right hand side of the second Sylvester equation in (15) analogously to (12) as

$$\begin{aligned} &-F(t_{k+1}, X_k) + \tfrac{1}{h\gamma} K_1 + \gamma h F_{t_k} - h^2 K_1 R_{k+1} K_1 + 2K_1 \\ &- \left(B_{k+1}^T - R_{k+1} X_k - \tfrac{1}{2h\gamma} I \right)^T K_1 - K_1 \left(A_{k+1} - R_{k+1} X_k - \tfrac{1}{2h\gamma} I \right), \end{aligned} \tag{16}$$

under the assumption that the size of the matrix X is large and due to the linearity of the Sylvester equation, it is more efficient to solve an additional equation (with the same coefficient matrices $\bar{A}_k$ and $\bar{B}_k$) than solving (15) directly. The right hand side of this equation is chosen as the common factor of the right hand sides of the equations involved in each stage. Then, the original solution is recovered afterwards. The standard implementation of this second order Rosenbrock method is sketched in Algorithm III.2, which can be endowed with step size control as in Algorithm III.1.
Again, note that the additional solve is cheap if we exploit the decompositions of $\bar{A}_K$, $\bar{B}_k$ and only perform the backward substitution step of the Hessenberg-Schur or Bartels-Stewart methods. Finally, we point out that a similar implementation can be straightforward performed for any s-stage Ronsenbrock method.

Algorithm III.2 Rosenbrock method of second order

Require: $Q(t)$, $A(t)$, $S(t)$, $\in \mathbb{R}^{n \times n}$ are piecewise continuous locally bounded matrix-valued functions $t \in [a, b]$, X_0, and h is the step size.
Ensure: (X_k, t_k) such that $X_k \approx X(t_k)$.

1: $t_0 = a$.
2: **for** $k = 0$ to $\lceil \frac{b-a}{h} \rceil$ **do**
3: $t_{k+1} = t_k + h$
4: $\bar{A}_k = A_k - R_k X_k - \frac{1}{2\gamma h} I$
5: $\bar{B}_k = \left(B_k^T - R_k X_k - \frac{1}{2\gamma h} I \right)^T$
6: Solve the Sylvester equation $\bar{B}_k K_{11} + K_{11} \bar{A}_k = -F(t_{k+1}, X_k)$
7: Solve the Sylvester equation $\bar{B}_k K_{12} + K_{12} \bar{A}_k = -F_{t_k}$
8: $K_1 = K_{11} + \gamma h K_{12}$
9: Solve the Sylvester equation

$$\begin{aligned} \bar{B}_k K_{21} + K_{21} \bar{A}_k &= -h^2 K_1 R_{k+1} K_1 - F(t_{k+1}, X_k) + \left(\frac{1}{h\gamma} + 2 \right) K_1 \\ &- \left(B_{k+1}^T - R_{k+1} X_k - \frac{1}{2h\gamma} I \right)^T h K_1 - h K_1 \left(A_{k+1} - R_{k+1} X_k - \frac{1}{2h\gamma} I \right) \end{aligned}$$

10: $K_2 = K_{21} - \gamma h K_{12}$
11: $X_{k+1} = X_k + \frac{3}{2} h K_1 + \frac{1}{2} h K_2$
12: **end for**

B. Autonomous DRE

For autonomous DREs, i.e., DREs, in which the coefficients matrices $Q(t)$, $A(t)$, $R(t)$ are constant, $F(t_{k+1}, X_k) \equiv F(X_k)$, so $F_{t_k} = 0$. Particularly, for the second order Rosenbrock method this term, arising in the second Sylvester equation,

$$-\left(B^T - RX_k - \frac{1}{2h\gamma}I\right)^T hK_1 - hK_1\left(A - RX_k - \frac{1}{2h\gamma}I\right) \tag{17}$$

is equal to $hF(X_k)$ because of the linearity of the solution. Therefore, the method can be written like:

$$\begin{aligned} X_{k+1} &= X_k + \frac{3}{2}hK_1 + \frac{1}{2}hK_2, && (18)\\ \hat{B}_k K_1 + K_1 \hat{A}_k &= -F(X_k), && (19)\\ \hat{B}_k K_2 + K_2 \hat{A}_k &= -h^2 K_1 R K_1 + \left(\frac{1}{h\gamma} + 2\right)K_1 + (h-1)F(X_k), && (20) \end{aligned}$$

