SLICOT Working Note 1999-14

CAREX — A Collection of Benchmark Examples for Continuous-Time Algebraic Riccati Equations (Version 2.0) [1]

Jörn Abels[2] Peter Benner[3]

November 1999

[1]This document presents research results of the European Community BRITE-EURAM III Thematic Networks Programme NICONET (contract number BRRT-CT97-5040) and is distributed by the Working Group on Software WGS. *WGS secretariat*: Mrs. Ida Tassens, ESAT - Katholieke Universiteit Leuven, K. Mercierlaan 94, 3001-Leuven-Heverlee, BELGIUM. This report is also available by anonymous ftp from `wgs.esat.kuleuven.ac.be` in the directory /pub/WGS/REPORTS/ with file name SLWN1999-14.ps.Z

[2]Fachbereich 3 – Mathematik und Informatik, Universität Bremen, D–28334 Bremen, Germany; `abels@informatik.uni-bremen.de`

[3]Zentrum für Technomathematik, Fachbereich 3 – Mathematik und Informatik, Universität Bremen, D–28334 Bremen, Germany; `benner@math.uni-bremen.de`

Abstract

A collection of benchmark examples is presented for the numerical solution of continuous-time algebraic Riccati equations. This collection may serve for testing purposes in the construction of new numerical methods, but may also be used as a reference set for the comparison of methods. The collected examples focus mainly on applications in linear-quadratic optimal control theory. This version updates an earlier benchmark collection and includes one new example.

0 Introduction

We present a collection of examples for continuous-time algebraic Riccati equations (CARE) of the form

$$0 = Q + A^T X + XA - XGX \tag{1}$$

where $A, G, Q, X \in \mathbb{R}^{n\times n}$. The matrices $Q = Q^T$ and $G = G^T$ may be given in factored form $Q = C^T\tilde{Q}C$, $G = BR^{-1}B^T$ with $C \in \mathbb{R}^{p\times n}$, $B \in \mathbb{R}^{n\times m}$, $\tilde{Q} = \tilde{Q}^T \in \mathbb{R}^{p\times p}$, and $R = R^T \in \mathbb{R}^{m\times m}$. The corresponding Hamiltonian matrix is defined by

$$H = \begin{bmatrix} A & -G \\ -Q & -A^T \end{bmatrix} = \begin{bmatrix} A & -BR^{-1}B^T \\ -C^T\tilde{Q}C & -A^T \end{bmatrix} \in \mathbb{R}^{2n\times 2n}.$$

The coefficient matrices in (1) often come from linear-quadratic control problems of the form

Minimize

$$J(x_0, u) = \frac{1}{2}\int_0^\infty \Big(y(t)^T\tilde{Q}y(t) + u(t)^T Ru(t)\Big)\, dt \tag{2}$$

subject to the dynamics

$$\dot{x}(t) = Ax(t) + Bu(t), \quad x(0) = x_0, \tag{3}$$

$$y(t) = Cx(t). \tag{4}$$

If, for example, $\tilde{Q} \geq 0$, $R > 0$, (A, B) stabilizable, and (A, C) detectable, then the solution of the optimal control problem (2)–(4) is given by the feedback law

$$u(t) = -R^{-1}B^T X_* x(t)$$

where X_* is the unique stabilizing, positive semidefinite solution of (1) (see e.g. [37, 49]). Most examples in this benchmark collection come from this application, often also referred to as the *deterministic linear-quadratic regulator (LQR) problem.* The only exceptions are Examples 2.5 and 2.9. While the first one represents a type of CAREs arising in H_∞–control, the second exception corresponds to the two CAREs in a *(stochastic) linear-quadratic Gaussian (LQG) optimal control problem.*

One common approach to solve (1) is to compute the stable invariant subspace of the Hamiltonian matrix H, i.e., the subspace corresponding to the eigenvalues of H in the open left half plane (e.g. [13, 34, 35, 37, 49]). If this subspace is spanned by $\begin{bmatrix} U_1 \\ U_2 \end{bmatrix}$ and U_1 is invertible, then $X_* = U_2U_1^{-1}$ is the stabilizing solution of (1).

The examples are grouped in four sections. The first section contains parameter-free examples of fixed dimension, the second parameter-dependent problems of fixed size. Sections 3 and 4 contain examples with scalable size where, in Section 4, the user can also choose one or several parameters.

All presented examples may be generated by the FORTRAN 77 subroutine DCAREX (see Appendix A).

The description of each example contains a table with some of the system properties. This information is summarized in Appendix B. For all parameters needed in the examples there exist default values that are also given in the tables. These default values are chosen in such a way that the collection of examples can be used as a testset for the comparison of methods. The tables contain information for one or more choices of the parameters. Underlined values indicate the default values.

For each example, we provide norms and condition numbers of the stabilizing solution X_* and the Hamiltonian matrix H. A large condition number of H may signal that one can expect problems using the sign function method [17, 24, 45] since the underlying Newton iteration depends on inversion of H. On the other hand, a large condition number may also be due to a large norm of H rather than to ill conditioning with respect to inversion as in Example 4.4.

If no analytical solution is available, we computed approximations by the multishift algorithm [1] and the Schur vector method [34]. If possible, these approximations were refined by Newton's method [32] possibly combined with an exact line search [8, 9] to achieve the highest possible accuracy. We then chose the approximate solution with smallest residual norm to compute the properties of the example. (Note that this is not necessarily the most accurate solution!) Only for Example 4.4, we used the sign function method to compute a first approximation to the solution which was then refined by Newton's method combined with exact line search.

Moreover, we give the minimum distance of the closed-loop eigenvalues, i.e., of the eigenvalues of $A_c := A - BR^{-1}B^T X_*$, to the imaginary axis. This distance is measured by the minimal absolute value of the real parts of all eigenvalues of A_c. Note that the spectrum of the Hamiltonian matrix H corresponding to the CARE (1) consists of the union of the spectra of A_c and $-A_c$. This number is an indicator of the degree of stability of the corresponding closed-loop system and the separation of the eigenvalues of H with respect to the imaginary axis. But note that this indicator may severely overestimate these distances and other measures such as the *stability radius* of A_c [18, 27] or the *separation* of A_c and $-A_c$ [25] may yield much better information.

By $\sigma(A)$ we denote the set of eigenvalues or spectrum of a matrix A. The spectral norm of a matrix is given by

$$\|A\| = \sqrt{\max\{|\lambda| \ : \ \lambda \in \sigma(A^T A)\}}$$

and the given matrix condition numbers are based upon the spectral norm,

$$\kappa(A) = \|A\|\|A^{-1}\|, \quad A \in \mathbb{R}^{l\times l}.$$

Norms and condition numbers were computed by the MATLAB functions `norm` and `cond`.

The "right" condition number of algebraic Riccati equations is still an open problem. It has been studied in several papers. Usually, one refers to the condition number which was introduced by Arnold [3], Arnold and Laub [4], and Byers [16] and the estimates and bounds given by Kenney and Hewer [30]. Let $\breve{A}$, $\breve{G}$, and $\breve{Q}$ be real $n \times n$ matrices "near" A, G, and Q with respect to the 2-norm and in addition, assume G, Q to be symmetric positive semidefinite. ($\breve{A}$, $\breve{G}$, $\breve{Q}$ may be considered as perturbed input data.) Define $\Delta A = \breve{A} - A$, $\Delta G = \breve{G} - G$, and $\Delta Q = \breve{Q} - Q$. Then denoting the stabilizing solution of the CARE (1) by X_*, the CARE condition number presented in [16, 30] is defined by

$$K_{CARE} = \lim_{\delta\to 0} \sup \left\{ \frac{\|\Delta X\|}{\delta\|X_*\|} \ : \ \|\Delta A\| \le \delta\|A\|, \ \|\Delta G\| \le \delta\|G\|, \ \|\Delta Q\| \le \delta\|Q\| \right\}.$$

Let Z_i, $i = 0, 1, 2$, be the solutions of the Lyapunov equations

$$(A - GX_*)^T Z_i + Z_i(A - GX_*) = -X_*^i, \qquad i = 0, 1, 2, \tag{5}$$

and define

$$K_U = \frac{\|Z_0\|\|Q\| + 2\sqrt{\|Z_0\|\|Z_2\|}\|A\| + \|Z_2\|\|G\|}{\|X_*\|}, \tag{6}$$

$$K_L = \frac{\|Z_0\|\|Q\| + 2\|Z_1\|\|A\| + \|Z_2\|\|G\|}{\|X_*\|}. \tag{7}$$

Then one can prove (see [30]) that $\frac{1}{3}K_L \leq K_{CARE} \leq K_U$. If K_L is close to K_U, this provides a very good approximation to K_{CARE}. Since this holds for most of the examples, we only give the upper bound K_U as an approximation for K_{CARE}. (Note that in [30] a more refined lower bound is given which in some cases is closer to K_U.)

In summary, the following values describing the examples are given in the tables:

n	—	the order of the Riccati equation, i.e., $X \in \mathbb{R}^{n\times n}$;
m	—	interpreting the data as data from a linear-quadratic optimal control problem, m denotes the number of inputs of the system;
p	—	interpreting the data as data from a linear-quadratic optimal control problem, p denotes the number of outputs of the system;
parameter	—	default setting for the parameters of the examples;
$\lvert\lambda_{\min}^{re}\rvert$	—	absolute value of the real part of eigenvalue(s) of H closest to the imaginary axis;
$\|X_*\|$	—	2-norm of the stabilizing solution of the CARE;
$\kappa(X_*)$	—	2-norm condition number of the stabilizing solution of the CARE;
$\|H\|$	—	2-norm of the Hamiltonian matrix corresponding to the CARE;
$\kappa(H)$	—	2-norm condition number of the Hamiltonian matrix corresponding to the stabilizing solution of the CARE;
K_U	—	upper bound for the CARE condition number as given in (6).

The given benchmark collection is based on the collection described in [10, 11]. The FORTRAN 77 subroutine has been modified, a couple of bugs have been corrected, and in particular a new example (Example 2.9) has been added.

1 Parameter-free problems of fixed size

Example 1.1 [34, Example 1]

n	m	p	parameter	$\lvert\lambda_{\min}^{re}\rvert$	$\|H\|$	$\kappa(H)$	$\|X_*\|$	$\kappa(X_*)$	K_U
2	1	2		1.0	2.4	5.8	3.0	3.0	5.0

This example can be used for a first verification of any solver for (1) since the solution may be computed by hand. The system matrices are

$$A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad R = 1, \quad Q = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}.$$

The solution is given by

$$X_* = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$$

and the spectrum of the closed-loop matrix

$$(A - BR^{-1}B^T X_*) = \begin{bmatrix} 0 & 1 \\ -1 & -2 \end{bmatrix}$$

is $\{-1, -1\}$.

Example 1.2 [34, Example 2]

n	m	p	parameter	$\lvert\lambda^{re}_{\min}\rvert$	$\lVert H\rVert$	$\kappa(H)$	$\lVert X_*\rVert$	$\kappa(X_*)$	K_U
2	1	2		0.50	16.2	2.6×10^2	31.4	∞	52.6

This is an example of stabilizable-detectable, but uncontrollable-unobservable data. We have the following system matrices:

$$A = \begin{bmatrix} 4 & 3 \\ -\frac{9}{2} & -\frac{7}{2} \end{bmatrix}, \quad B = \begin{bmatrix} 1 \\ -1 \end{bmatrix}, \quad R = 1, \quad Q = \begin{bmatrix} 9 & 6 \\ 6 & 4 \end{bmatrix}$$

with stabilizing solution

$$X_* = \begin{bmatrix} 9(1+\sqrt{2}) & 6(1+\sqrt{2}) \\ 6(1+\sqrt{2}) & 4(1+\sqrt{2}) \end{bmatrix}$$

and closed-loop spectrum $\{-1/2, -\sqrt{2}\}$.

The remaining examples of this chapter consist of some real-world applications. The description of these problems is kept to the minimum necessary information. For the physical, chemical, or engineering background see the given references and the references given therein. Besides their original reference, Examples 1.3–1.5 may be found in [22].

Example 1.3 [7]

n	m	p	parameter	$\lvert\lambda^{re}_{\min}\rvert$	$\lVert H\rVert$	$\kappa(H)$	$\lVert X_*\rVert$	$\kappa(X_*)$	K_U
4	2	4		0.73	7.8	55.8	6.1	2.2×10^2	21.9

Here the system matrices describe a mathematical model of an L-1011 aircraft.

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -1.89 & 0.39 & -5.53 \\ 0 & -0.034 & -2.98 & 2.43 \\ 0.034 & -0.0011 & -0.99 & -0.21 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 0 \\ 0.36 & -1.6 \\ -0.95 & -0.032 \\ 0.03 & 0 \end{bmatrix},$$

$$Q = \begin{bmatrix} 2.313 & 2.727 & 0.688 & 0.023 \\ 2.727 & 4.271 & 1.148 & 0.323 \\ 0.688 & 1.148 & 0.313 & 0.102 \\ 0.023 & 0.323 & 0.102 & 0.083 \end{bmatrix}, \quad R = I_2.$$

In this example, Q has one negative eigenvalue of order $O(10^{-3})$. This may reflect a small perturbation in the input data. The computed stabilizing solution is nevertheless positive definite with eigenvalues greater than one.

