Technische Universität Chemnitz-Zwickau

DFG-Forschergruppe “SPC” · Fakultät für Mathematik

Peter Benner Alan J. Laub Volker Mehrmann

A Collection of Benchmark Examples for the Numerical Solution of Algebraic Riccati Equations I: Continuous-Time Case

Preprint-Reihe der Chemnitzer DFG-Forschergruppe
“Scientific Parallel Computing”

SPC 95_22 Oktober 1995

A Collection of Benchmark Examples for the Numerical Solution of Algebraic Riccati Equations I: Continuous-Time Case

Peter Benner* Alan J. Laub† Volker Mehrmann*

October 18, 1995

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.

1 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

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

subject to the dynamics

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t), \quad x(0) = x_0, & (3)\\ y(t) &= Cx(t). & (4) \end{aligned}$$

*Fakultät für Mathematik, Technische Universität Chemnitz–Zwickau, 09107 Chemnitz, FRG. e-mail: benner@mathematik.tu-chemnitz.de, mehrmann@mathematik.tu-chemnitz.de. These authors have been supported by *Deutsche Forschungsgemeinschaft*, research grant *Me 790/7-1 Singuläre Steuerungsprobleme*.

†Department of Electrical and Computer Engineering, University of California, Santa Barbara, CA 93106-9560, e-mail: laub@ece.ucsb.edu. This author has been supported by *National Science Foundation Grant ECS-9120643* and *Air Force Office of Scientific Research Grant F49620-94-1-0104DEF*.

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. [31, 42]).
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. [11, 27, 28, 31, 42]). If this subspace is spanned by $\begin{bmatrix} U_1 \\ U_2 \end{bmatrix}$ and U_1 is invertible, then $X = U_2 U_1^{-1}$ 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 4 and 5 contain examples with scalable size where, in Section 5, the user can also choose one or several parameters.
All presented examples may be generated by the FORTRAN 77 subroutine CAREX.F (see Appendix A) and the MATLAB[1] function `carex.m` (see Appendix B). Appendix C describes how to obtain the software.
The description of each example contains a table with some of the system properties. This information is summarized in Appendix D. 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 [15, 20, 38] 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 20. If no analytical solution is available, we computed approximations by the multishift algorithm [1] and the Schur vector method [27]. If possible, these approximations were refined by Newton's method [25] 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 20, 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.

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,

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

[1] MATLAB is a trademark of The MathWorks, Inc.

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 [14] and the estimates and bounds given by Kenney and Hewer [23]. 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 [14, 23] is defined by

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

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

$$\begin{aligned} K_U &= \frac{\|Z_0\|\|Q\| + 2\sqrt{\|Z_0\|\|Z_2\|}\|A\| + \|Z_2\|\|G\|}{\|X\|}, \\ K_L &= \frac{\|Z_0\|\|Q\| + 2\|Z_1\|\|A\| + \|Z_2\|\|G\|}{\|X\|}. \end{aligned}$$

Then one can prove (see [23]) 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 [23] a more refined lower bound is given which in some cases is closer to K_U.)

2 Parameter-free problems of fixed size

Example 1 [27, Example 1]

n	m	p	parameter	$\|H\|$	$\kappa(H)$	$\|X\|$	$\kappa(X)$	K_U
2	1	2	–	2.41	5.83	3.00	3.00	5.04

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 2 [27, Example 2]

n	m	p	parameter	$\|H\|$	$\kappa(H)$	$\|X\|$	$\kappa(X)$	K_U
2	1	2	–	16.16	257.05	31.39	∞	52.59

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 3–5 may be found in [18].

Example 3 [7]

n	m	p	parameter	$\|H\|$	$\kappa(H)$	$\|X\|$	$\kappa(X)$	K_U
4	2	4	–	7.82	55.78	6.12	215.28	21.90

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 4 [10]

n	m	p	parameter	$\|H\|$	$\kappa(H)$	$\|X\|$	$\kappa(X)$	K_U
8	2	8	–	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

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

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

$$Q = \begin{bmatrix} 1 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0.1 \\ 0 & 1 & 0 & 0 & 0.1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0.5 & 9 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0.5 & 0.1 & 0 & 0 & 0.1 & 0 & 0 & 0 \\ 0 & 0 & 0.5 & 0 & 0 & 0.1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.1 & 0 \\ 0.1 & 0 & 0 & 0 & 0 & 0 & 0 & 0.1 \end{bmatrix}, \qquad R = I_2.$$

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

Example 5 [34]

n	m	p	parameter	$\|H\|$	$\kappa(H)$	$\|X\|$	$\kappa(X)$	K_U
9	3	9	–	216.70	3.39×10^3	2.73	1.10×10^3	850.39

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 identiy matrices of size 9 and 3, respectively.

Example 6 [17]

n	m	p	parameter	$\|H\|$	$\kappa(H)$	$\|X\|$	$\kappa(X)$	K_U
30	3	5	–	1.44×10^8	1.50×10^{10}	3.56×10^3	∞	3.73×10^9

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 sytem 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

$$A^{(1)} =$$

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

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

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

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

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

The output of the system is given by

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

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 $O(10^2)$; all of them are of multiplicity one and do not appear in any kind of clusters. Thus, no numerical problems should be expected from the eigenvalue distribution.