where $\hat{A}_k = A - RX_k - \frac{1}{2h\gamma}I$, $\hat{B}_k = \left(B^T - RX_k - \frac{1}{2h\gamma}I\right)^T$. Moreover, it can be efficiently computed as:

$$\begin{aligned} X_{k+1} &= X_k + \frac{3}{2}hK_1 + \frac{1}{2}hK_2, && (21)\\ \hat{B}_k K_1 + K_1 \hat{A}_k &= -F(X_k), && (22)\\ \hat{B}_k K_{21} + K_{21} \hat{A}_k &= -h^2 K_1 R K_1 + \left(\frac{1}{h\gamma} + 2\right)K_1, && (23)\\ K_2 &= K_{21} - (h-1)K_1. && (24) \end{aligned}$$

In control theory, particularly solving finite dimensional linear quadratic control problems we have to deal with symmetric DREs. Thus, instead of a Sylvester equation, the solution of a Lyapunov equation in every stage of a Rosenbrock method is required. Particularly, for large scale DREs arising from the discretization in space of the optimal control problems governed by partial differential equations of parabolic type an efficient implementation of the Rosenbrock methods is proposed in [19]. The key ingredient there is to find a low rank approximation of the solution and to rewrite the method for the these low rank factors of the solution.

IV. Numerical results

A. Example 1

Let us now consider the following symmetric DRE of size n,

$$\begin{aligned} \dot{X}(t) &= -X^2(t) + k^2 I_n, \qquad t_0 \le t \le T\\ X(t_0) &= X_0. \end{aligned} \tag{25}$$

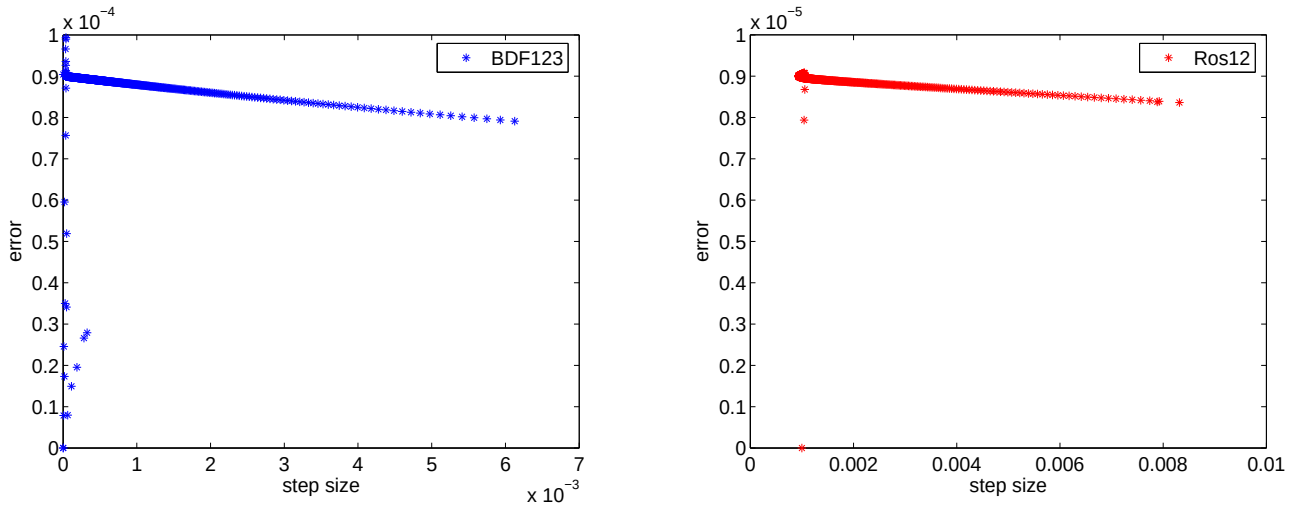
If X_0 is diagonalizable, i.e, $X_0 = S\Lambda S^{-1}$ with Λ=diag$[\lambda_i]$, then the analytic solution of (25) is:

$$X(t) = S\text{diag}\left[\frac{k\sinh kt + \lambda_i \cosh kt}{\cosh kt + \frac{\lambda_i}{k}\sinh kt}\right]S^{-1},$$

refer to [9] for a detailed explanation. Here, we choose

$$X_0 = I_n, \quad k = 3, \quad n = 60.$$

In Figure 1 the error vs. step size by a variable step size/order method is shown in (a) for the BDF methods up to order three (BDF123) and in (b) for the Rosenbrock method of up to order two (Ros12). The tolerance to accept or redo the current step was choosen as $Tol = 1e-4$. We can observe that the performance of the Rosenbrock of order two is competitive achieving higher precision with essentially the same step size at a low computational cost.