Example 1.4 [12]

n	m	p	parameter	$\lvert\lambda_{\min}^{re}\rvert$	$\lVert H\rVert$	$\kappa(H)$	$\lVert X_*\rVert$	$\kappa(X_*)$	K_U
8	2	8		0.10	3.41	305.91	4.75	1.28×10^3	33.58

A mathematical model of a binary distillation column with condenser, reboiler, and nine plates is given by

$$A = \begin{bmatrix} -0.991 & 0.529 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.522 & -1.051 & 0.596 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.522 & -1.118 & 0.596 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.522 & -1.548 & 0.718 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.922 & -1.640 & 0.799 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.922 & -1.721 & 0.901 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.922 & -1.823 & 1.021 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.922 & -1.943 \end{bmatrix},$$

$$B^T = 10^{-3} \times \begin{bmatrix} 3.84 & 4.00 & 37.60 & 3.08 & 2.36 & 2.88 & 3.08 & 3.00 \\ -2.88 & -3.04 & -2.80 & -2.32 & -3.32 & -3.82 & -4.12 & -3.96 \end{bmatrix},$$

$$Q = \begin{bmatrix} 1 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0.1 \\ 0 & 1 & 0 & 0 & 0.1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0.5 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0.5 & 0.1 & 0 & 0 & 0.1 & 0 & 0 & 0 \\ 0 & 0 & 0.5 & 0 & 0 & 0.1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.1 & 0 \\ 0.1 & 0 & 0 & 0 & 0 & 0 & 0 & 0.1 \end{bmatrix}, \qquad R = I_2.$$

Note that Q is indefinite and the computed stabilizing solution is indefinite, too.

Example 1.5 [41]

n	m	p	parameter	$\lvert\lambda_{\min}^{re}\rvert$	$\lVert H\rVert$	$\kappa(H)$	$\lVert X_*\rVert$	$\kappa(X_*)$	K_U
9	3	9		0.34	216.7	3.4×10^3	2.7	1.1×10^3	8.5×10^2

This is the data for a 9th-order continuous state space model of a tubular ammonia reactor. It should be noted that the underlying model includes a disturbance term which is neglected in this context.

$$A = \begin{bmatrix} -4.019 & 5.12 & 0 & 0 & -2.082 & 0 & 0 & 0 & 0.87 \\ -0.346 & 0.986 & 0 & 0 & -2.34 & 0 & 0 & 0 & 0.97 \\ -7.909 & 15.407 & -4.069 & 0 & -6.45 & 0 & 0 & 0 & 2.68 \\ -21.816 & 35.606 & -0.339 & -3.87 & -17.8 & 0 & 0 & 0 & 7.39 \\ -60.196 & 98.188 & -7.907 & 0.34 & -53.008 & 0 & 0 & 0 & 20.4 \\ 0 & 0 & 0 & 0 & 94.0 & -147.2 & 0 & 53.2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 94.0 & -147.2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 12.8 & 0 & -31.6 & 0 \\ 0 & 0 & 0 & 0 & 12.8 & 0 & 0 & 18.8 & -31.6 \end{bmatrix},$$

$$B^T = \begin{bmatrix} 0.010 & 0.003 & 0.009 & 0.024 & 0.068 & 0 & 0 & 0 & 0 \\ -0.011 & -0.021 & -0.059 & -0.162 & -0.445 & 0 & 0 & 0 & 0 \\ -0.151 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

Q and R are chosen as identity matrices of size 9 and 3, respectively.

Example 1.6 [20]

n	m	p	parameter	$\lvert\lambda^{re}_{\min}\rvert$	$\lVert H\rVert$	$\kappa(H)$	$\lVert X_*\rVert$	$\kappa(X_*)$	K_U
30	3	5		0.18	1.4×10^8	1.5×10^{10}	3.6×10^3	∞	3.7×10^9

This control problem for a J–100 jet engine is a special case of a multivariable servomechanism problem. The system state is given by the state of the jet engine denoted by $x^{(1)} \in \mathbb{R}^{16}$, the actuators $x^{(2)} \in \mathbb{R}^8$, and the sensors $x^{(3)} \in \mathbb{R}^6$. There are three actuators in this problem: one for the fuel flow (denoted by $x^{(2,1)} \in \mathbb{R}^2$), one for the nozzle jet area ($x^{(2,2)} \in \mathbb{R}^3$), and one for the inlet guide vane position ($x^{(2,3)} \in \mathbb{R}^3$). The fuel flow x^{ff}, nozzle jet area x^{nja}, and inlet guide vane positions x^{igvp} themselves are given by the first component of the corresponding state variables, i.e.,

$$x^{ff} = x_1^{(2,1)}, \qquad x^{nja} = x_1^{(2,2)}, \qquad x^{igvp} = x_1^{(2,3)}.$$

The dynamics of the system are then given by the following set of equations: The state of the jet engine is described by

$$\dot{x}^{(1)} = A^{(1)}x^{(1)} + A^{(1,2,1)}x^{ff} + A^{(1,2,2)}x^{nja} + A^{(1,2,3)}x^{igvp} + B^{(1)}u^{(1)},$$

where $B^{(1)} = 0$ and

$A^{(1)} =$

$$\begin{bmatrix}
-4.328D{+}00 & 1.714D{-}01 & 5.376D{+}00 & 4.016D{+}02 & -7.246D{+}02 & -1.933D{+}00 & 1.020D{+}00 & -9.820D{-}01 \\
-4.402D{-}01 & -5.643D{+}00 & 1.275D{+}02 & -2.335D{+}02 & -4.343D{+}02 & 2.659D{+}01 & 2.040D{+}00 & -2.592D{+}00 \\
1.038D{+}00 & 6.073D{+}00 & -1.650D{+}02 & -4.483D{+}00 & 1.049D{+}03 & -8.245D{+}01 & -5.314D{+}00 & 5.097D{+}00 \\
5.304D{-}01 & -1.086D{-}01 & 1.313D{+}02 & -5.783D{+}02 & 1.020D{+}02 & -9.240D{+}00 & -1.146D{+}00 & -2.408D{+}00 \\
8.476D{-}03 & -1.563D{-}02 & 5.602D{-}02 & 1.573D{+}00 & -1.005D{+}01 & 1.952D{-}01 & -8.804D{-}03 & -2.110D{-}02 \\
8.350D{-}01 & -1.249D{-}02 & -3.567D{-}02 & -6.074D{-}01 & 3.765D{+}01 & -1.979D{+}01 & -1.813D{-}01 & -2.952D{-}02 \\
6.768D{-}01 & -1.264D{-}02 & -9.683D{-}02 & -3.567D{-}01 & 8.024D{+}01 & -8.239D{-}02 & -2.047D{+}01 & -3.928D{-}02 \\
-9.696D{-}02 & 8.666D{-}01 & 1.687D{+}01 & 1.051D{+}00 & -1.023D{+}02 & 2.966D{+}01 & 5.943D{-}01 & -1.997D{+}01 \\
-8.785D{-}03 & -1.636D{-}02 & 1.847D{-}01 & 2.169D{-}01 & -8.420D{+}00 & 7.003D{-}01 & 5.666D{-}02 & 6.623D{+}00 \\
-1.298D{-}04 & -2.430D{-}04 & 2.718D{-}03 & 3.214D{-}03 & -1.246D{-}01 & 1.037D{-}02 & 8.395D{-}04 & 9.812D{-}02 \\
-1.207D{+}00 & -6.717D{+}00 & 2.626D{+}01 & 1.249D{+}01 & -1.269D{+}03 & 1.030D{+}02 & 7.480D{+}00 & 3.684D{+}01 \\
-2.730D{-}02 & -4.539D{-}01 & -5.272D{+}01 & 1.988D{+}02 & -2.809D{+}01 & 2.243D{+}00 & 1.794D{-}01 & 9.750D{+}00 \\
-1.206D{-}03 & -2.017D{-}02 & -2.343D{+}00 & 8.835D{+}00 & -1.248D{+}00 & 9.975D{-}02 & 8.059D{-}03 & 4.333D{-}01 \\
-1.613D{-}01 & -2.469D{-}01 & -2.405D{+}01 & 2.338D{+}01 & 1.483D{+}02 & 1.638D{+}00 & 1.385D{-}01 & 4.488D{+}00 \\
-1.244D{-}02 & 3.020D{-}02 & -1.198D{-}01 & -4.821D{-}02 & 5.575D{+}00 & -4.525D{-}01 & 1.981D{+}01 & 1.249D{-}01 \\
-1.653D{+}00 & 1.831D{+}00 & -3.822D{+}00 & 1.134D{+}02 & 3.414D{+}02 & -2.734D{+}01 & -2.040D{+}00 & -6.166D{-}01
\end{bmatrix} \cdots$$

$$\cdots \begin{bmatrix}
9.990D{-}01 & 1.521D{+}00 & -4.062D{+}00 & 9.567D{+}00 & 1.008D{+}01 & -6.017D{-}01 & -1.312D{-}01 & 9.602D{-}02 \\
1.132D{+}01 & 1.090D{+}01 & -4.071D{+}00 & -5.739D{-}02 & -6.063D{-}01 & -7.488D{-}02 & -5.936D{-}01 & -9.602D{-}02 \\
-9.389D{-}03 & 1.352D{-}01 & 5.638D{+}00 & 2.246D{-}02 & 1.797D{-}01 & 2.407D{-}02 & 1.100D{+}00 & 2.743D{-}02 \\
-3.081D{+}00 & -4.529D{+}00 & 5.707D{+}00 & -2.346D{-}01 & -2.111D{+}00 & -2.460D{-}01 & -4.686D{-}01 & -3.223D{-}01 \\
2.090D{-}03 & -5.256D{-}02 & -4.077D{-}02 & -9.182D{-}03 & -5.178D{-}02 & 3.425D{-}02 & 4.995D{-}03 & -1.256D{-}02 \\
-1.953D{-}02 & -1.622D{-}01 & -6.439D{-}03 & -2.346D{-}02 & -2.201D{-}01 & -2.514D{-}02 & -3.749D{-}03 & -3.351D{-}02 \\
1.878D{-}02 & -2.129D{-}01 & -9.337D{-}03 & -3.144D{-}02 & -2.919D{-}01 & -3.370D{-}02 & 8.873D{-}02 & -4.458D{-}02 \\
2.253D{-}02 & 1.701D{-}01 & 8.371D{-}03 & 2.645D{-}02 & 2.560D{-}01 & 2.835D{-}02 & -3.749D{-}02 & 3.635D{-}02 \\
-4.999D{+}01 & 6.760D{-}02 & 3.946D{+}01 & 4.991D{-}03 & 8.983D{-}02 & 5.349D{-}03 & 0.000D{+}00 & 1.372D{-}02 \\
-6.666D{-}01 & -6.657D{-}01 & 5.847D{-}01 & 6.654D{-}05 & 1.347D{-}03 & 7.131D{-}05 & 0.000D{+}00 & 2.057D{-}04 \\
2.854D{-}01 & 2.332D{+}00 & -4.765D{+}01 & 3.406D{-}01 & 3.065D{+}00 & 3.624D{-}01 & -4.343D{-}01 & 4.681D{-}01 \\
-9.627D{+}00 & -9.557D{+}00 & 3.848D{+}01 & -5.001D{+}01 & 1.011D{-}01 & 1.203D{-}02 & -4.686D{-}02 & 1.715D{-}02 \\
-4.278D{-}01 & -4.245D{-}01 & 1.710D{+}00 & -2.000D{+}00 & -1.996D{+}00 & 5.349D{-}04 & -1.999D{-}03 & 7.544D{-}04 \\
-4.414D{+}00 & -4.354D{+}00 & 1.766D{+}01 & -3.113D{+}00 & -3.018D{+}00 & -1.977D{+}01 & -4.999D{-}02 & 1.509D{-}02 \\
-1.127D{-}03 & -6.760D{-}03 & 1.835D{-}02 & -9.981D{-}04 & -1.347D{-}02 & -1.070D{-}03 & -2.000D{+}01 & -2.057D{-}03 \\
5.004D{-}01 & -1.437D{-}01 & -2.416D{+}00 & -1.073D{-}01 & -1.078D{+}00 & 3.053D{+}01 & 1.989D{+}01 & -5.016D{+}01
\end{bmatrix},$$

$$A^{(1,2,1)} = \begin{bmatrix} -4.570D{-}02 \\ 1.114D{-}01 \\ 2.153D{-}01 \\ 3.262D{-}01 \\ 9.948D{-}03 \\ 2.728D{-}02 \\ 1.716D{-}02 \\ -7.741D{-}02 \\ 3.855D{-}02 \\ 5.707D{-}04 \\ 5.727D{+}00 \\ 1.392D{-}01 \\ 6.172D{-}03 \\ 6.777D{-}02 \\ 1.880D{-}03 \\ 1.677D{-}01 \end{bmatrix}, \quad A^{(1,2,2)} = \begin{bmatrix} -4.516D{+}02 \\ -5.461D{+}02 \\ 1.362D{+}03 \\ 2.080D{+}02 \\ -9.839D{+}01 \\ 7.162D{+}01 \\ 7.171D{+}01 \\ -1.412D{+}02 \\ -7.710D{+}00 \\ -1.144D{-}01 \\ -1.745D{+}03 \\ -2.430D{+}01 \\ -1.082D{+}00 \\ 1.660D{+}01 \\ 9.147D{+}00 \\ 4.358D{+}02 \end{bmatrix}, \quad A^{(1,2,3)} = \begin{bmatrix} -1.058D{+}02 \\ -6.575D{+}00 \\ 1.346D{+}01 \\ -2.888D{+}00 \\ 5.069D{-}01 \\ 9.608D{+}00 \\ 8.571D{+}00 \\ -8.215D{-}01 \\ -4.371D{-}02 \\ -6.359D{-}04 \\ -8.940D{+}00 \\ -2.736D{-}01 \\ -1.183D{-}02 \\ 3.980D{-}01 \\ -8.241D{-}01 \\ -5.994D{+}01 \end{bmatrix}.$$

The actuator for the fuel flow is defined by

$$\begin{aligned} \dot{x}^{(2,1)} &= A^{(2,1)}x^{(2,1)} + B^{(2,1)}u^{(2,1)} \\ &= \begin{bmatrix} 0 & 1 \\ -500 & -60 \end{bmatrix} x^{(2,1)} + \begin{bmatrix} 0 \\ 500 \end{bmatrix} u^{(2,1)}, \end{aligned}$$

the nozzle jet area actuator is given by

$$\begin{aligned} \dot{x}^{(2,2)} &= A^{(2,2)}x^{(2,2)} + B^{(2,2)}u^{(2,2)} \\ &= \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -3600 & -708 & -106.72 \end{bmatrix} x^{(2,2)} + \begin{bmatrix} 0 \\ 0 \\ 3600 \end{bmatrix} u^{(2,2)}, \end{aligned}$$

and the inlet guide vane position actuator is described by

$$\begin{aligned} \dot{x}^{(2,3)} &= A^{(2,3)}x^{(2,3)} + B^{(2,3)}u^{(2,3)} \\ &= \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -12000 & -5240 & -150 \end{bmatrix} x^{(2,3)} + \begin{bmatrix} 0 \\ 0 \\ 12000 \end{bmatrix} u^{(2,3)}. \end{aligned}$$

Since the actuator states are originally given as third-order differential equations, the first entry of $x^{(2,i)}$, $i = 1, 2, 3$, in the first-order model equations above represents the state of the actuators.