3 Parameter-dependent problems of fixed size

Example 7 [4, Example 1]

n	m	p	parameter	$\|H\|$	$\kappa(H)$	$\|X\|$	$\kappa(X)$	K_U
2	1	1	$\varepsilon = 1$	2.95	2.54	2.45	12.91	2.57
			$\underline{\varepsilon = 10^{-6}}$	2.96	5.20	2.00×10^{12}	8.00×10^{12}	3.00

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 8 [4, Example 3]

n	m	p	parameter	$\|H\|$	$\kappa(H)$	$\|X\|$	$\kappa(X)$	K_U
2	2	1	$\varepsilon = 1.0$	1.01×10^4	6.78×10^7	9.88×10^3	4.16×10^3	54.42
			$\underline{\varepsilon = 10^{-8}}$	1.01×10^6	1.46×10^8	9.30×10^3	9.38×10^6	6.69×10^9

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 9 [24, Example 2]

n	m	p	parameter	$\|H\|$	$\kappa(H)$	$\|X\|$	$\kappa(X)$	K_U
2	1	2	$\varepsilon = 1$	1.62	2.62	2.73	3.73	4.15
			$\underline{\varepsilon = 10^6}$	1.00×10^6	1.00×10^6	1.41×10^3	2.00×10^6	8.66×10^5
			$\varepsilon = 10^{-6}$	1.00	1.00×10^{12}	1.00×10^6	1.00×10^6	5.00×10^{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 10 [6]

n	m	p	parameter	$\|H\|$	$\kappa(H)$	$\|X\|$	$\kappa(X)$	K_U
2	2	2	$\varepsilon = 1$	3.16	2.24	6.16	2.55	2.45
			$\underline{\varepsilon = 10^{-7}}$	2.56	1.28×10^{14}	4.00	1.66×10^{7}	3.76×10^{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 11 [22]

n	m	p	parameter	$\|H\|$	$\kappa(H)$	$\|X\|$	$\kappa(X)$	K_U
2	1	2	$\varepsilon = 1$	10.60	24.15	2.62	6.85	8.11
			$\underline{\varepsilon = 0}$	15.44	58.42	2.62	6.85	∞

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 [30]. 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 [23] is not defined.

Example 12 [36]

n	m	p	parameter	$\|H\|$	$\kappa(H)$	$\|X\|$	$\kappa(X)$	K_U
3	3	3	$\varepsilon = 1$	3.16	2.24	6.16	2.55	2.45
			$\underline{\varepsilon = 10^{6}}$	3.54×10^{6}	3.54	6.00×10^{12}	3.00	2.73

This example is constructed as follows. Let

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

and

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

Then

$$A = VA_0V, \qquad G = \frac{1}{\varepsilon} I_3, \qquad Q = VQ_0V.$$

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 [24, 33].

Example 13 [16]

n	m	p	parameter	$\|H\|$	$\kappa(H)$	$\|X\|$	$\kappa(X)$	K_U
4	1	2	$\varepsilon = 1$	1.63	116.85	22.29	1.03×10^3	93.04
			$\underline{\varepsilon = 10^{-6}}$	1.00×10^{12}	5.72×10^{13}	13.17	9.10×10^8	4.06×10^{13}

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

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

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 14 [4, Example 2]

n	m	p	parameter	$\|H\|$	$\kappa(H)$	$\|X\|$	$\kappa(X)$	K_U
4	1	1	$\varepsilon = 1$	4.45	9.90	11.84	140.08	36.43
			$\underline{\varepsilon = 10^{-6}}$	4.24	17.94	1.00	1.00	1.00×10^{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.

4 Examples of scalable size without parameters

Example 15 [27, Example 4], [5]

n	m	p	parameter	$\|H\|$	$\kappa(H)$	$\|X\|$	$\kappa(X)$	K_U
$2N-1$	N	$N-1$	$N=5$	10.00	41.01	12.24	∞	11.10
			$\underline{N=20}$	10.00	425.44	28.80	∞	50.94

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 $n = 2N-1$.

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

where

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

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 16 [27, Example 5]

n	m	p	parameter	$\|H\|$	$\kappa(H)$	$\|X\|$	$\kappa(X)$	K_U
n	n	n	$n = 8$	4.12	4.12	1.00	8.12	5.00
			$\underline{n = 64}$	4.12	4.12	1.00	8.12	5.00

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 [12, 13, 40], multishift QR [1, 37]), 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 $i = 0, \dots, n-1$,

$$x_i = \frac{1}{n} \sum_{k=0}^{n-1} \left\{ -2 + 2\cos\left(\frac{2\pi k}{n}\right) + \sqrt{5 - 8\cos\left(\frac{2\pi k}{n}\right) + 4\cos^2\left(\frac{2\pi k}{n}\right)} \right\} \omega_n^{ik} \tag{6}$$

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 [27]. Since the imaginary part of the sum in (6) adds to 0, ω_n^{ik} may be replaced by $\cos\left(\frac{2\pi k i}{n}\right)$ for keeping computations real.

5 Parameter-dependent examples of scalable size

Example 17 [27, Example 6]

n	m	p	parameter	$\|H\|$	$\kappa(H)$	$\|X\|$	$\kappa(X)$	K_U
n	1	1	$\underline{n = 21,\ q = r = 1.0}$	1.00	1.00	2.41×10^9	∞	1.26×10^9
			$n = 21,\ q = r = 100.0$	100.0	1.00×10^4	2.41×10^{11}	∞	1.26×10^9

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 [27].)