(a) (b)

Fig. 1. Example 1 (a) error vs. step size by the variable step size and order BDF method up to order 3 (BDF123), (b) error vs. step size y the variable step size Rosenbrock method of order two (Ros12)

B. *Example 2*

This example was taken from the SLICOT collection of benchmark examples for continuous-time algebraic Riccati equations [1]. Here, we consider the associated DRE

$$\begin{aligned}\dot{X}(t) &= (C^T\tilde{Q})(C^T\tilde{Q})^T + A^TX(t) + X(t)A \\ &\quad -X(t)(B\tilde{R}^{-1}\tilde{R})(B\tilde{R}^{-1}\tilde{R})^TX(t),\end{aligned} \tag{26}$$

where the matrices C, $\tilde{Q}$, A, B and $\tilde{R}$ come from the ARE arising in the discretized problem. The initial condition for (26) is equal to zero.
Besides the system dimension $n = 100$, the problem parameters are set as the default values of this benchmark example, [1].

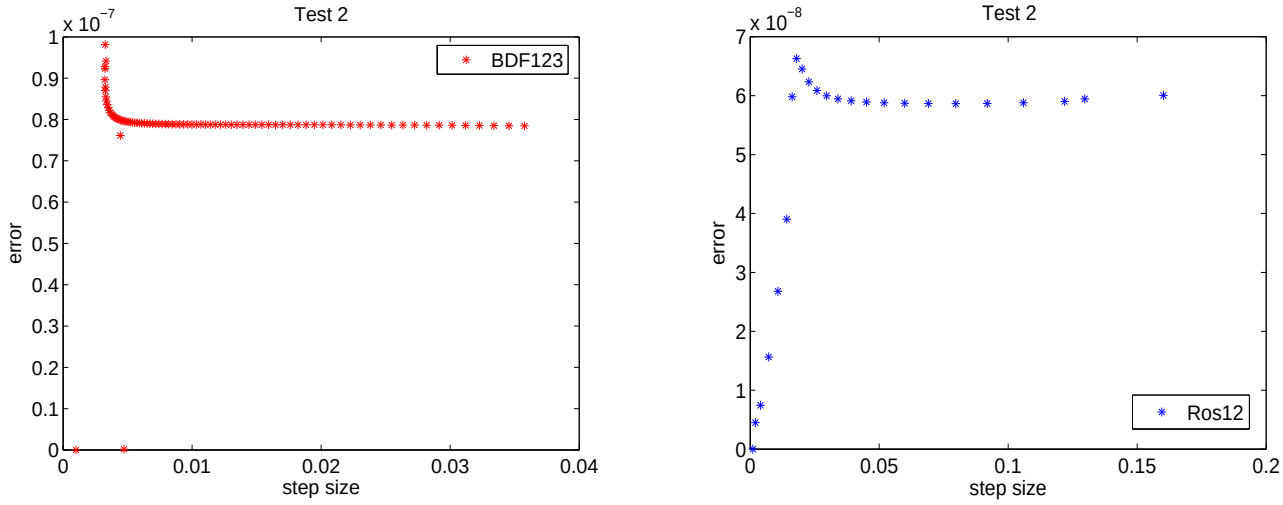
In Figure 2 the error vs. step size by BDF123 is shown in (a). The same data are plotted for Ros12 in (b), respectively. The tolerance for accept or redo the current step was choosen as $Tol = 1e-7$. As in the previous example we observe that the Rosenbrock methods achieve higher precision than BDF methods even with much fewer steps. Recall that in Rosenbrock methods, only Lyapunov equations need to be solved which is significantly cheaper than solving AREs as in BDF methods.

V. Conclusions

Solving differential Riccati equations is a central issue in many applications like, e.g., control design problems. By rewriting the Rosenbrock methods in terms of matrix operations, it turns out that a Sylvester(or Laypunov) equation has to be solved in each time step. The computational cost of solving DREs by this method is significantly smaller compared with the BDF methods which require the solution of an algebraic Riccati equations to be solved in every step. The good stability properties and other nice features of the Rosenbrock methods make these methods a practical alternative to efficiently solve DREs specially for the autonomous case which allows an efficient implementation.