Finally, the sensor state is given by

$$\begin{aligned} \dot{x}^{(3)} &= A^{(3)}x^{(3)} + A^{(3,1)}x^{(1)} + B^{(3)}u^{(3)} \\ &= \begin{bmatrix} -33.3 & 0 & 0 & 0 & 0 & 0 \\ 0 & -20 & 0 & 0 & 0 & 0 \\ 0 & 0 & -20 & 0 & 0 & 0 \\ 0 & 0 & 0 & -20 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & -0.306 & -1.86 \end{bmatrix} x^{(3)} + \begin{bmatrix} 33.3x_1^{(1)} \\ 20x_2^{(1)} \\ 20x_3^{(1)} \\ 20x_5^{(1)} \\ 0.645(x_{12}^{(1)} + x_{13}^{(1)}) \\ -0.894(x_{12}^{(1)} + x_{13}^{(1)}) \end{bmatrix} + 0 \cdot u^{(3)}. \end{aligned}$$

We can thus write the above equations in the standard form $\dot{x} = Ax + Bu$ with

$$A = \begin{bmatrix} A^{(1)} & \begin{bmatrix} A^{(1,2,1)} & 0 \end{bmatrix} & \begin{bmatrix} A^{(1,2,2)} & 0 & 0 \end{bmatrix} & \begin{bmatrix} A^{(1,2,3)} & 0 & 0 \end{bmatrix} & 0 \\ 0 & A^{(2,1)} & & & 0 \\ 0 & 0 & A^{(2,2)} & & 0 \\ 0 & 0 & 0 & A^{(2,3)} & 0 \\ A^{(3,1)} & 0 & 0 & 0 & A^{(3)} \end{bmatrix},$$

$$B = \begin{bmatrix} 0 & 0 & 0 \\ \vdots & \vdots & \vdots \\ 0 & 0 & 0 \\ B^{(2,1)} & 0 & 0 \\ 0 & B^{(2,2)} & 0 \\ 0 & 0 & B^{(2,3)} \\ 0 & 0 & 0 \\ \vdots & \vdots & \vdots \\ 0 & 0 & 0 \end{bmatrix}.$$

The output of the system is given by

$$y = Cx$$

$$= \begin{bmatrix} 4.865D{-}01 & 1.383D{-}02 & 0.000D{+}00 & 7.418D{-}05 & 1.538D{-}05 \\ -6.741D{-}01 & 2.789D{-}06 & 0.000D{+}00 & 5.496D{-}06 & 1.201D{-}04 \\ 5.392D{+}00 & 0.000D{+}00 & 0.000D{+}00 & 4.790D{-}06 & -2.579D{-}03 \\ 9.542D{+}01 & 0.000D{+}00 & 0.000D{+}00 & 1.478D{-}04 & -1.609D{-}04 \\ 2.403D{+}01 & -1.081D{-}02 & 0.000D{+}00 & -1.504D{-}02 & 1.618D{-}02 \\ 1.052D{+}01 & -5.545D{-}05 & 0.000D{+}00 & -6.503D{-}05 & -1.071D{-}03 \\ 8.190D{-}01 & 4.722D{-}05 & 0.000D{+}00 & 8.820D{-}05 & -9.561D{-}05 \\ -4.492D{-}01 & 0.000D{+}00 & 0.000D{+}00 & 4.999D{-}06 & -5.503D{-}06 \\ 5.195D{-}01 & 0.000D{+}00 & 0.000D{+}00 & 3.434D{-}06 & -3.732D{-}06 \\ 8.437D{-}01 & 0.000D{+}00 & 0.000D{+}00 & 2.727D{-}05 & -2.996D{-}05 \\ -1.863D{+}00 & 0.000D{+}00 & 1.000D{+}00 & 1.128D{-}06 & -1.234D{-}06 \\ 5.709D{-}02 & 0.000D{+}00 & 0.000D{+}00 & 4.002D{-}06 & -4.380D{-}06 \\ 4.815D{-}01 & 0.000D{+}00 & 0.000D{+}00 & 3.673D{-}05 & -4.024D{-}05 \\ 3.428D{+}00 & 0.000D{+}00 & 0.000D{+}00 & 4.290D{-}06 & -4.721D{-}06 \\ 2.161D{+}00 & 0.000D{+}00 & 0.000D{+}00 & -4.958D{-}06 & 5.324D{-}06 \\ 7.681D{-}02 & 0.000D{+}00 & 0.000D{+}00 & 5.609D{-}06 & -6.103D{-}06 \\ -6.777D{-}02 & 1.282D{-}04 & 0.000D{+}00 & 1.030D{-}06 & 8.109D{-}06 \\ 0.000D{+}00 & 0.000D{+}00 & 0.000D{+}00 & 0.000D{+}00 & 0.000D{+}00 \\ -4.205D{+}02 & 3.353D{-}01 & 0.000D{+}00 & -1.193D{-}02 & 2.328D{-}02 \\ 0.000D{+}00 & 0.000D{+}00 & 0.000D{+}00 & 0.000D{+}00 & 0.000D{+}00 \\ 0.000D{+}00 & 0.000D{+}00 & 0.000D{+}00 & 0.000D{+}00 & 0.000D{+}00 \\ 3.297D{+}01 & 6.804D{-}01 & 0.000D{+}00 & -5.806D{-}03 & 1.178D{-}04 \\ 0.000D{+}00 & 0.000D{+}00 & 0.000D{+}00 & 0.000D{+}00 & 0.000D{+}00 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0.000D{+}00 & \ldots & \ldots & \ldots & 0.000D{+}00 \end{bmatrix}^T \begin{bmatrix} x^{(1)} \\ x^{(2,1)} \\ x^{(2,2)} \\ x^{(2,3)} \\ x^{(3)} \end{bmatrix}.$$

R and $\tilde{Q}$ are chosen as identities of appropriate size such that $G = BB^T$, $Q = C^TC$.

The data of this example differ by 10 orders of magnitude and the norm and condition number of H are quite large. The eigenvalues of H vary in magnitude from $O(10^{-1})$ to $O(10^2)$. As far as the eigenvalue distribution is concerned, the only source from which numerical problems may arise are triple eigenvalues at $\pm\lambda = \pm 20.0$.

2 Parameter-dependent problems of fixed size

Example 2.1 [4, Example 1]

n	m	p	parameter	$\lvert\lambda^{re}_{\min}\rvert$	$\lVert H\rVert$	$\kappa(H)$	$\lVert X_*\rVert$	$\kappa(X_*)$	K_U
2	1	1	$\varepsilon = 1$	1.4	2.95	2.54	2.45	12.91	2.57
			$\underline{\varepsilon = 10^{-6}}$	1.0	3.0	5.2	2.0×10^{12}	8.0×10^{12}	3.0

Consider the system defined by

$$\begin{aligned} A &= \begin{bmatrix} 1 & 0 \\ 0 & -2 \end{bmatrix}, \qquad B = \begin{bmatrix} \varepsilon \\ 0 \end{bmatrix}, \\ R &= 1, \qquad C = \begin{bmatrix} 1 & 1 \end{bmatrix}, \qquad \tilde{Q} = 1. \end{aligned}$$

The exact solution of the Riccati equation is

$$X_* = \begin{bmatrix} \dfrac{1+\sqrt{1+\varepsilon^2}}{\varepsilon^2} & \dfrac{1}{2+\sqrt{1+\varepsilon^2}} \\ \dfrac{1}{2+\sqrt{1+\varepsilon^2}} & \dfrac{1}{4}\left(1 - \dfrac{\varepsilon^2}{\left(2+\sqrt{1+\varepsilon^2}\right)^2}\right) \end{bmatrix}.$$

As $\varepsilon \to 0$, the matrix pair (A, B) becomes unstabilizable and x_{11} tends to ∞. Numerical methods for computing the stable invariant subspace of H yield condition estimates for U_1 of order $1/\varepsilon^2$ in accordance to $\kappa(X_*) \approx 8/\varepsilon^2$.

Example 2.2 [4, Example 3]

n	m	p	parameter	$\lvert\lambda^{re}_{\min}\rvert$	$\lVert H\rVert$	$\kappa(H)$	$\lVert X_*\rVert$	$\kappa(X_*)$	K_U
2	2	1	$\varepsilon = 1.0$	1.00	1.0×10^4	6.8×10^7	9.9×10^3	4.2×10^3	54.4
			$\underline{\varepsilon = 10^{-8}}$	0.70	1.0×10^6	1.5×10^8	9.3×10^3	9.4×10^6	6.7×10^9

This is an example with increasingly ill-conditioned *control weighting matrix* R.

$$\begin{aligned} A &= \begin{bmatrix} -0.1 & 0 \\ 0 & -0.02 \end{bmatrix}, \qquad B = \begin{bmatrix} 0.1 & 0 \\ 0.001 & 0.01 \end{bmatrix}, \\ R &= \begin{bmatrix} 1+\varepsilon & 1 \\ 1 & 1 \end{bmatrix}, \qquad C = \begin{bmatrix} 10 & 100 \end{bmatrix}, \qquad \tilde{Q} = 1. \end{aligned}$$

If $\varepsilon < 1$, then $\kappa(R) = O(1/\varepsilon)$ and as $\varepsilon \to 0$, the elements of $G = BR^{-1}B^T$ become increasingly large.

Example 2.3 [31, Example 2]

n	m	p	parameter	$\lvert\lambda^{re}_{\min}\rvert$	$\lVert H\rVert$	$\kappa(H)$	$\lVert X_*\rVert$	$\kappa(X_*)$	K_U
2	1	2	$\varepsilon = 1$	0.86	1.6	2.6	2.7	3.7	4.2
			$\underline{\varepsilon = 10^6}$	7.1×10^2	1.0×10^6	1.0×10^6	1.4×10^3	2.0×10^6	8.7×10^5
			$\varepsilon = 10^{-6}$	1.0×10^{-6}	1.0	1.0×10^{12}	1.0×10^6	1.0×10^6	5.0×10^{11}

In this example, the matrix A contains a parameter ε.

$$A = \begin{bmatrix} 0 & \varepsilon \\ 0 & 0 \end{bmatrix}, \qquad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \qquad R = 1, \qquad Q = I_2.$$

The exact solution, which is stabilizing for $\varepsilon > 0$, is given by

$$X_* = \begin{bmatrix} \dfrac{\sqrt{1+2\varepsilon}}{\varepsilon} & 1 \\ 1 & \sqrt{1+2\varepsilon} \end{bmatrix}.$$

As ε grows, $\|X_*\|$ increases like $\sqrt{\varepsilon}$ and the Riccati equation becomes ill conditioned in terms of K_{CARE} and K_U. Closed-loop eigenvalues are $\left\{-\frac{1}{2}\left(\sqrt{1+2\varepsilon} \pm \sqrt{1-2\varepsilon}\right)\right\}$ and hence one eigenvalue approaches 0 as $\varepsilon \to 0$. In this case, $\kappa(X_*) = O(1/\varepsilon)$ and $\kappa(H) = O(1/\varepsilon^2)$.

Hence, this example may be used to test the ability of a CARE solver to deal with bad scaling due to the A–matrix and mild ill conditioning ($\varepsilon \to +\infty$), with closed-loop eigenvalues close to the imaginary axis as well as very ill-conditioned Hamiltonian matrices ($\varepsilon \to 0$).