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

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 [27]). 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 18 [39]

n	m	p	parameter	$\\|H\\|$	$\kappa(H)$	$\\|X\\|$	$\kappa(X)$	K_U
n	1	1	$n = 20$, $a = 0.05$, $b = c = 0.1$, $[\beta_1, \beta_2] = [0.1, 0.5]$, $[\gamma_1, \gamma_2] = [0.1, 0.5]$.	260.22	547.34	1.02×10^{-4}	∞	495.49
			$n = 100$, $a = 0.01$, $b = c = 1.0$, $[\beta_1, \beta_2] = [0.2, 0.3]$, $[\gamma_1, \gamma_2] = [0.2, 0.3]$.	1.22×10^3	1.40×10^5	7.10×10^{-4}	∞	1.03×10^4

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

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

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 [39]. 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 [19, 29]. In this type of problems, the dynamical system is given in terms of a second-order differential equation

$$M\ddot{z} + L\dot{z} + Kz = Du \tag{7}$$

and an associated output

$$y = Nz + P\dot{z} \tag{8}$$

or alternatively

$$\tilde{y} = \begin{bmatrix} N & 0\\ 0 & P\end{bmatrix}\begin{bmatrix} z\\ \dot{z}\end{bmatrix} \tag{9}$$

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

$$
\begin{aligned}
\dot{x} &= \begin{bmatrix} 0 & I\\ -M^{-1}K & -M^{-1}L\end{bmatrix} x + \begin{bmatrix} 0\\ M^{-1}D\end{bmatrix} u \qquad (10)\\
y &= [\,N \quad P\,]\,x, \qquad (11)
\end{aligned}
$$

or, with (9),

$$\tilde{y} = \begin{bmatrix} N & 0\\ 0 & P\end{bmatrix} x. \tag{12}$$

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 19 [21, Example 3]

n	m	p	parameter	$\|H\|$	$\kappa(H)$	$\|X\|$	$\kappa(X)$	K_U
2ℓ	2	2ℓ	$\underline{\ell = 30}$, $\underline{\mu = 4.0,\ \delta = 4.0,\ \kappa = 1.0}$	2.19	1.14×10^5	218.02	447.98	1.52×10^3

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

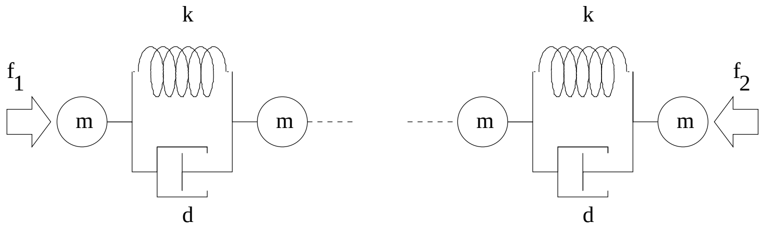

Figure 1: Coupled Spring Experiment ($k \sim \kappa$, $m \sim \mu$, $d \sim \delta$)

this problem, the matrices in (7), (9) are

$$
\begin{aligned}
M &= \mu I_\ell, \qquad L = \delta I_\ell, \qquad N = P = I_\ell,\\
K &= \kappa \begin{bmatrix} 1 & -1 & 0 & \dots & 0 & 0 \\ -1 & 2 & -1 & \dots & 0 & 0 \\ 0 & -1 & 2 & \dots & 0 & 0 \\ \vdots & & \ddots & \ddots & \ddots & \dots \\ 0 & 0 & \dots & -1 & 2 & -1 \\ 0 & 0 & \dots & 0 & -1 & 1 \end{bmatrix}, \qquad D = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ \vdots & \vdots \\ \vdots & \vdots \\ 0 & 0 \\ 0 & -1 \end{bmatrix}.
\end{aligned}
$$

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 20 [35, 41]

n	m	p	parameter	$\|H\|$	$\kappa(H)$	$\|X\|$	$\kappa(X)$	K_U
$2\ell - 1$	ℓ	ℓ	$\underline{\ell = 211}$	4.06×10^{11}	5.50×10^{14}	7.21×10^7	1.40×10^{22}	3.00×10^8

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 (7) and (8) are given as

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

$$
\begin{aligned}
L &= \begin{bmatrix} \delta_1+\gamma_1 & -\gamma_1 & & & \\ -\gamma_1 & \gamma_1+\delta_2+\gamma_2 & -\gamma_2 & & \\ & \ddots & \ddots & \ddots & \\ & & -\gamma_{\ell-2} & \gamma_{\ell-2}+\delta_{\ell-1}+\gamma_{\ell-1} & -\gamma_{\ell-1} \\ & & & -\gamma_{\ell-1} & \gamma_{\ell-1}+\delta_\ell \end{bmatrix}, \qquad D = I_\ell, \\
N &= \begin{bmatrix} 0 & 0 & & \\ \kappa_1 & -\kappa_1 & & \\ & \ddots & \ddots & \\ & & \kappa_{\ell-1} & -\kappa_{\ell-1} \end{bmatrix}, \qquad P = \begin{bmatrix} 1 & 0 & & \\ \gamma_1 & -\gamma_1 & & \\ & \ddots & \ddots & \\ & & \gamma_{\ell-1} & -\gamma_{\ell-1} \end{bmatrix}.
\end{aligned}
$$

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., [42].

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 (10) 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

$$
\begin{aligned}
\hat{A} &= T^{-1}\begin{bmatrix} 0 & I \\ -M^{-1}K & -M^{-1}L \end{bmatrix} T = \begin{bmatrix} -\hat{T}^{-1}M^{-1}L\hat{T} & -\hat{T}^{-1}M^{-1}L\hat{T} \\ I & 0 \end{bmatrix}, \\
\hat{B} &= T^{-1}\begin{bmatrix} 0 \\ M^{-1} \end{bmatrix} = \begin{bmatrix} \hat{T}^{-1}M^{-1} \\ 0 \end{bmatrix}, \\
\hat{C} &= [\,N \;\; P\,]T = [\,P\hat{T} \;\; N\hat{T}\,].
\end{aligned} \tag{13}
$$

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 (13) 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 [26] 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 and MATLAB (see Appendices A and B).

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 [15] 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.

6 Acknowledgements

We wish to thank V. Sima for pointing out some sources of industrial examples to us and A. Varga for fruitful discussions about the design of the FORTRAN 77 subroutine CAREX.F. Furthermore, we wish to thank W. Lang and T. Penzl for providing Example 20.