References

[1] J. Abels and P. Benner. CAREX - a collection of benchmark examples for continuous-time algebraic Riccati equations. Technical report, 1999. SLICOT working note 1999-14. Available from http://www.slicot.org. 13



(a) (b)

Fig. 2. Example 2, interval of integration $[0, 1]$, $Tol = 1e-7$ (a) error vs. step size for BDF123 (b) error vs. step size for Ros12 .

[2] H. Abou-Kandil, G. Freiling, V. Ionescu, and G. Jank. *Matrix Riccati Equations in Control and Systems Theory*. Birkhäuser, Basel, Switzerland, 2003. 2

[3] E. Arias, V. Hernández, J. Ibáñes, and J. Peinado. A fixed point-based BDF method for solving Riccati equations. *Applied Mathematics and Computation*, 188:1319–1333, 2007. 2

[4] R.H. Bartels and G.W. Stewart. Solution of the matrix equation $AX + XB = C$: Algorithm 432. *Comm. ACM*, 15:820–826, 1972. 8

[5] P. Benner and H. Mena. BDF methods for large-scale differential Riccati equations. In B. De Moor, B. Motmans, J. Willems, P. Van Dooren, and V. Blondel, editors, *Proc. of Mathematical Theory of Network and Systems, MTNS 2004*, 2004. 2

[6] J.G. Blom, W. Hundsdorfer, E.J. Spee, and J.G. Verwer. A second order Rosenbrock method applied to photochemical dispersion problems. *SIAM J. Sci. Comput.*, 20(4):1456–1480, 1999. 5, 6

[7] F. Bornemann and P. Deuflhard. *Scientific Computing with Ordinary Differential Equations*, volume 42 of *Text in Applied Mathematics*. Springer-Verlag, New York, 2002. 2

[8] J. Bruder. Numerical results for a parallel linearly-implicit Runge-Kutta method. *Computing*, 59(2):139–151, 1997. 4

[9] C. Choi and A.J. Laub. Constructing Riccati differential equations with known analytic solutions for numerical experiments. *IEEE Trans. Automat. Control*, 35:437–439, 1990. 13

[10] C. Choi and A.J. Laub. Efficient matrix-valued algorithms for solving stiff Riccati differential equations. *IEEE Trans. Automat. Control*, 35:770–776, 1990. 2, 3

[11] E.J. Davison and M.C. Maki. The numerical solution of the matrix Riccati differential equation. *IEEE Trans. Automat. Control*, 18:71–73, 1973. 2

[12] L. Dieci. Numerical integration of the differential Riccati equation and some related issues. *SIAM J. Numer. Anal.*, 29(3):781–815, 1992. 2, 3

[13] A.D. Freed and I.s. Iskovitz. Development and applications of a Rosenbrock integrator. Technical Report NASA Tech. Memorandum 4709, Lewis Research Center, Cleveland, Ohio, 1996. 6

[14] E. Hairer and G. Wanner. *Solving Ordinary Differential Equations II-Stiff and Differential Algebraic Problems*. Springer Series in Computational Mathematics. Springer-Verlag, New York, 2000. 2, 4, 5

[15] A. Ichikawa and H. Katayama. Remarks on the time-varying H_∞ Riccati equations. *Sys. Cont. Lett.*, 37(5):335–345, 1999. 2

[16] C. Kenney and R.B. Leipnik. Numerical integration of the differential matrix Riccati equation. *IEEE Trans. Automat. Control*, AC-30:962–970, 1985. 2

[17] P. Lancaster and L. Rodman. *The Algebraic Riccati Equation*. Oxford University Press, Oxford, 1995. 3, 7

[18] A.J. Laub. Schur techniques for Riccati differential equations. In D. Hinrichsen and A. Isidori, editors, *Feedback Control of Linear and Nonlinear Systems*, pages 165–174, New York, 1982. Sringer-Verlag. 2

[19] H. Mena. *Numerical Solution of Differential Riccati Equations Arising in Optimal Control Problems for Parabolic Partial Differential Equations*. PhD thesis, Escuela Politecnica Nacional, 2007. 2, 12

[20] I.R. Petersen, V.A. Ugrinovskii, and A.V.Savkin. *Robust Control Design Using H^∞ Methods*. Springer-Verlag, London, UK, 2000. 2

[21] L. F. Shampine. Implementation of Rosenbrock Methods. *ACM Transactions on Mathematical Software*, 8(2):93–103, 1982. 6

[22] D. A. Voss and A. Q. M. Khaliq. Parallel Rosenbrock methods for chemical systems. *Computers and Chemistry*, 25:101–107, 2001. 4, 6