Example 2.4 [6]

n	m	p	parameter	$\lvert\lambda^{re}_{\min}\rvert$	$\lVert H\rVert$	$\kappa(H)$	$\lVert X_*\rVert$	$\kappa(X_*)$	K_U
2	2	2	$\varepsilon = 1$	1.4	3.2	2.2	6.2	2.6	2.5
			$\underline{\varepsilon = 10^{-7}}$	1.4×10^{-7}	2.6	1.3×10^{14}	4.0	1.7×10^7	3.8×10^3

Here, the system matrices are

$$A = \begin{bmatrix} \varepsilon + 1 & 1 \\ 1 & \varepsilon + 1 \end{bmatrix}, \qquad G = I_2, \qquad Q = \begin{bmatrix} \varepsilon^2 & 0 \\ 0 & \varepsilon^2 \end{bmatrix}.$$

The exact stabilizing solution X_* of (1) is given by (note the correction in $x_{12} = x_{21}$ from [6])

$$\begin{aligned} x_{11} &= x_{22} = \frac{1}{2}\left(2(\varepsilon+1) + \sqrt{2(\varepsilon+1)^2 + 2} + \sqrt{2}\varepsilon\right), \\ x_{12} &= x_{21} = \frac{x_{11}}{x_{11} - (\varepsilon + 1)}. \end{aligned}$$

As $\varepsilon \to 0$, then H becomes increasingly ill conditioned, i.e., $\kappa(H) = O(1/\varepsilon^2)$, whereas $\kappa(X_*)$ behaves like $1/\varepsilon$. Note that for $\varepsilon = 10^{-7}$, then $K_L = 2.0$ which is three orders of magnitude smaller than K_U. This shows that K_U, K_L may sometimes be far apart and thus, K_U *may* overestimate K_{CARE} by orders of magnitude.

Example 2.5 [28]

n	m	p	parameter	$\lvert\lambda^{re}_{\min}\rvert$	$\lVert H\rVert$	$\kappa(H)$	$\lVert X_*\rVert$	$\kappa(X_*)$	K_U
2	1	2	$\varepsilon = 1$	1.0	10.6	24.2	2.6	6.9	8.1
			$\underline{\varepsilon = 0}$	0.0	15.4	58.4	2.6	6.9	∞

Let

$$A = \begin{bmatrix} 3-\varepsilon & 1 \\ 4 & 2-\varepsilon \end{bmatrix}, \qquad B = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \qquad R = 1, \qquad Q = \begin{bmatrix} 4\varepsilon - 11 & 2\varepsilon - 5 \\ 2\varepsilon - 5 & 2\varepsilon - 2 \end{bmatrix}.$$

This example represents a type of algebraic Riccati equation arising in H_∞–control problems as presented, e.g., in [50]. The matrix

$$X_* = \begin{bmatrix} 2 & 1 \\ 1 & 1 \end{bmatrix}$$

solves (1) for arbitrary ε. This is the stabilizing solution for $\varepsilon > 0$ and for $\varepsilon = 0$ it is still the solution obtained by an H–invariant *Lagrangian* subspace, i.e., the required solution in the sense of H_∞–control. The spectrum of H is $\{\pm\varepsilon \pm j\}$. Hence the closed-loop eigenvalues approach the imaginary axis as $\varepsilon \to 0$.

Note that $K_U = \infty$ for $\varepsilon = 0$ does not represent an ill-conditioned Riccati equation. In this case, the condition number K_{CARE} as given in [30] is not defined.

Example 2.6 [43]

n	m	p	parameter	$\lvert\lambda^{re}_{\min}\rvert$	$\lVert H\rVert$	$\kappa(H)$	$\lVert X_*\rVert$	$\kappa(X_*)$	K_U
3	3	3	$\varepsilon = 1$	1.4	3.2	2.2	6.2	2.6	2.5
			$\underline{\varepsilon = 10^6}$	1.0×10^6	3.5×10^6	3.5	6.0×10^{12}	3.0	2.7

This example is constructed as follows. Let

$$V = I - \frac{2}{3} v v^T, \qquad v^T = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$$

and

$$A_0 = \varepsilon\, \mathrm{diag}(1,\ 2,\ 3), \qquad Q_0 = \mathrm{diag}(\frac{1}{\varepsilon},\ 1,\ \varepsilon).$$

Then

$$A = V A_0 V, \qquad G = \frac{1}{\varepsilon} I_3, \qquad Q = V Q_0 V.$$

Note that a factorization $Q = C^T \tilde{Q} C$ can be obtained by setting $C := V$ and $\tilde{Q} := Q_0$. This is used in both the FORTRAN 77 and MATLAB implementations if a factored form is required.

As solution we get

$$X_* = V\, \mathrm{diag}(x_1,\ x_2,\ x_3)\, V$$

where

$$\begin{aligned} x_1 &= \varepsilon^2 + \sqrt{\varepsilon^4 + 1}, \\ x_2 &= 2\varepsilon^2 + \sqrt{4\varepsilon^4 + \varepsilon}, \\ x_3 &= 3\varepsilon^2 + \sqrt{9\varepsilon^4 + \varepsilon^2}. \end{aligned}$$

For growing ε, the corresponding Hamiltonian matrix becomes more and more badly scaled which leads to a significant loss of accuracy in all CARE solvers based on eigenvalue methods. This demonstrates the need to use an appropriate scaling as proposed in [31, 40].

Example 2.7 [19]

n	m	p	parameter	$\lvert\lambda^{re}_{\min}\rvert$	$\lVert H\rVert$	$\kappa(H)$	$\lVert X_*\rVert$	$\kappa(X_*)$	K_U
4	1	2	$\varepsilon = 1$	0.19	1.6	1.2×10^2	22.3	1.0×10^3	93.0
			$\underline{\varepsilon = 10^{-6}}$	0.25	1.0×10^{12}	5.7×10^{13}	13.2	9.1×10^8	4.1×10^{13}

The data of this example describes a magnetic tape control problem.

$$A = \begin{bmatrix} 0 & 0.4 & 0 & 0 \\ 0 & 0 & 0.345 & 0 \\ 0 & -0.524/\varepsilon & -0.465/\varepsilon & 0.262/\varepsilon \\ 0 & 0 & 0 & -1/\varepsilon \end{bmatrix}, \qquad B = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1/\varepsilon \end{bmatrix},$$
$$Q = \mathrm{diag}(1,\ 0,\ 1,\ 0), \qquad R = 1.$$

A full rank factorization C^TC of Q yields

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}.$$

As $\varepsilon \to 0$, the pair (A, B) becomes unstabilizable and all condition numbers increase. The Hamiltonian matrix H becomes very badly scaled.

Example 2.8 [4, Example 2]

n	m	p	parameter	$\lvert\lambda^{re}_{\min}\rvert$	$\lVert H\rVert$	$\kappa(H)$	$\lVert X_*\rVert$	$\kappa(X_*)$	K_U
4	1	1	$\varepsilon = 1$	0.52	4.5	9.9	11.8	1.4×10^2	36.4
			$\underline{\varepsilon = 10^{-6}}$	5.0×10^{-13}	4.2	17.9	1.0	1.0	1.0×10^{13}

Here, we have the following system matrices:

$$A = \begin{bmatrix} -\varepsilon & 1 & 0 & 0 \\ -1 & -\varepsilon & 0 & 0 \\ 0 & 0 & \varepsilon & 1 \\ 0 & 0 & -1 & \varepsilon \end{bmatrix}, \qquad B = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}, \qquad R = 1, \qquad C = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}^T, \qquad \tilde{Q} = 1.$$

As $\varepsilon \to 0$, a pair of complex conjugate eigenvalues of the Hamiltonian matrix H approaches the imaginary axis, (A, B) gets close to an unstabilizable system, and the CARE becomes fairly ill conditioned as measured by K_U.

Example 2.9 [21, IFAC Benchmark Problem 90-06]

n	m	p	parameter	$\lvert\lambda^{re}_{\min}\rvert$	$\lVert H\rVert$	$\kappa(H)$	$\lVert X_*\rVert$	$\kappa(X_*)$	K_U
55	2	10	$\underline{\#\ 1}$	2.9×10^{-2}	4.4×10^{10}	1.3×10^{16}	1.2×10^3	∞	5.3×10^{12}
55	2	3	# 2	4.2×10^{-2}	4.4×10^8	6.4×10^{13}	2.0×10^5	1.5×10^{12}	2.1×10^{11}

The two CAREs # 1 and # 2 in this example correspond to the LQR and filter CAREs of a linear-quadratic Gaussian optimal control problem. The linear system is defined by

$$\begin{aligned} \dot{x}(t) &= Ax(t) + B_1u(t) + B_2w(t), \\ z(t) &= C_1x(t), \\ y(t) &= C_2x(t) + v(t), \end{aligned}$$

where $x \in \mathbb{R}^n, y \in \mathbb{R}^{p_2}, z \in \mathbb{R}^{p_1}, u \in \mathbb{R}^{m_1}, v \in \mathbb{R}^{p_2}, w \in \mathbb{R}^{m_2}$ are the state vector, the measured or sensor outputs, the performance outputs, the control inputs, the sensor noise, and the process noise, respectively. The error signals v and w are assumed to be white noise vectors with covariance (or spectral density) matrices V and W, respectively. The design cost function is given as

$$J(u) = \frac{1}{2}\int_0^\infty \left(z(t)^T \tilde{Q} z(t) + u(t)^T R u(t)\right)\, dt, \tag{8}$$

yielding the LQR CARE

$$0 = C_1^T \tilde{Q} C_1 + A^T X + XA - XB_1 R^{-1} B_1^T X, \tag{9}$$

which is exactly the CARE of the deterministic LQR problem as considered in the introduction. The state observer for the LQG design is obtained as a Kalman filter, defined via the solution of the filter CARE

$$0 = B_2 W B_2^T + AY + YA^T - YC_2^T V^{-1} C_2 Y. \tag{10}$$

In both cases, the required solution is stabilizing, i.e., for the solutions we have that $A - B_1 R^{-1} B_1^T X_*$ and $A - Y_* C_2^T V^{-1} C_2$ are stable. For details on LQG control and the related CAREs see, e.g., [29].

The particular example of an LQG design with the data used here is given in [21, Problem # 90-06] where flutter control and gust load alleviation for a modified Boeing B-767 airplane at flutter condition are considered. The plant is described by an aeroelastic model of the airplane. Due to the state dimension $n = 55$ we refrain from re-producing the corresponding matrix A; the data can be found in [21]. For the same reasons we do not list the matrices $B_1 \in \mathbb{R}^{55\times 2}$, $B_2 \in \mathbb{R}^{55\times 3}$, $C_1 \in \mathbb{R}^{10\times 55}$, and $C_2 \in \mathbb{R}^{2\times 55}$. The two sensor outputs are the aircraft pitch rate and the wing-tip acceleration. There are ten performance outputs of which only the bending moment and the torsion at the inboard wing station and the torsion at the outboard wing station are penalized in the design cost function. That is, the weighting matrix $\tilde{Q}$ in (8) is a diagonal matrix with the only nonzero entries $\tilde{q}_{2,2} = 3.76 \times 10^{-14}$, $\tilde{q}_{3,3} = 1.20 \times 10^{-13}$, and $\tilde{q}_{6,6} = 2.45 \times 10^{-12}$. The control weighting matrix R and the covariance matrices involved in the Kalman filter design are

$$\begin{aligned} R &= \begin{bmatrix} 364.7 & 0 \\ 0 & 14.59 \end{bmatrix}, \qquad V = \begin{bmatrix} 6.85 \times 10^{-6} & 0 \\ 0 & 373.0 \end{bmatrix}, \\ W &= \begin{bmatrix} 2.8224 \times 10^4 & 0 & 0 \\ 0 & 2.742 \times 10^{-5} & 0 \\ 0 & 0 & 6.854 \times 10^{-4} \end{bmatrix}. \end{aligned}$$

As default we use the LQR CARE (# 1). In order to obtain the data related to the filter CARE (10), one has to set the DCAREX input parameter `IPAR(1)` to '2'; see Appendix A. That is, as default the output matrices $(A, B, C, R, \tilde{Q})$ of DCAREX are $(A, B_1, C_1, R, \tilde{Q})$ for the LQR CARE (subexample # 1) while for the filter CARE, they are given by $(A^T, C_2^T, B_2^T, V, W)$ (subexample # 2).

Both CAREs are relatively hard to solve. The closed-loop spectrum is relatively bad separated from the imaginary axis. The norms and condition numbers of the Hamiltonian matrices are large. In particular, the condition numbers of both CAREs are pretty high. The upper and lower bounds for the condition numbers are relatively tight here. For the LQR equation, $K_L = 5.2 \times 10^{12}$ while for the filter equation, $K_L = 1.7 \times 10^{11}$.