A The FORTRAN 77 subroutine CAREX.F

This is the prolog of a FORTRAN 77 subroutine for generating all presented examples. The subroutine was documented according to standards for SLICOT[2] [32]. For some of the examples, CAREX reads the data from data files delivered together with CAREX.F. These are Examples 3–6 and 20. The corresponding data files are CAREX3.DAT, CAREX4.DAT, CAREX5.DAT, CAREX6.DAT, and CAREX20.DAT.
A data file for Example 20 may be supplied by the user. In this case, on entry to CAREX, `N` must contain the integer ℓ, i.e., the order of the second order model (7) 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 and BLAS [2], CAREX calls the subroutines SP2SY and SY2SP which are used to convert symmetric matrices from general storage mode to packed storage mode and vice versa. These subroutines are provided together with CAREX.F and the necessary data files. If you have no access to LAPACK and BLAS, please contact the authors.

```
      SUBROUTINE CAREX(NO, N, M, P, NPAR, DPARAM, DATAF, A, LDA, B,
     1                 LDB, C, LDC, G, LDG, Q, LDQ, X, LDX, NOTE, STORE,
     2                 FORM, RWORK, IERR)
C
C     PURPOSE
C
C     To generate the benchmark examples for the numerical solution of
C     continuous-time algebraic Riccati equations as presented in [1]
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 Q0 C .
C
C     Here, C is P-by-N, Q0 P-by-P, B N-by-M, and R M-by-M, where Q0
C     and R are symmetric. In linear-quadratic control problems,
C     usually Q0 is positive semidefinite and R positive definite.
C
C     ARGUMENT LIST
C       ARGUMENTS IN
C
```