3 Examples of scalable size without parameters

Example 3.1 [34, Example 4], [5]

n	m	p	parameter	$\lvert\lambda^{re}_{\min}\rvert$	$\|H\|$	$\kappa(H)$	$\|X_*\|$	$\kappa(X_*)$	K_U
$2N-1$	N	$N-1$	$N=5$	1.0	10.0	41.0	12.2	∞	11.1
			$\underline{N=20}$	0.66	10.0	4.3×10^2	28.8	∞	50.9

The matrices presented here describe a mathematical model of position and velocity control for a string of high-speed vehicles. (This problem is also known as "smart highway" or "intelligent highway".) If N vehicles are to be controlled, the size of the system matrices will be $n = 2N-1$.

$$A = \begin{bmatrix} A_{11} & A_{12} & 0 & \dots & & 0 \\ 0 & A_{22} & A_{23} & 0 & \dots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ & & 0 & A_{N-2,N-2} & A_{N-2,N-1} & 0 \\ & & & 0 & A_{N-1,N-1} & \begin{bmatrix} 0 \\ -1 \end{bmatrix} \\ 0 & \dots & & & 0 \;\; 0 & -1 \end{bmatrix} \in \mathbb{R}^{(2N-1)\times(2N-1)},$$

where

$$\begin{aligned} A_{k,k} &= \begin{bmatrix} -1 & 0 \\ 1 & 0 \end{bmatrix} \qquad \text{for} \quad 1 \le k \le N-1, \\ A_{k,k+1} &= \begin{bmatrix} 0 & 0 \\ -1 & 0 \end{bmatrix} \qquad \text{for} \quad 1 \le k \le N-2, \end{aligned}$$

and

$$\begin{aligned} G &= \operatorname{diag}(1,\ 0,\ 1,\ 0,\ \dots, 1,\ 0,\ 1), \\ Q &= \operatorname{diag}(0,\ 10,\ 0,\ 10, \dots, 0,\ 10,\ 0). \end{aligned}$$

Full rank factorizations of G and Q are $G = BB^T$, $Q = 10C^TC = C^T\tilde{Q}C$, where

$$\begin{aligned} B &= \begin{bmatrix} 1 & 0 & \dots & \dots & 0 \\ 0 & 0 & & & \vdots \\ 0 & 1 & & & \\ 0 & 0 & & & \\ \vdots & \vdots & & & \\ & & & & 0 \\ 0 & 0 & \dots & 0 & 1 \end{bmatrix} \in \mathbb{R}^{(2N-1)\times N}, \\ C &= \begin{bmatrix} 0 & 1 & 0 & & & \dots & 0 \\ 0 & 0 & 0 & 1 & 0 & \dots & 0 \\ \vdots & & & & & & \vdots \\ 0 & \dots & & \dots & 0 & 1 & 0 \end{bmatrix} \in \mathbb{R}^{(N-1)\times(2N-1)}, \\ \tilde{Q} &= 10 I_{N-1}. \end{aligned}$$

The stabilizing solution is singular ($\operatorname{rank}(X_*) = n-1$). The system does not have any particular bad properties for growing n. All condition numbers only grow very slowly. The closed-loop eigenvalues are all of magnitude $O(1)$. Hence, this example is especially well suited for testing how an algorithm behaves when the dimension of the problem increases.

Example 3.2 [34, Example 5]

n	m	p	parameter	$\lvert\lambda^{re}_{\min}\rvert$	$\lVert H\rVert$	$\kappa(H)$	$\lVert X_*\rVert$	$\kappa(X_*)$	K_U
n	n	n	$n = 8$	1.0	4.1	4.1	1.0	8.1	5.0
			$\underline{n = 64}$	1.0	4.1	4.1	1.0	8.1	5.0

In this example, all system matrices and the solution of (1) are circulant.

$$A = \begin{bmatrix} -2 & 1 & 0 & & \dots & 0 & 1 \\ 1 & -2 & 1 & 0 & \dots & & 0 \\ 0 & 1 & -2 & 1 & 0 & \dots & 0 \\ \vdots & & \ddots & \ddots & \ddots & & \vdots \\ 0 & & & & & & 1 \\ 1 & 0 & \dots & & 0 & 1 & -2 \end{bmatrix}, \qquad G = Q = I_n.$$

Most eigenvalues of the Hamiltonian matrix have multiplicity 2. For invariant subspace methods that use deflation techniques (e.g., Hamiltonian SR [14, 15, 47], multishift QR [1, 44]), this may cause a lot of deflation steps and hence may slow down convergence. Growth of the problem size n does not influence norms and condition numbers. All the closed-loop modes λ are real and of magnitude $O(1)$. Therefore, this example is perfectly suited to test the behavior of algorithms for growing problem size.

The CARE may be solved using an inverse discrete Fourier transformation and the theory of circulant matrices. The stabilizing solution is the circulant matrix

$$X_* = \begin{bmatrix} x_0 & x_{n-1} & x_{n-2} & \dots & x_1 \\ x_1 & x_0 & x_{n-1} & \dots & x_2 \\ x_2 & x_1 & x_0 & & \\ \vdots & \vdots & & \ddots & \vdots \\ x_{n-1} & x_{n-2} & \dots & & x_0 \end{bmatrix},$$

where for $i = 0, \dots, n-1$,

$$x_i = \frac{1}{n} \sum_{k=0}^{n-1} \left\{ -2 + 2\cos\left(\frac{2\pi k}{n}\right) + \sqrt{5 - 8\cos\left(\frac{2\pi k}{n}\right) + 4\cos^2\left(\frac{2\pi k}{n}\right)} \right\} \omega_n^{ik} \tag{11}$$

and ω_n^i is an nth root of unity. Note that the coefficient of the second term of the radicand should be 8 instead of 4 as in [34]. Since the imaginary part of the sum in (11) adds to 0, ω_n^{ik} may be replaced by $\cos\left(\frac{2\pi ki}{n}\right)$ for keeping computations real.

4 Parameter-dependent examples of scalable size

Example 4.1 [34, Example 6]

n	m	p	parameter	$\lvert\lambda^{re}_{\min}\rvert$	$\lVert H\rVert$	$\kappa(H)$	$\lVert X_*\rVert$	$\kappa(X_*)$	K_U
n	1	1	$n = 21,\ q = r = 1.0$	7.5×10^{-2}	1.0	1.0	2.4×10^{9}	∞	1.3×10^{9}
			$n = 21,\ q = r = 100.0$	1.6×10^{-2}	1.0×10^{2}	1.0×10^{4}	2.4×10^{11}	∞	1.3×10^{9}

This example describes a system of n integrators connected in series and a feedback controller is supposed to be applied to the nth system. (For more details about the physical background see [34].)

$$A = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ & & & & 0 \\ 0 & & & 0 & 1 \\ 0 & \dots & & 0 & 0 \end{bmatrix}, \qquad B = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix}, \qquad R = r, \qquad C = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}^T, \qquad \tilde{Q} = q.$$

The eigenvalues of the Hamiltonian matrix are the roots of

$$\lambda^{2n} + (-1)^n qr = 0.$$

It is known that $x_{1n} = \sqrt{qr}$ (note the correction from [34]). Therefore, the relative error in x_{1n}, i.e., $\dfrac{|x_{1n} - \sqrt{qr}|}{\sqrt{qr}}$, may be used as an indicator of the accuracy of the results. The difficulty in this example lies in the fact that U_1 becomes extremely ill conditioned with respect to inversion as n increases and the elements of X_* become very large in magnitude. Observe that the condition number of U_1 for the second parameter combination is two orders of magnitude greater than for the first combination whereas K_U remains constant. This reflects the fact that both values may (or may not) signal some kind of ill conditioning of the CARE.

Example 4.2 [46]

n	m	p	parameter	$\lvert\lambda^{re}_{\min}\rvert$	$\lVert H\rVert$	$\kappa(H)$	$\lVert X_*\rVert$	$\kappa(X_*)$	K_U
n	1	1	$n = 20,$ $a = 0.05,\ b = c = 0.1,$ $[\beta_1,\ \beta_2] = [0.1,\ 0.5],$ $[\gamma_1,\ \gamma_2] = [0.1,\ 0.5].$	0.49	2.6×10^{2}	5.5×10^{2}	1.0×10^{-4}	∞	5.0×10^{2}
			$n = 100,$ $a = 0.01,\ b = c = 1.0,$ $[\beta_1,\ \beta_2] = [0.2,\ 0.3],$ $[\gamma_1,\ \gamma_2] = [0.2,\ 0.3].$	0.1	1.2×10^{3}	1.4×10^{5}	7.1×10^{-4}	∞	1.0×10^{4}

The data of this example come from a linear-quadratic control problem of one-dimensional heat flow. This problem is described in terms of infinite-dimensional operators on a Hilbert space. Using a standard finite element approach based on linear B–splines, a finite-dimensional approximation to the problem may be obtained by the solution of algebraic Riccati equations (1). If N denotes the approximation index, then with this approach we obtain a system of order $n = N - 1$. The data are constructed as follows.

The linear B–splines define the tridiagonal *Gram* matrix

$$M_N = \frac{1}{6N}\begin{bmatrix} 4 & 1 & 0 & & \dots & 0 \\ 1 & 4 & 1 & & & \\ & \ddots & \ddots & \ddots & & \vdots \\ & & & 1 & 4 & 1 \\ 0 & \dots & & & 1 & 4 \end{bmatrix}.$$

Then the system matrices are given by

$$A = M_N^{-1}K_N, \quad B = M_N^{-1}b_N, \quad R = 1, \quad C = c_N^T, \quad \tilde{Q} = 1,$$

where the *stiffness* matrix $K_N \in \mathbb{R}^{n\times n}$ is defined as

$$K_N = -aN\begin{bmatrix} 2 & -1 & 0 & & \dots & 0 \\ -1 & 2 & -1 & & & \\ & \ddots & \ddots & \ddots & & \vdots \\ & & & -1 & 2 & -1 \\ 0 & \dots & & & -1 & 2 \end{bmatrix}$$

and $b_N, c_N \in \mathbb{R}^{n\times 1}$ are given by

$$\begin{aligned} (b_N)_i &= \int_0^1 \beta(s)\varphi_i^N(s)ds, \qquad i = 1,\dots,n, \\ (c_N)_i &= \int_0^1 \gamma(s)\varphi_i^N(s)ds, \qquad i = 1,\dots,n. \end{aligned}$$

Here $\{\varphi_i^N\}_{i=1}^n$ is the B–spline basis for the chosen finite-dimensional subspace of the underlying Hilbert space. The functions $\beta, \gamma \in L_2(0,1)$ used here are defined by

$$\begin{aligned} \beta(s) &= \begin{cases} b, & s \in [\beta_1,\ \beta_2] \\ 0, & \text{otherwise} \end{cases} \\ \gamma(s) &= \begin{cases} c, & s \in [\gamma_1,\ \gamma_2] \\ 0, & \text{otherwise} \end{cases} \end{aligned}$$

Thus, besides the system dimension n, the problem has the parameters a, b, c, β_1, β_2, γ_1, and γ_2. The default values given in the table are taken from [46]. Any other parameter combination may be used for generating the data. Increasing values of n, respectively N, result in a finer grid for the underlying approximation scheme.

Approximate solution of infinite-dimensional operator Riccati equations is one source of large-scale matrix Riccati equations. Another is the optimal control problem for *second-order models* as described for example in [23, 36]. In this type of problems, the dynamical system is given in terms of a second-order differential equation

$$M\ddot{z} + L\dot{z} + Kz = Du \tag{12}$$

and an associated output

$$y = Nz + P\dot{z} \tag{13}$$

or alternatively

$$\tilde{y} = \begin{bmatrix} N & 0 \\ 0 & P \end{bmatrix} \begin{bmatrix} z \\ \dot{z} \end{bmatrix} \tag{14}$$

where $z \in \mathbb{R}^\ell$, $M, L, K \in \mathbb{R}^{\ell \times \ell}$, $D \in \mathbb{R}^{\ell \times m}$, and $N, P \in \mathbb{R}^{p \times \ell}$. Often, M and K are symmetric where M is positive definite, K is positive semidefinite, and L is the sum of a symmetric positive semidefinite and a skew-symmetric matrix. Usually, M is called the *mass matrix*, L is the *Rayleigh* matrix representing damping (the symmetric part) and gyroscopic (the skew-symmetric part) forces, and K is the *stiffness matrix*. Second-order models are often used to model mechanical systems such as large flexible space structures.

A first-order realization of this problem may be obtained by introducing the state vector $x = \begin{bmatrix} z \\ \dot{z} \end{bmatrix}$. This yields a system of the form

$$\dot{x} = \begin{bmatrix} 0 & I \\ -M^{-1}K & -M^{-1}L \end{bmatrix} x + \begin{bmatrix} 0 \\ M^{-1}D \end{bmatrix} u \tag{15}$$

$$y = [\, N \;\; P \,]\, x, \tag{16}$$

or, with (14),

$$\tilde{y} = \begin{bmatrix} N & 0 \\ 0 & P \end{bmatrix} x. \tag{17}$$

This is a standard system as in (3) with $n = 2\ell$. The weighting matrices $\tilde{Q}$ and R in the cost functional (2) can then be chosen depending on the problem.

Here we give two examples of linear-quadratic control problems for second-order models.

Example 4.3 [26, Example 3]

n	m	p	parameter	$\lvert\lambda^{re}_{\min}\rvert$	$\lVert H \rVert$	$\kappa(H)$	$\lVert X_* \rVert$	$\kappa(X_*)$	K_U
2ℓ	2	2ℓ	$\ell = 30$, $\mu = 4.0$, $\delta = 4.0$, $\kappa = 1.0$	6.2×10^{-3}	2.2	1.1×10^5	2.2×10^2	4.5×10^2	1.5×10^3

This is a model of a string consisting of coupled springs, dashpots, and masses as shown in Figure 1. The inputs are two forces, one acting on the left end of the string, the other one on the right end. For this

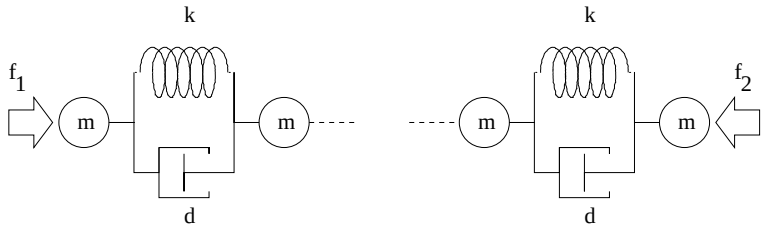

Figure 1: Coupled Spring Experiment ($k \sim \kappa$, $m \sim \mu$, $d \sim \delta$)

problem, the matrices in (12), (14) are

$$M = \mu I_\ell, \qquad L = \delta I_\ell, \qquad N = P = I_\ell,$$

$$K = \kappa \begin{bmatrix} 1 & -1 & 0 & \dots & 0 & 0 \\ -1 & 2 & -1 & \dots & 0 & 0 \\ 0 & -1 & 2 & \dots & 0 & 0 \\ \vdots & & \ddots & \ddots & \ddots & \dots \\ 0 & 0 & \dots & -1 & 2 & -1 \\ 0 & 0 & \dots & 0 & -1 & 1 \end{bmatrix}, \qquad D = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ \vdots & \vdots \\ \vdots & \vdots \\ 0 & 0 \\ 0 & -1 \end{bmatrix}.$$

The cost functional (2) is chosen as $J(x_0, u) = \int_0^\infty \left(y(t)^T y(t) + u(t)^T u(t)\right)\, dt$, i.e., $\tilde{Q} = I_{2\ell}$ and $R = I_2$.

Example 4.4 [42, 48]

n	m	p	parameter	$\lvert\lambda^{re}_{\min}\rvert$	$\lVert H\rVert$	$\kappa(H)$	$\lVert X_*\rVert$	$\kappa(X_*)$	K_U
$2\ell-1$	ℓ	ℓ	$\underline{\ell = 211}$	1.6×10^{-2}	4.1×10^{11}	5.5×10^{14}	7.2×10^{7}	1.4×10^{22}	3.0×10^{8}

This example describes a problem arising in power plants. We consider a model of a rotating axle with several masses placed upon it. These masses may be parts of turbines or generators and are assumed to be symmetric with respect to the axle. The input to the system consists of changing loads which act on the masses. This causes vibrations in the axle. The aim is to minimize the moments between two neighboring masses in order to maximize the life expectancy of the axle.

The system matrices in (12) and (13) are given as

$$M = \begin{bmatrix} \mu_1 & & \\ & \ddots & \\ & & \mu_\ell \end{bmatrix}, \qquad K = \begin{bmatrix} \kappa_1 & -\kappa_1 & & & & \\ -\kappa_1 & \kappa_1 + \kappa_2 & -\kappa_2 & & & \\ & \ddots & \ddots & \ddots & & \\ & & -\kappa_{n-2} & \kappa_{n-2} + \kappa_{n-1} & -\kappa_{n-1} & \\ & & & -\kappa_{n-1} & \kappa_{n-1} \end{bmatrix},$$

$$L = \begin{bmatrix} \delta_1 + \gamma_1 & -\gamma_1 & & & \\ -\gamma_1 & \gamma_1 + \delta_2 + \gamma_2 & -\gamma_2 & & \\ & \ddots & \ddots & \ddots & \\ & & -\gamma_{\ell-2} & \gamma_{\ell-2} + \delta_{\ell-1} + \gamma_{\ell-1} & -\gamma_{\ell-1} \\ & & & -\gamma_{\ell-1} & \gamma_{\ell-1} + \delta_\ell \end{bmatrix}, \qquad D = I_\ell,$$

$$N = \begin{bmatrix} 0 & 0 & & \\ \kappa_1 & -\kappa_1 & & \\ & \ddots & \ddots & \\ & & \kappa_{\ell-1} & -\kappa_{\ell-1} \end{bmatrix}, \qquad P = \begin{bmatrix} 1 & 0 & & \\ \gamma_1 & -\gamma_1 & & \\ & \ddots & \ddots & \\ & & \gamma_{\ell-1} & -\gamma_{\ell-1} \end{bmatrix}.$$

Hence the mathematical model of this problem is defined by ℓ and the parameter vectors

$\mu \in \mathbb{R}^\ell$	—	the moments of inertia of the masses,
$\delta \in \mathbb{R}^\ell$	—	the outer damping forces,
$\gamma \in \mathbb{R}^{\ell-1}$	—	the damping forces between two neighboring masses, and
$\kappa \in \mathbb{R}^{\ell-1}$	—	the spring constants of the axle part between two neighboring masses.

The resulting system is neither observable nor detectable. We may overcome this problem by eliminating the unobservable state variable as follows.

At first, a linear transformation in the state space is performed. It is known that such a transformation preserves the system properties (i.e., controllability, observability, stabilizability, detectability) if the transformation matrix is regular; see, e.g., [49].

As transformation matrix we choose

$$T = \begin{bmatrix} 0 & \hat{T} \\ \hat{T} & 0 \end{bmatrix} \in \mathbb{R}^{2\ell \times 2\ell},$$

where $\hat{T} \in \mathbb{R}^{\ell \times \ell}$ is the lower triangular matrix

$$\hat{T} = \begin{bmatrix} 1 & 0 & & & \\ 1 & -1 & & & \\ 1 & -1 & -1 & & \\ \vdots & \vdots & \vdots & \ddots & \\ 1 & -1 & -1 & \dots & -1 \end{bmatrix}.$$

The inverse of T is

$$T^{-1} = \begin{bmatrix} 0 & \hat{T}^{-1} \\ \hat{T}^{-1} & 0 \end{bmatrix}, \qquad \hat{T}^{-1} = \begin{bmatrix} 1 & & & & \\ 1 & -1 & & & \\ & 1 & -1 & & \\ & & \ddots & \ddots & \\ & & & 1 & -1 \end{bmatrix}.$$

The resulting system corresponding to (15) is then given by

$$\begin{aligned} \dot{\hat{x}} &= \hat{A}\hat{x} + \hat{B}u, \\ y &= \hat{C}\hat{x}, \end{aligned}$$

where $\hat{x} = T^{-1}x$ and

$$\begin{aligned} \hat{A} &= T^{-1} \begin{bmatrix} 0 & I \\ -M^{-1}K & -M^{-1}L \end{bmatrix} T = \begin{bmatrix} -\hat{T}^{-1}M^{-1}L\hat{T} & -\hat{T}^{-1}M^{-1}L\hat{T} \\ I & 0 \end{bmatrix}, \\ \hat{B} &= T^{-1} \begin{bmatrix} 0 \\ M^{-1} \end{bmatrix} = \begin{bmatrix} \hat{T}^{-1}M^{-1} \\ 0 \end{bmatrix}, \\ \hat{C} &= [N \;\; P]T = [P\hat{T} \;\; N\hat{T}]. \end{aligned} \tag{18}$$

Now the $(\ell+1)$st columns of $\hat{A}$ and $\hat{C}$ are zero, that is, the $(\ell+1)$st component of $\hat{x}$ is the undetectable state variable. We thus obtain a stabilizable/detectable system with the same input/output behavior as (18) by removing this component from the system. This is equivalent to removing the $(\ell+1)$st columns of $\hat{A}$ and $\hat{C}$ and the $(\ell+1)$st rows of $\hat{A}$ and $\hat{B}$. The resulting system matrices are $A \in \mathbb{R}^{(2\ell-1)\times(2\ell-1)}$, $B \in \mathbb{R}^{(2\ell-1)\times\ell}$, and $C \in \mathbb{R}^{\ell\times(2\ell-1)}$.

The weighting matrix $\tilde{Q}$ in the cost functional (2) is chosen to normalize the rows of C, i.e., $\tilde{Q} = W_C^T W_C$ where $W_C \in \mathbb{R}^{\ell\times\ell}$ is a diagonal weighting matrix such that the rows of $W_C C$ have unit length. The control weighting matrix R is chosen as an identity matrix of size $\ell \times \ell$.

As default values we use data provided by [33] corresponding to a generator axle in a power plant. The dimension of the problem ($\ell = 211$) prevents printing the data. For generating the system matrices we provide data files for use with FORTRAN 77 (see Appendix A).

For the default data, the Hamiltonian matrix has a very large norm and condition number despite the scaling of the output matrix. (Without the scaling corresponding to W_C, these values are even larger by about 10 orders of magnitude.) This is due to the large entries in A, i.e., the large values κ_j. The reference solution was computed by the sign function method as proposed in [17] where the defect correction was performed using Newton's method combined with exact line search [9]. Due to the bad scaling of this example, it was necessary to scale the Lyapunov equation (5) by $\|A\|_\infty$ when computing K_U.

Acknowledgments

We would like to express our gratitude to the co-authors of the first version of the benchmark collection CAREX, Volker Mehrmann and Alan J. Laub who have layed the basis for the collection. Moreover, we wish to thank Vasile Sima for his efforts in integrating the FORTRAN 77 subroutine DCAREX into SLICOT.

A The FORTRAN 77 subroutine DCAREX.F

This is the prolog of a FORTRAN 77 subroutine for generating all presented examples. The subroutine was documented according to standards for SLICOT[1] and the SLICOT benchmark collection [39, 38]. The routine will serve as the basis for the SLICOT benchmark routine `BB01AD`. Slight modifications of DCAREX.F due to the integration into the library routine `BB01AD` may be necessary.

For some of the examples, DCAREX reads the data from data files delivered together with DCAREX.F. These are Examples 1.3–1.6, and 4.4. The corresponding data files are, according to the naming convention proposed in [38],

```
BB01103.dat   BB01104.dat   BB01105.dat   BB01106.dat
BB012091.dat  BB012092.dat
BB01404.dat
```

A data file for Example 4.4 may be supplied by the user. In this case, on entry to DCAREX, `N` must contain the integer ℓ, i.e., the order of the second order model (12) and the CHARACTER variable `DATAF` must contain the name of the file. In the data file, the user must provide in consecutive order vectors μ (length ℓ), δ (length ℓ), γ (length $\ell - 1$), and κ (length $\ell - 1$).

Besides calls to LAPACK[2] and BLAS[3] [2], DCAREX calls the subroutines DSP2SY and DSY2SP which are used to convert symmetric matrices from general storage mode to packed storage mode and vice versa. These subroutines are provided in the file DCAREX.F.

The SLICOT subroutine BB01AD together with an example program calling BB01AD, test data and results as well as online documentation can be found in the benchmark chapter of SLICOT at `ftp://wgs.esat.kuleuven.ac.be/pub/WGS/SLICOT/libindex.html#B`.