[2] **S**ubroutine **LI**brary in **CO**ntrol and Systems **T**heory

```
C         NO - INTEGER.
C              The number of the benchmark example to generate according
C              to [1].
C
C          N - INTEGER.
C              This integer determines the actual state dimension, i.e.,
C              the order of the matrix A as follows:
C              N = number of vehicles for Example 15.
C              N = order of matrix A for Examples 16-18.
C              N = dimension of second-order system, i.e., order of
C                  stiffness matrix for Examples 19 and 20. The order of
C                  the output matrix A is 2*N for Example 19 and 2*N-1
C                  for Example 20.
C              N is fixed for the examples of Sections 2 and 3 of [1],
C              i.e., currently Examples 1-14.
C              NOTE that N is overwritten for Examples 1-14 and for the
C              other examples if N is set by default.
C
C       M, P - INTEGER.
C              M is the number of columns in the matrix B from (I) (in
C              control problems, the number of inputs of the system).
C              P is the number of rows in the matrix C from (II) (in
C              control problems, the number of outputs of the system).
C              Currently, M and P are fixed or determined by N for all
C              examples and thus are not referenced on input.
C              NOTE that M and P are overwritten.
C
C       NPAR - INTEGER.
C              Number of input parameters supplied by the user.
C              Examples 1-6 (Section 3 of [1]) have no parameters.
C              Examples 7-14 (Section 4 of [1]) each have one DOUBLE
C              PRECISION parameter which may be supplied in DPARAM(1).
C              Examples 15,16 have one INTEGER parameter which determines
C              the size of the problem. This parameter may be supplied in
C              the input argument N.
C              Examples 17-19 have one INTEGER (supplied in N) and
C              several DOUBLE PRECISION parameters (supplied in DPARAM).
C              If for Example 20 user supplied data is to be used, i.e.,
C              NPAR > 0, the INTEGER input argument N must contain an
C              INTEGER l (as described in [1]) and the CHARACTER input
C              argument DATAF must contain the name of a data file.
C              If the input value of NPAR is less than the number of
C              parameters of the Example NO (according to [1]), the
C              missing parameters are set by default.
C
C     DPARAM - DOUBLE PRECISION array of DIMENSION at least 7.
C              Double precision parameter vector. For explanation of the
```

```
C              parameters see [1].
C              DPARAM(1) defines the parameters 'epsilon' for the
C              examples in Section 3 (NO = 7,...,14), the parameter 'q'
C              for NO = 17, 'a' for NO = 18, and 'mu' for NO = 19.
C              DPARAM(2) defines parameters 'r' for NO = 17, 'b' for
C              NO = 18, and 'delta' for NO = 19.
C              DPARAM(3) defines 'c' for NO = 18 and 'kappa' for NO = 19.
C              DPARAM(4) - DPARAM(7) are only used to generate Example
C              18 and define in consecutive order the intervals
C              ['beta_1', 'beta_2'],  ['gamma_1', 'gamma_2'].
C              If NPAR is smaller than the number of used parameters in
C              Example NO (as described in [1]), default values are
C              used and returned in corresponding components of DPARAM.
C              NOTE that those entries of DPARAM are overwritten which
C              are used to generate the example but were not supplied by
C              the user.
C
C      DATAF - CHARACTER*255.
C              The name of a data file supplied by the user. In the
C              current version, only Example 20 allows a user-defined
C              data file. This file must contain consecutively DOUBLE
C              PRECISION vectors mu, delta, gamma, and kappa. The length
C              of these vectors is determined by the input value for N.
C              If on entry N = l, then mu, delta must each contain l,
C              gamma, kappa each l-1 DOUBLE PRECISION values.
C
C        LDA - INTEGER.
C              The leading dimension of array A as declared in the
C              calling program.
C              LDA .GE. N  where N is the order of the matrix A, i.e.,
C              the output value of the integer N.
C
C        LDB - INTEGER.
C              The leading dimension of array B as declared in the
C              calling program.
C              LDB .GE. N  (output value of N).
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 either defined by default or
C              depends upon N. (For all examples, P .LE. N, where N is
C              the output value of the argument N.)
C
C        LDG - INTEGER.
C              If full storage mode is used for G, i.e., STORE = 'F'
C              or 'f', then G is stored like a 2-dimensional array
```

```
C              with leading dimension LDG. If packed symmetric storage
C              mode is used, then LDG is not referenced.
C              LDG .GE. N  if STORE = 'F' or 'f'.
C
C        LDQ - INTEGER.
C              If full storage mode is used for Q, i.e., STORE = 'F'
C              or 'f', then Q is stored like a 2-dimensional array
C              with leading dimension LDQ. If packed symmetric storage
C              mode is used, then LDQ is not referenced.
C              LDQ .GE. N  if STORE = 'F' or 'f'.
C
C        LDX - INTEGER.
C              The leading dimension of array X as declared in the
C              calling program.
C              LDX .GE. N.
C
C       ARGUMENTS OUT
C
C          N - INTEGER.
C              The order of matrix A.
C
C          M - INTEGER.
C              The number of columns of matrix B from (I), rank(G) <= M.
C
C          P - INTEGER.
C              The number of rows of matrix C from (II), rank(Q) <= P.
C
C     DPARAM - DOUBLE PRECISION array of DIMENSION at least 7.
C              Double precision parameter vector. For explanation of the
C              parameters see [1].
C              DPARAM(1) defines the parameters 'epsilon' for the
C              examples in Section 3 (NO = 7,...,14), the parameter 'q'
C              for NO = 17, 'a' for NO = 18, and 'mu' for NO = 19.
C              DPARAM(2) defines 'r' for NO = 17, 'b' for NO = 18, and
C              'delta' for NO = 19.
C              DPARAM(3) defines 'c' for NO = 18 and 'kappa' for NO = 19.
C              DPARAM(4) - DPARAM(7) are only used to generate Example
C              18 and define in consecutive order the intervals
C              ['beta_1', 'beta_2'],  ['gamma_1', 'gamma_2'].
C
C          A - DOUBLE PRECISION array of DIMENSION (LDA,N).
C              The leading N by N part of this array contains the
C              coefficient matrix A of the ARE.
C
C          B - DOUBLE PRECISION array of DIMENSION (LDB,M).
C              If (FORM .EQ. 'F' or 'f' or 'G' or 'g') then array B
C              contains the matrix B of the factored form (I) of G.
```

```
C             Otherwise, B is used as workspace.
C
C         C - DOUBLE PRECISION array of DIMENSION (LDC,N).
C             If (FORM .EQ. 'F' or 'f' or 'Q' or 'q') then array C
C             contains the matrix C of the factored form (II) of Q.
C             Otherwise, C is used as workspace.
C
C         G - DOUBLE PRECISION array of DIMENSION at least ng.
C             If STORE = 'F' or 'f'               then ng = LDG*N.
C             If STORE = 'U' or 'u' or 'L' or 'l' then ng = N*(N+1)/2.
C             If (FORM .EQ. 'P' or 'p' or 'Q' or 'q'), then array G
C             contains the coefficient matrix G of the ARE.
C             If (FORM .EQ. 'F' or 'f' or 'G' or 'g'), then array G
C             contains the 'control weighting matrix' R of G's factored
C             form as in (I).
C             The symmetric matrix contained in array G is stored
C             according to MODE PARAMETER STORE.
C
C         Q - DOUBLE PRECISION array of DIMENSION at least nq.
C             If STORE = 'F' or 'f'               then nq = LDQ*N.
C             If STORE = 'U' or 'u' or 'L' or 'l' then nq = N*(N+1)/2.
C             If (FORM .EQ. 'P' or 'p' or 'G' or 'g'), then array Q
C             contains the coefficient matrix Q of the ARE.
C             If (FORM .EQ. 'F' or 'f' or 'Q' or 'q'), then array Q
C             contains the 'output weighting matrix' Q0 of Q's factored
C             form as in (II).
C             The symmetric matrix contained in array Q is stored
C             according to MODE PARAMETER STORE.
C
C         X - DOUBLE PRECISION array of DIMENSION (LDX,N).
C             If an exact solution is available (NO = 1,2,7,9-12,16),
C             then the leading N-by-N part of this array contains
C             the solution matrix X. Otherwise, X is not referenced.
C
C      NOTE - CHARACTER*70.
C             String containing short information about the chosen
C             example.
C
C     WORK SPACE
C
C     RWORK - DOUBLE PRECISION array of DIMENSION at least N*MAX(4,N).
C
C     MODE PARAMETERS
C
C      FORM - CHARACTER.
C             Specifies the output format of the examples, i.e., if Q
C             and G are returned in factored form (I),(II), or not.
```

```
C              FORM = 'P' or 'p': The matrices Q and G are returned.
C              FORM = 'G' or 'g': G is returned in factored form, i.e.,
C                                 B and R from (I) are returned, array Q
C                                 contains the coefficient matrix Q.
C              FORM = 'Q' or 'q': Q is returned in factored form, i.e.,
C                                 C and Q0 from (II) are returned, array
C                                 G contains the coefficient matrix G.
C              FORM = 'F' or 'f': Q and G are given in factored form,
C                                 i.e., B, R, C, and Q0 from (I) and (II)
C                                 are returned.
C              Otherwise, CAREX returns with an error.
C              NOTE that for factored forms, output array G contains R
C              from (I) whereas output array Q contains Q0 from (II).
C
C       STORE - CHARACTER.
C              Specifies the storage mode for arrays G and Q.
C              STORE = 'F' or 'f': Full symmetric matrices are stored in
C                                  G and Q, i.e., the leading N-by-N
C                                  part of these arrays each contain a
C                                  symmetric matrix.
C              STORE = 'L' or 'l': Matrices contained in arrays G and Q
C                                  are stored in lower packed mode,
C                                  i.e., the lower triangle of a
C                                  symmetric n-by-n matrix is stored by
C                                  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              STORE = 'U' or 'u': Matrices contained in arrays G and Q
C                                  are stored in upper packed mode,
C                                  i.e., the upper triangle of a
C                                  symmetric n-by-n matrix is stored by
C                                  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, CAREX returns with an error.
C
C     ERROR INDICATOR
C
C       IERR - INTEGER.
C              Unless the routine detects an error (see next section),
C              IERR contains 0 on exit.
C
C
C     WARNINGS AND ERRORS DETECTED BY THE ROUTINE
C
C     IERR = 1 : (NO .LT. 1) or (NO .GT. NEX).
C                (NEX = number of available examples.)
```

```
C     IERR = 2 : (N .LT. 1) or (N .GT. LDA) or (N .GT. LDB) or
C                (N. GT. LDX) or (P .GT. LDC).
C     IERR = 3 : MODE PARAMETER STORE had an illegal value on input.
C     IERR = 4 : MODE PARAMETER FORM had an illegal value on input.
C     IERR = 5 : Data file could not be opened or had wrong format.
C     IERR = 6 : Division by zero.
C     IERR = 7 : G can not be computed as in (I) due to a singular R
C                matrix.
C
C     REFERENCE
C
C     [1] 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     [2] E. ANDERSON ET AL.
C         LAPACK Users' Guide, second edition.
C         SIAM, Philadelphia, PA (1994).
C
C     CONTRIBUTOR
C
C     Peter Benner and Volker Mehrmann (TU Chemnitz-Zwickau)
C     Alan J. Laub (University of California, Santa Barbara)
C
C     KEYWORDS
C
C     algebraic Riccati equation, Hamiltonian matrix
C
C     REVISIONS
C
C     1995, January 18.
C     1995, October 12.
C
C***********************************************************************
```