[1] **S**ubroutine **LI**brary in **CO**ntrol and Systems **T**heory
[2] Available from `http://www.netlib.org/lapack`
[3] Available from `http://www.netlib.org/blas`

```
      SUBROUTINE DCAREX(DEF, NR, DPAR, IPAR, BPAR, CHPAR, VEC, N, M, P,
     1                  A, LDA, B, LDB, C, LDC, G, LDG, Q, LDQ, X, LDX,
     2                  DWORK, INFO)
C
C     .. Scalar Arguments ..
      INTEGER           LDA, LDB, LDC, LDG, LDQ, LDX, INFO, N, M, P
      CHARACTER         DEF
C
C     .. Array Arguments ..
      INTEGER           NR(2), IPAR(3)
      DOUBLE PRECISION A(LDA,*), B(LDB,*), C(LDC,*), G(*), Q(*),
     1                  X(LDX,*), DPAR(*), DWORK(*)
      CHARACTER         CHPAR*255
      LOGICAL           BPAR(6), VEC(9)
C
C
C     PURPOSE
C
C     To generate the benchmark examples for the numerical solution of
C     continuous-time algebraic Riccati equations (CARE) of the form
C
C       0 = Q + A'X + XA - XGX
C
C     corresponding to the Hamiltonian matrix
C
C            (  A   G  )
C        H = (       T ).
C            (  Q  -A  )
C
C     A,G,Q,X are real N-by-N matrices, Q and G are symmetric and may
C     be given in factored form
C
C                    -1 T                          T
C      (I)   G = B R  B  ,           (II)   Q = C W C .
C
C     Here, C is P-by-N, W P-by-P, B N-by-M, and R M-by-M, where W
C     and R are symmetric. In linear-quadratic optimal control problems,
C     usually W is positive semidefinite and R positive definite.  The
C     factorized form can be used if the CARE is solved the deflating
C     subspaces of the extended Hamiltonian pencil
C
C                  (  A   0   B  )       (  I   0   0  )
C                  (      T     )       (             )
C        H - s K = (  Q   A   0  )  -  s (  0  -I   0  ) ,
C                  (      T     )       (             )
```

```
C                 (  0   B   R  )       (  0   0   0  )
C
C     where I and 0 denote the identity and zero matrix, respectively,
C     of appropriate dimensions.
C
C     NOTE: the formulation of the CARE and the related matrix (pencils)
C           used here does not include CAREs as they arise in robust
C           control (H_infinity optimization).
C
C
C     ARGUMENTS
C
C     Mode Parameters
C
C     DEF     CHARACTER
C             This parameter specifies if the default parameters are
C             to be used or not.
C             = 'N' or 'n' : The parameters given in the input vectors
C                            xPAR (x = 'D', 'I', 'B', 'CH') are used.
C             = 'D' or 'd' : The default parameters for the example
C                            are used.
C             This parameter is not referenced if NR(1) = 1.
C
C     Input/Output Parameters
C
C     NR      (input) INTEGER array, dimension (2)
C             This array determines the example for which CAREX returns
C             data. NR(1) is the group of examples.
C             NR(1) = 1 : parameter-free problems of fixed size.
C             NR(1) = 2 : parameter-dependent problems of fixed size.
C             NR(1) = 3 : parameter-free problems of scalable size.
C             NR(1) = 4 : parameter-dependent problems of scalable size.
C             NR(2) is the number of the example in group NR(1).
C             Let be NEXi the number of examples in group i. Currently,
C             NEX1 = 6, NEX2 = 9, NEX3 = 2, NEX4 = 4.
C             1 .LE. NR(1) .LE. 4.
C             1 .LE. NR(2) .LE. NEXi , where i = NR(1).
C
C     IPAR    (input/output) INTEGER array, dimension (3)
C             On input, IPAR(1) determines the actual state dimension,
C             i.e., the order of the matrix A as follows, where
C             NO = NR(1).NR(2).
C             NR(1) = 1 or 2.1-2.8: IPAR(1) is ignored.
C             NO = 2.9            : IPAR(1) = 1 generates the CARE for
C                                   optimal state feedback (default);
```

```
C                                IPAR(1) = 2 generates the Kalman
C                                filter CARE.
C            NO = 3.1          : IPAR(1) is the number of vehicles
C                                (parameter 'l' in the description
C                                 in [1]).
C            NO = 3.2, 4.1 or 4.2: IPAR(1) is the order of the matrix
C                                A.
C            NO = 4.3 or 4.4   : IPAR(1) determines the dimension of
C                                the second-order system, i.e., the
C                                order of the stiffness matrix for
C                                Examples 4.3 and 4.4 (parameter 'l'
C                                in the description in [1]).
C
C            The order of the output matrix A is N = 2*IPAR(1) for
C            Example 4.3 and N = 2*IPAR(1)-1 for Examples 3.1 and 4.4.
C            NOTE that IPAR(1) is overwritten for Examples 1.1-2.8. For
C            the other examples, IPAR(1) is overwritten if the default
C            parameters are to be used.
C            On output, IPAR(1) contains the order of the matrix A.
C
C            On input, IPAR(2) is the number of colums in the matrix B
C            in (I) (in control problems, the number of inputs of the
C            system). Currently, IPAR(2) is fixed or determined by
C            IPAR(1) for all examples and thus is not referenced on
C            input.
C            On output, IPAR(2) is the number of columns of the
C            matrix B from (I).
C            NOTE that currently IPAR(2) is overwritten and that
C            rank(G) <= IPAR(2).
C
C            On input, IPAR(3) is the number of rows in the matrix C
C            in (II) (in control problems, the number of outputs of the
C            system). Currently, IPAR(3) is fixed or determined by
C            IPAR(1) for all examples and thus is not referenced on
C            input.
C            On output, IPAR(3) contains the number of rows of the
C            matrix C in (II).
C            NOTE that currently IPAR(3) is overwritten and that
C            rank(Q) <= IPAR(3).
C
C     DPAR   (input/output) DOUBLE PRECISION array, DIMENSION (7)
C            Double precision parameter vector. For explanation of the
C            parameters see [1].
C            DPAR(1)          : defines the parameters
C                               'delta' for NR(1) = 3,
```

```
C                                 'q' for NR(1).NR(2) = 4.1,
C                                 'a' for NR(1).NR(2) = 4.2, and
C                                 'mu' for NR(1).NR(2) = 4.3.
C             DPAR(2)           : defines parameters
C                                 'r' for NR(1).NR(2) = 4.1,
C                                 'b' for NR(1).NR(2) = 4.2, and
C                                 'delta' for NR(1).NR(2) = 4.3.
C             DPAR(3)           : defines parameters
C                                 'c' for NR(1).NR(2) = 4.2 and
C                                 'kappa' for NR(1).NR(2) = 4.3.
C             DPAR(j), j=4,5,6,7: These arguments are only used to
C                                 generate Example 4.2 and define in
C                                 consecutive order the intervals
C                                 ['beta_1', 'beta_2'],
C                                 ['gamma_1', 'gamma_2'].
C             NOTE that DPAR is overwritten whith the default parameters
C             if DEF = 'D' or 'd'.
C
C     BPAR    (input/output) BOOLEAN array, dimension 6.
C             This array defines the form of the output of the examples
C             and the storage mode of the matrices G and Q.
C             BPAR(1) = .TRUE.  : G is returned.
C             BPAR(1) = .FALSE. : G is returned in factored form, i.e.,
C                                 B and R from (I) are returned.
C             BPAR(2) = .TRUE.  : The matrix returned in array G (i.e.,
C                                 G if BPAR(1) = .TRUE. and R if
C                                 BPAR(1) = .FALSE.) is stored as full
C                                 matrix.
C             BPAR(2) = .FALSE. : The matrix returned in array G is
C                                 provided in packed storage mode.
C             BPAR(3) = .TRUE.  : If BPAR(2) = .FALSE., the matrix
C                                 returned in array G is stored in upper
C                                 packed mode, i.e., the upper triangle
C                                 of a symmetric n-by-n matrix is stored
C                                 by columns, e.g., the matrix entry
C                                 G(i,j) is stored in the array entry
C                                 G(i+j*(j-1)/2)  for i <= j.
C                                 Otherwise, this entry is ignored.
C             BPAR(3) = .FALSE. : If BPAR(2) = .FALSE., the matrix
C                                 returned in array G is stored in lower
C                                 packed mode, i.e., the lower triangle
C                                 of a symmetric n-by-n matrix is stored
C                                 by columns, e.g., the matrix entry
C                                 G(i,j) is stored in the array entry
C                                 G(i+(2*n-j)*(j-1)/2) for j <= i.
```

```
C                                Otherwise, this entry is ignored.
C            BPAR(4) = .TRUE.  : Q is returned.
C            BPAR(4) = .FALSE. : Q is returned in factored form, i.e.,
C                                C and W from (II) are returned.
C            BPAR(5) = .TRUE.  : The matrix returned in array Q (i.e.,
C                                Q if BPAR(4) = .TRUE. and W if
C                                BPAR(4) = .FALSE.) is stored as full
C                                matrix.
C            BPAR(5) = .FALSE. : The matrix returned in array Q is
C                                provided in packed storage mode.
C            BPAR(6) = .TRUE.  : If BPAR(5) = .FALSE., the matrix
C                                returned in array Q is stored in upper
C                                packed mode (see above).
C                                Otherwise, this entry is ignored.
C            BPAR(6) = .FALSE. : If BPAR(5) = .FALSE., the matrix
C                                returned in array Q is stored in lower
C                                packed mode (see above).
C                                Otherwise, this entry is ignored.
C            NOTE that there are no default values for BPAR.  If all
C            entries are declared to be .TRUE., then matrices G and Q
C            are returned in conventional storage mode, i.e., as an
C            N-by-N arrays where the array element Z(I,J) contains the
C            matrix entry Z_{i,j}.
C
C     CHPAR  (input/output) CHARACTER*255.
C            On input, this is the name of a data file supplied by the
C            user.
C            In the current version, only Example 4.4 allows a
C            user-defined data file. This file must contain
C            consecutively DOUBLE PRECISION vectors mu, delta, gamma,
C            and kappa. The length of these vectors is determined by
C            the input value for IPAR(1).
C            If on entry, IPAR(1) = L, then mu and delta must each
C            contain L, gamma and kappa each L-1 DOUBLE PRECISION
C            values.
C            On output, this string contains short information about the
C            chosen example.
C
C     VEC    (output) LOGICAL array, dimension (9)
C            Flag vector which displays the availability of the output
C            data:
C            VEC(j), j=1,2,3, refer to N, M, and P, respectively, and
C            are always .TRUE.
C            VEC(4) refers to A and is always .TRUE.
C            VEC(5) is .TRUE. if BPAR(1) = .FALSE., i.e., the factors B
```

```
C             and R from (I) are returned.
C             VEC(6) is .TRUE. if BPAR(4) = .FALSE., i.e., the factors C
C             and W from (II) are returned.
C             VEC(7) refers to G and is always .TRUE.
C             VEC(8) refers to Q and is always .TRUE.
C             VEC(9) refers to X and is = .TRUE. if the exact solution
C             matrix is available.
C
C     N       (output) INTEGER
C             The order of the matrices A, X, G if BPAR(1) = .TRUE., and
C             Q if BPAR(4) = .TRUE.
C
C     M       (output) INTEGER
C             The number of columns in the matrix B (or the dimension of
C             the control input space of the underlying dynamical
C             system).
C
C     P       (output) INTEGER
C             The number of rows in the matrix C (or the dimension of
C             the output space of the underlying dynamical system).
C
C     A       (output) DOUBLE PRECISION array, DIMENSION (LDA,N)
C             The leading N-by-N part of this array contains the
C             coefficient matrix A of the CARE.
C
C     LDA     INTEGER
C             The leading dimension of array A as declared in the
C             calling program.
C             LDA .GE. N.
C
C     B       (output) DOUBLE PRECISION array, DIMENSION (LDB,M).
C             If (BPAR(1) = .FALSE.) then array B contains the matrix B
C             of the factored form (I) of G. Otherwise, B is used as
C             workspace.
C
C     LDB     INTEGER
C             The leading dimension of array B as declared in the
C             calling program.
C             LDB .GE. M.
C
C     C       (output) DOUBLE PRECISION array, DIMENSION (LDC,N).
C             If (BPAR(4) = .FALSE.) then array C contains the matrix C
C             of the factored form (II) of Q. Otherwise, C is used as
C             workspace.
C
```

```
C     LDC     INTEGER
C             The leading dimension of array C as declared in the
C             calling program.
C             LDC .GE. P  where P is the number of rows of the matrix C,
C             i.e., the output value of IPAR(3). (For all examples,
C             P .LE. N, where N equals the output value of the argument
C             IPAR(1), i.e., LDC .GE. LDA is always safe.)
C
C     G       (output) DOUBLE PRECISION array, DIMENSION at least NG
C             If (BPAR(2) = .TRUE.)  then NG = LDG*N.
C             If (BPAR(2) = .FALSE.) then NG = N*(N+1)/2.
C             If (BPAR(1) = .TRUE.), then array G contains the
C             coefficient matrix G of the CARE.
C             If (BPAR(1) = .FALSE.), then array G contains the 'control
C             weighting matrix' R of G's factored form as in (I).
C             The symmetric matrix contained in array G is stored
C             according to BPAR(2) and BPAR(3).
C
C     LDG     INTEGER
C             If conventional storage mode is used for G, i.e.,
C             BPAR(2)  = .TRUE., then G is stored like a 2-dimensional
C             array with leading dimension LDG. If packed symmetric
C             storage mode is used, then LDG is not referenced.
C             LDG .GE. N if BPAR(2) = .TRUE..
C
C     Q       (output) DOUBLE PRECISION array, DIMENSION at least NQ.
C             If (BPAR(5) = .TRUE.)  then NQ = LDQ*N.
C             If (BPAR(5) = .FALSE.) then NQ = N*(N+1)/2.
C             If (BPAR(4) = .TRUE.), then array Q contains the
C             coefficient matrix Q of the CARE.
C             If (BPAR(4) = .FALSE.)), then array Q contains the 'output
C             weighting matrix' W of Q's factored form as in (II).
C             The symmetric matrix contained in array Q is stored
C             according to BPAR(5) and BPAR(6).
C
C     LDQ     INTEGER
C             If conventional storage mode is used for Q, i.e.,
C             BPAR(5) = .TRUE., then Q is stored like a 2-dimensional
C             array with leading dimension LDQ. If packed symmetric
C             storage mode is used, then LDQ is not referenced.
C             LDQ .GE. N if BPAR(5) = .TRUE..
C
C     X       (output) DOUBLE PRECISION array, DIMENSION (LDX,IPAR(1)).
C             If an exact solution is available (Nr = 1.1, 1.2, 2.1,
C             2.3-2.6, 3.2), then the leading N-by-N part of this array
```

```
C             contains the solution matrix X in conventional storage
C             mode. Otherwise, X is not referenced.
C
C     LDX     INTEGER.
C             The leading dimension of array X as declared in the
C             calling program.
C             LDX .GE. N if NR = 1.1, 1.2, 2.1, 2.3-2.6, 3.2.
C
C     Work Space
C
C     DWORK   DOUBLE PRECISION array, dimension (LDWORK)
C
C     LDWORK  INTEGER
C             The length of the array DWORK.
C             LDWORK .GE. N*MAX(4,N).
C
C
C     Error Indicator
C
C     INFO    INTEGER.
C             = 0 : successful exit;
C             < 0 : if INFO = -i, the argument no. i had an illegal value.
C             = 1 : Data file could not be opened or had wrong format.
C             = 2 : Division by zero.
C             = 3 : G can not be computed as in (I) due to a singular R
C                   matrix.
C
C
C     REFERENCE
C
C     [1] J. ABELS and P. BENNER
C         CAREX - A Collection of Benchmark Examples for Continuous-Time
C         Algebraic Riccati Equations (Version 2.0)
C         SLICOT Working Note 1999-14, November 1999. Available from
C         http://www.win.tue.nl/niconet/NIC2/reports.html.
C
C     This is an updated and extended version of
C
C     [2] P. BENNER, A.J. LAUB and V. MEHRMANN
C         A Collection of Benchmark Examples for the Numerical Solution
C         of Algebraic Riccati Equations I: Continuous-Time Case.
C         Technical Report SPC 95_22, Fak. f. Mathematik,
C         TU Chemnitz-Zwickau (Germany), October 1995.
C
C
```

```
C     NUMERICAL ASPECTS
C
C     If the original data as taken from the literature is given via
C     matrices G and Q, but factored forms are requested as output, then
C     these factors are obtained from Cholesky or LDLT decompositions of
C     G and Q, i.e., the output data will be corrupted by roundoff
C     errors.
C
C     CONTRIBUTOR
C
C     Peter Benner (Universitaet Bremen)
C
C     For questions concerning the collection or for the submission of
C     test examples, please send e-mail to benner@math.uni-bremen.de.
C
C     REVISIONS
C
C     1995, January 18.
C     1995, October 12.
C     1996, January 18 (bug fix).
C     1999, November 15.
C
C     KEYWORDS
C
C     algebraic Riccati equation, Hamiltonian matrix
C
```