B The MATLAB function carex.m

The prolog of the MATLAB function `carex.m` is listed below. For all listed examples, it is possible to return the matrices A, G, Q and the factors B, R, C, $\tilde{Q}$ = `Q0`. If the solution is not available, the output matrix `X` contains an empty matrix. Otherwise, X is returned as well as the upper and lower bounds K_U and K_L for K_{CARE} computed by a MATLAB function `carecond`. Note that `carecond` uses the Lyapunov solver `lyap` from the MATLAB CONTROL TOOLBOX.

For Examples 6 and 20, we provide data files `carex6.mat` and `carex20.mat`, respectively. These files contain the necessary data for the corresponding examples in internal MATLAB format. For Example 20, the user may provide a data file containing an integer `l` and column vectors `mu`, `delta`, `gamma`, and `kappa` of appropriate size such that in a MATLAB environment, the command

```
>> load <filename>
```

would generate the necessary information. The name of this file can then serve as input to `carex` which generates the system matrices according to this data.

```
function [A,G,Q,X,parout,B,R,C,Q0]=carex(index,parin)
%CAREX
%
% Test examples for the continuous-time algebraic Riccati equation
%
%  (CARE)   0 = Q  +  A' X  +  X A  -  X G X.
%
% Here, A, G, Q, X are n-by-n matrices,  G and Q are symmetric.
% G, Q may be in factored form  G = B R^(-1) B',    Q = C' Q0 C.
% Then  B is n-by-m, R m-by-m, C p-by-n, and Q0 is p-by-p. The
% corresponding Hamiltonian matrix is defined as
%
%                          ( A   -G )   (     A    -B/R B')
%      H := Ham(A,G,Q) := (        ) = (                 ).
%                          (-Q   -A')   (-C' Q0 C    -A' )
%
% Input:
%  - index : number of example to generate, indices refer to example
%            numbers in [1].
%  - parin : input parameters (optional, default values given in [1]).
%      For Example number
%      +  1- 6: not referenced ([1], Section 2).
%      +  7-14: parin(1) = real-valued scalar, ([1], Section 3).
%      + 15   : parin(1) = N,  n = 2*N-1,  m = N,  p = N-1.
%      + 16-18: parin(1) = n = problem size = size of A, G, Q, X.
%      + 17   : parin(2) = Q0 (real scalar).
%               parin(3) = R  (real scalar).
%      + 18   : parin(2:8) = real-valued scalars, where
%               parin(2:4) = [a, b, c].
```

```
%               parin(5:6) = [beta_1,beta_2].
%               parin(7:8) = [gamma_1,gamma_2].
%       + 19   : parin(1) = l = number of springs (n = 2l).
%               parin(2:4) = [mu, lambda, kappa].
%       + 20   : parin = name of data file containing the number of masses
%                       l (n = 2*l-1), and the vectors mu, delta, gamma,
%                       kappa described in [1].
%
% Output:
%  - A, G, Q : system matrices from CARE.
%  - X       : exact stabilizing solution of CARE (if available).
%              If an exact solution is not available, the empty matrix
%              is returned.
%              For Example 17,  X = X(1,n) = X(n,1) = sqrt(Q0*R), which
%              is the only available information.
%  - parout  : Vector with system properties,
%              parout(1:3) = [n, m, p]
%              parout(4)   = norm(H) = 2-norm of H = Ham(A,G,Q)
%              parout(5)   = 2-norm condition number of H.
%              The following parameters are only returned if an analytical
%              solution of the CARE is available:
%              parout(6)   = 2-norm of X
%              parout(7)   = 2-norm condition number of X
%              parout(8:9) = [KU,KL] = upper and lower bound for condition
%                            number of CARE (see [2]) (not available for
%                            Example 11).
%
%  - B,R,C,Q0: optional output matrices if factored form is required.
%
% References:
%
% [1] P.BENNER, A.J. LAUB, V. MEHRMANN: 'A Collection of Benchmark
%     Examples for the Numerical Solution of Algebraic Riccati
%     Equations I: Continuous-Time Case', Tech. Report SPC 95_23,
%     Fak. f. Mathematik, TU Chemnitz-Zwickau (Germany), October 1995.
% [2] C. KENNEY, G. HEWER: 'The sensitivity of the algebraic and
%     differential Riccati equations', SIAM J. Control. Optim.,
%     vol. 28 (1990), pp.50-69.
```

C How to obtain the software

The codes corresponding to this paper may be obtained via anonymous ftp at TU Chemnitz-Zwickau. Proceed as follows.

> `ftp ftp.tu-chemnitz.de`
> Name: `anonymous`
> Password: *your complete e-mail address*
> `cd pub/Local/mathematik/Benner`

Observe the capital "L" in `Local` !
Now get the compressed FORTRAN 77 subroutines CAREX.F, SP2SY.F, SY2SP.F, a sample Makefile and program as well as the data files CAREX3.DAT, CAREX4.DAT, CAREX5.DAT, CAREX6.DAT, CAREX20.DAT by

> `get carex_f.tar.Z`

or the compressed MATLAB functions `carex.m`, `carecond.m` and data files `carex6.mat`, `carex20.mat` by

> `get carex_m.tar.Z`

After exiting ftp, extracting the MATLAB codes and data files is achieved by the following commands:

> `uncompress carex_m.tar.Z`
> `tar xf carex_m.tar`

Analogously, the FORTRAN 77 codes and corresponding data files are obtained by

> `uncompress carex_f.tar.Z`
> `tar xf carex_f.tar`

In both cases, the command `tar xf` creates a directory containing all required files. For `carex_m.tar.Z`, this directory is called `carex_m` and for `carex_f.tar.Z`, it will be `carex_f`. If there occur any problems obtaining or running the codes please contact one of the authors.

D Reference table

Table 1 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 ("–").

Table 1

no.	n	m	p	default	X^*	$\|H\|$	$\kappa(H)$	$\|X\|$	$\kappa(X)$	K_U
1	2	1	2	–	+	2.41	5.83	3.00	3.00	5.04
2	2	1	2	–	+	16.16	257.05	31.39	∞	52.59
3	4	2	4	–	–	7.82	55.78	6.12	215.28	21.90
4	8	2	8	–	–	3.41	305.91	4.75	1.28×10^3	33.58
5	9	3	9	–	–	216.70	3.39×10^3	2.73	1.10×10^3	850.39
6	30	3	5	–	–	1.44×10^8	1.50×10^{10}	3.56×10^3	∞	3.73×10^9
7	2	1	1	$\varepsilon = 10^{-6}$	+	2.96	5.20	2.00×10^{12}	8.00×10^{12}	3.00
8	2	2	1	$\varepsilon = 10^{-8}$	–	1.01×10^6	1.46×10^8	9.30×10^3	9.38×10^6	6.69×10^9
9	2	1	2	$\varepsilon = 10^6$	+	1.00×10^6	1.00×10^6	1.41×10^3	2.00×10^6	8.66×10^5
10	2	2	2	$\varepsilon = 10^{-7}$	+	2.56	1.28×10^{14}	4.00	1.66×10^7	3.76×10^3
11	2	1	2	$\varepsilon = 0$	+	15.44	58.42	2.62	6.85	∞
12	3	3	3	$\varepsilon = 10^6$	+	3.54×10^6	3.54	6.00×10^{12}	3.00	2.73
13	4	1	2	$\varepsilon = 10^{-6}$	–	1.00×10^{12}	5.72×10^{13}	13.17	9.10×10^8	4.06×10^{13}
14	4	1	1	$\varepsilon = 10^{-6}$	–	4.24	17.94	1.00	1.00	1.00×10^{13}
15	$2N-1$	N	$N-1$	$N = 20$	–	10.00	425.44	28.80	∞	50.94
16	n	n	n	$n = 64$	+	4.12	4.12	1.00	8.12	5.00
17	n	1	1	$n = 21$, $q = r = 1.0$	–	1.00	1.00	2.41×10^9	∞	1.26×10^9
18	n	1	1	$n = 100$, $a = 0.01$, $b = c = 1.0$, $[\beta_1, \beta_2] = [0.2, 0.3]$, $[\gamma_1, \gamma_2] = [0.2, 0.3]$	–	1.22×10^3	1.40×10^5	7.10×10^{-4}	∞	1.03×10^4
19	2ℓ	2	2ℓ	$\ell = 30$, $\mu = \delta = 4.0$, $\kappa = 1.0$	–	2.19	1.14×10^5	218.02	447.98	1.52×10^3
20	$2\ell-1$	ℓ	ℓ	$\ell = 211$	–	4.06×10^{11}	5.50×10^{14}	7.21×10^7	1.40×10^{22}	3.00×10^8

References

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

[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 ed., 1994.

[3] W. 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. Arnold, III and A. Laub, *Generalized eigenproblem algorithms and software for algebraic Riccati equations*, Proc. IEEE, 72 (1984), pp. 1746–1754.

[5] M. Athans, W. 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, Sept. 1967.

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

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

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

[9] ———, *Newton's method with exact line search for solving continuous-time algebraic Riccati equations*, Tech. Report SPC 95_24, Fakultät für Mathematik, Technische Universität Chemnitz–Zwickau, 09107 Chemnitz, FRG, Oct. 1995.

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

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

[12] 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 (1986), pp. 1104–1113.

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

[14] R. Byers, *Numerical condition of the algebraic Riccati equation*, Contemp. Math., 47 (1985), pp. 35–49.

[15] ——, *Solving the algebraic Riccati equation with the matrix sign function*, Linear Algebra Appl., 85 (1987), pp. 267–279.

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

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

[18] Z. Gajić and X. Shen, *Parallel Algorithms for Optimal Control of Large Scale Linear Systems*, Springer-Verlag, London, 1993.