B Reference table

The Table on the next page summarizes the properties of all the presented examples. A value "∞" for $\kappa(X)$ or $\kappa(H)$ means that the corresponding matrix is not invertible with respect to the numerical rank computed by MATLAB. $K_U = \infty$ represents a singular Lyapunov equation (5). The column X_* indicates whether an analytical stabilizing solution is available ("+") or not ("–").

no.	n	m	p	default	X_*	$\lvert\lambda^{re}_{\min}\rvert$	$\lVert H\rVert$	$\kappa(H)$	$\lVert X\rVert$	$\kappa(X)$	K_U
1.1	2	1	2	–	+	1.0	2.4	5.8	3.0	3.0	5.0
1.2	2	1	2	–	+	0.50	16.2	2.6×10^2	31.4	∞	52.6
1.3	4	2	4	–	–	0.73	7.8	55.8	6.1	2.2×10^2	21.9
1.4	8	2	8	–	–	0.10	3.4	3.1×10^2	4.8	1.3×10^3	33.6
1.5	9	3	9	–	–	0.34	2.2×10^2	3.4×10^3	2.7	1.1×10^3	8.5×10^2
1.6	30	3	5	–	–	0.18	1.4×10^8	1.5×10^{10}	3.6×10^3	∞	3.7×10^9
2.1	2	1	1	$\varepsilon = 10^{-6}$	+	1.0	3.0	5.2	2.0×10^{12}	8.0×10^{12}	3.0
2.2	2	2	1	$\varepsilon = 10^{-8}$	–	0.70	1.0×10^6	1.5×10^8	9.3×10^3	9.4×10^6	6.7×10^9
2.3	2	1	2	$\varepsilon = 10^6$	+	7.1×10^2	1.0×10^6	1.0×10^6	1.4×10^3	2.0×10^6	8.7×10^5
2.4	2	2	2	$\varepsilon = 10^{-7}$	+	1.4×10^{-7}	2.6	1.3×10^{14}	4.0	1.7×10^7	3.8×10^3
2.5	2	1	2	$\varepsilon = 0$	+	0.0	15.4	58.4	2.6	6.9	∞
2.6	3	3	3	$\varepsilon = 10^6$	+	1.0×10^6	3.5×10^6	3.5	6.0×10^{12}	3.0	2.7
2.7	4	1	2	$\varepsilon = 10^{-6}$	–	0.25	1.0×10^{12}	5.7×10^{13}	13.2	9.1×10^8	4.1×10^{13}
2.8	4	1	1	$\varepsilon = 10^{-6}$	–	5.0×10^{-13}	4.2	17.9	1.0	1.0	1.0×10^{13}
2.9	55	2	10	# 1	–	2.9×10^{-2}	4.4×10^{10}	1.3×10^{16}	1.2×10^3	∞	5.3×10^{12}
3.1	$2N-1$	N	$N-1$	$N = 20$	–	0.66	10.0	4.3×10^2	28.8	∞	50.9
3.2	n	n	n	$n = 64$	+	1.0	4.1	4.1	1.0	8.1	5.0
4.1	n	1	1	$n = 21$, $q = r = 1.0$	–	7.5×10^{-2}	1.0	1.0	2.4×10^9	∞	1.3×10^9
4.2	n	1	1	$n = 100$, $a = 0.01$, $b = c = 1.0$, $[\beta_1, \beta_2] = [0.2, 0.3]$, $[\gamma_1, \gamma_2] = [0.2, 0.3]$	–	0.1	1.2×10^3	1.4×10^5	7.1×10^{-4}	∞	1.0×10^4
4.3	2ℓ	2	2ℓ	$\ell = 30$, $\mu = \delta = 4.0$, $\kappa = 1.0$	–	6.2×10^{-3}	2.2	1.1×10^5	2.2×10^2	4.5×10^2	1.5×10^3
4.4	$2\ell-1$	ℓ	ℓ	$\ell = 211$	–	$1.6 \times 10^{--2}$	4.1×10^{11}	5.5×10^{14}	7.2×10^7	1.4×10^{22}	3.0×10^8

References

[1] G.S. Ammar, P. Benner, and V. Mehrmann. A multishift algorithm for the numerical solution of algebraic Riccati equations. *Electr. Trans. Num. Anal.*, 1:33–48, 1993.

[2] E. Anderson, Z. Bai, C. Bischof, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, S. Ostrouchov, and D. Sorensen. *LAPACK Users' Guide*. SIAM, Philadelphia, PA, second edition, 1995.

[3] W.F. Arnold, III. *On the Numerical Solution of Algebraic Matrix Riccati Equations*. PhD thesis, Univ. of Southern California, Dept. of Elec. Eng.—Systems, Los Angeles, CA, December 1983.

[4] W.F. Arnold, III and A.J. Laub. Generalized eigenproblem algorithms and software for algebraic Riccati equations. *Proc. IEEE*, 72:1746–1754, 1984.

[5] M. Athans, W.S. Levine, and A. Levis. A system for the optimal and suboptimal position and velocity control for a string of high-speed vehicles. In *Proc. 5th Int. Analogue Computation Meetings*, Lausanne, Switzerland, September 1967.

[6] Z. Bai and Q. Qian. Inverse free parallel method for the numerical solution of algebraic Riccati equations. In J.G. Lewis, editor, *Proc. Fifth SIAM Conf. Appl. Lin. Alg., Snowbird, UT, June 1994*, pages 167–171. SIAM, Philadelphia, PA, 1994.

[7] S. Beale and B. Shafai. Robust control system design with a proportional-integral observer. *Internat. J. Control*, 50:97–111, 1989.

[8] P. Benner and R. Byers. Step size control for Newton's method applied to algebraic Riccati equations. In J.G. Lewis, editor, *Proc. Fifth SIAM Conf. Appl. Lin. Alg., Snowbird, UT*, pages 177–181. SIAM, Philadelphia, PA, 1994.

[9] P. Benner and R. Byers. An exact line search method for solving generalized continuous-time algebraic Riccati equations. *IEEE Trans. Automat. Control*, 43(1):101–107, 1998.

[10] P. Benner, A.J. Laub, and V. Mehrmann. A collection of benchmark examples for the numerical solution of algebraic Riccati equations I: Continuous-time case. Technical Report SPC 95_22, Fakultät für Mathematik, TU Chemnitz–Zwickau, 09107 Chemnitz, FRG, 1995. Available from `http://www.tu-chemnitz.de/sfb393/spc95pr.html`.

[11] P. Benner, A.J. Laub, and V. Mehrmann. Benchmarks for the numerical solution of algebraic Riccati equations. *IEEE Control Systems Magazine*, 7(5):18–28, 1997.

[12] S. Bhattacharyya, A. Del Nero Gomes, and J. Howze. The structure of robust disturbance rejection control. *IEEE Trans. Automat. Control*, AC-28:874–881, 1983.

[13] A. Bunse-Gerstner, R. Byers, and V. Mehrmann. Numerical methods for algebraic Riccati equations. In S. Bittanti, editor, *Proc. Workshop on the Riccati Equation in Control, Systems, and Signals*, pages 107–116, Como, Italy, 1989.

[14] A. Bunse-Gerstner and V. Mehrmann. A symplectic QR-like algorithm for the solution of the real algebraic Riccati equation. *IEEE Trans. Automat. Control*, AC-31:1104–1113, 1986.

[15] A. Bunse-Gerstner, V. Mehrmann, and D. Watkins. An SR algorithm for Hamiltonian matrices based on Gaussian elimination. *Methods of Operations Research*, 58:339–358, 1989.

[16] R. Byers. Numerical condition of the algebraic Riccati equation. *Contemp. Math.*, 47:35–49, 1985.

[17] R. Byers. Solving the algebraic Riccati equation with the matrix sign function. *Linear Algebra Appl.*, 85:267–279, 1987.

[18] R. Byers. A bisection method for measuring the distance of a stable to unstable matrices. *SIAM J. Sci. Statist. Comput.*, 9:875–881, 1988.

[19] J. Chow and P. Kokotovic. A decomposition of near–optimum regulators for systems with slow and fast modes. *IEEE Trans. Automat. Control*, AC-21:701–705, 1976.

[20] E.J. Davison and W. Gesing. The systematic design of control systems for the multivariable servomechanism problem. In M.K Sain, J.L. Peczkowsky, and J.M. Melsa, editors, *Alternatives for Linear Multivariable Control*. Nat. Eng. Consortium Inc., Chicago, IL, 1978.

[21] E.J. Davison (editor). Benchmark problems for control system design. Technical report, Intl. Fed. Automatic Control (IFAC), May 1990.

[22] Z. Gajić and X. Shen. *Parallel Algorithms for Optimal Control of Large Scale Linear Systems*. Springer-Verlag, London, GB, 1994.

[23] J. D. Gardiner. Stabilizing control for second-order models and positive real systems. *AIAA J. Guidance, Dynamics and Control*, 15(1):280–282, 1992.

[24] J.D. Gardiner and A.J. Laub. A generalization of the matrix-sign-function solution for algebraic Riccati equations. *Internat. J. Control*, 44:823–832, 1986.

[25] G.H. Golub and C.F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, Baltimore, second edition, 1989.

[26] J.J. Hench, C. He, V. Kučera, and V. Mehrmann. Dampening controllers via a Riccati equation approach. Technical Report SPC 95_18, Fakultät für Mathematik, Technische Universität Chemnitz–Zwickau, 09107 Chemnitz, FRG, May 1995.

[27] D. Hinrichsen and A.J. Pritchard. Stability radius for structured perturbations and the algebraic Riccati equation. *Sys. Control Lett.*, 8:105–113, 1986.

[28] G.D. Ianculescu, J. Ly, A.J. Laub, and P.M. Papadopoulos. Space station Freedom solar array H_∞ control. Talk at 31st IEEE Conf. on Decision and Control, Tucson, AZ, December 1992.

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

[30] C. Kenney and G. Hewer. The sensitivity of the algebraic and differential Riccati equations. *SIAM J. Cont. Optim.*, 28:50–69, 1990.

[31] C. Kenney, A.J. Laub, and M. Wette. A stability-enhancing scaling procedure for Schur-Riccati solvers. *Sys. Control Lett.*, 12:241–250, 1989.

[32] D.L. Kleinman. On an iterative technique for Riccati equation computations. *IEEE Trans. Automat. Control*, AC-13:114–115, 1968.

[33] W. Lang. Personal communication, 1995.

[34] A.J. Laub. A Schur method for solving algebraic Riccati equations. *IEEE Trans. Automat. Control*, AC-24:913–921, 1979.

[35] A.J. Laub. Invariant subspace methods for the numerical solution of Riccati equations. In S. Bittanti, A.J. Laub, and J.C. Willems, editors, *The Riccati Equation*, pages 163–196. Springer-Verlag, Berlin, 1991.

[36] A.J. Laub and J.D. Gardiner. Hypercube implementation of some parallel algorithms in control. In M.J. Denham and A.J. Laub, editors, *Advanced Computing Concepts and Techniques in Control Engineering*, pages 361–390. Springer-Verlag, Berlin, 1988.

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

[38] V. Mehrmann and Thilo Penzl. Benchmark collections in SLICOT. SLICOT Working Note 1998-5, The Working Group on Software, June 1998. Available from `http://www.win.tue.nl/niconet/NIC2/reports.html`.

[39] The Working Group on Software. Slicot implementation and documentation standards 2.1. WGS-report 96-1, Available from `http://www.win.tue.nl/wgs/reports.html`, 1996.

[40] P. Pandey. On scaling an algebraic Riccati equation. In *Proc. American Control Conf.*, pages 1583–1587, San Francisco, CA, June 1993.

[41] L.M. Patnaik, N. Viswanadham, and I.G. Sarma. Computer control algorithms for a tubular ammonia reactor. *IEEE Trans. Automat. Control*, AC-25(4):642–651, 1980.

[42] T. Penzl. Vorkonditionierung bei der iterativen Lösung von Lyapunovgleichungen. Diplomarbeit, Technische Universität Chemnitz–Zwickau, Fak. f. Mathematik, 09107 Chemnitz, FRG, July 1994.

[43] P.Hr. Petkov, N.D. Christov, and M.M. Konstantinov. On the numerical properties of the Schur approach for solving the matrix Riccati equation. *Sys. Control Lett.*, 9:197–201, 1987.

[44] A.C. Raines and D.S. Watkins. A class of Hamiltonian–symplectic methods for solving the algebraic Riccati equation. *Linear Algebra Appl.*, 205/206:1045–1060, 1994.

[45] J.D. Roberts. Linear model reduction and solution of the algebraic Riccati equation by use of the sign function. *Internat. J. Control*, 32:677–687, 1980. (Reprint of Technical Report No. TR-13, CUED/B-Control, Cambridge University, Engineering Department, 1971).

[46] I.G. Rosen and C. Wang. A multi–level technique for the approximate solution of operator Lyapunov and algebraic Riccati equations. *SIAM J. Numer. Anal.*, 32(2):514–541, 1995.

[47] V. Sima. Algorithm GRICSR solving continuous-time algebraic Riccati equations using Gaussian symplectic transformations. *Stud. Res. Comp. Inf.*, 1:237–254, 1992.

[48] H. Wilharm. Ein praktisch orientiertes Verfahren der Beobachterauslegung bei schwach gedämpften Mehrmassensystemen mit deterministischen Meßstörungen. *Fortschritt–Berichte der VDI–Zeitschriften*, Reihe 8(64).

[49] W.M. Wonham. *Linear Multivariable Control: A Geometric Approach.* Springer-Verlag, New York, second edition, 1979.

[50] K. Zhou, J.C. Doyle, and K. Glover. *Robust and Optimal Control.* Prentice-Hall, Upper Saddle River, NJ, 1995.