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

[20] J. Gardiner and A. Laub, *A generalization of the matrix-sign-function solution for algebraic Riccati equations*, Internat. J. Control, 44 (1986), pp. 823–832. (see also *Proc. 1985 CDC*, pp. 1233–1235).

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

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

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

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

[25] D. Kleinman, *On an iterative technique for Riccati equation computations*, IEEE Trans. Automat. Control, AC-13 (1968), pp. 114–115.

[26] W. Lang. personal communication, 1995.

[27] A. Laub, *A Schur method for solving algebraic Riccati equations*, IEEE Trans. Automat. Control, AC-24 (1979), pp. 913–921. (see also *Proc. 1978 CDC (Jan. 1979)*, pp. 60-65).

[28] ——, *Invariant subspace methods for the numerical solution of Riccati equations*, in The Riccati Equation, S. Bittanti, A. Laub, and J. Willems, eds., Springer-Verlag, Berlin, 1991, pp. 163–196.

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

[30] A. Linnemann, *Numerische Methoden für lineare Regelungssysteme*, BI Wissenschaftsverlag, Mannheim, 1993.

[31] V. Mehrmann, *The Autonomous Linear Quadratic Control Problem, Theory and Numerical Solution*, no. 163 in Lecture Notes in Control and Information Sciences, Springer-Verlag, Heidelberg, July 1991.

[32] Numerical Algorithms Group, *Implementation and documentation standards for the subroutine library in control and systems theory SLICOT*, Publication NP2032, Numerical Algorithms Group, Eindhoven/Oxford, 1990.

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

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

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

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

[37] A. Raines and D. Watkins, *A class of Hamiltonian–symplectic methods for solving the algebraic Riccati equation*, Linear Algebra Appl., 205/206 (1994), pp. 1045–1060.

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

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

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

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

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

Other titles in the SPC series:

95_1 T. Apel, G. Lube. Anisotropic mesh refinement in stabilized Galerkin methods Januar 1995.

95_2 M. Meisel, A. Meyer. Implementierung eines parallelen vorkonditionierten Schur-Komplement CG-Verfahrens in das Programmpaket FEAP. Januar 1995.

95_3 S. V. Nepomnyaschikh. Optimal multilevel extension operators. January 1995

95_4 M. Meyer. Grafik-Ausgabe vom Parallelrechner für 3D-Gebiete. Januar 1995

95_5 T. Apel, G. Haase, A. Meyer, M. Pester. Parallel solution of finite element equation systems: efficient inter-processor communication. Februar 1995

95_6 U. Groh. Ein technologisches Konzept zur Erzeugung adaptiver hierarchischer Netze für FEM-Schemata. Mai 1995

95_7 M. Bollhöfer, C. He, V. Mehrmann. Modified block Jacobi preconditioners for the conjugate gradient method. Part I: The positive definit case. January 1995

95_8 P. Kunkel, V. Mehrmann, W. Rath, J. Weickert. GELDA: A Software Package for the Solution of General Linear Differential Algebraic Equation. February 1995

95_9 H. Matthes. A DD preconditioner for the clamped plate problem. February 1995

95_10 G. Kunert. Ein Residuenfehlerschätzer für anisotrope Tetraedernetze und Dreiecksnetze in der Finite-Elemente-Methode. März 1995

95_11 M. Bollhöfer. Algebraic Domain Decomposition. March 1995

95_12 B. Nkemzi. Partielle Fourierdekomposition für das lineare Elastizitätsproblem in rotationssymmetrischen Gebieten. März 1995

95_13 A. Meyer, D. Michael. Some remarks on the simulation of elasto-plastic problems on parallel computers. March 1995

95_14 B. Heinrich, S. Nicaise, B. Weber. Elliptic interface problems in axisymmetric domains. Part I: Singular functions of non-tensorial type. April 1995

95_15 B. Heinrich, B. Lang, B. Weber. Parallel computation of Fourier-finite-element approximations and some experiments. May 1995

95_16 W. Rath. Canonical forms for linear descriptor systems with variable coefficients. May 1995

95_17 C. He, A. J. Laub, V. Mehrmann. Placing plenty of poles is pretty preposterous. May 1995

95_18 J. J. Hench, C. He, V. Kučera, V. Mehrmann. Dampening controllers via a Riccati equation approach. May 1995

95_19 M. Meisel, A. Meyer. Kommunikationstechnologien beim parallelen vorkonditionierten Schur-Komplement CG-Verfahren. Juni 1995

95_20 G. Haase, T. Hommel, A. Meyer and M. Pester. Bibliotheken zur Entwicklung paralleler Algorithmen. Juni 1995.

95_21 A. Vogel. Solvers for Lamé equations with Poisson ratio near 0.5. June 1995.

95_22 P. Benner, A. J. Laub, V. Mehrmann. A collection of benchmark examples for the numerical solution of algebraic Riccati equations I: Continuous-time case. October 1995.

95_23 P. Benner, A. J. Laub, V. Mehrmann. A collection of benchmark examples for the numerical solution of algebraic Riccati equations II: Discrete-time case. December 1995.

95_24 P. Benner, R. Byers. Newton's method with exact line search for solving the algebraic Riccati equation. October 1995.

95_25 P. Kunkel, V. Mehrmann. Local and Global Invariants of Linear Differential-Algebraic Equations and their Relation. July 1995.

95_26 C. Israel. NETGEN69 - Ein hierarchischer paralleler Netzgenerator. August 1995.

95_27 M. Jung. Parallelization of multi-grid methods based on domain decomposition ideas. November 1995.

95_28 P. Benner, H. Faßbender. A restarted symplectic Lanczos method for the Hamiltonian eigenvalue problem. October 1995.

95_29 G. Windisch. Exact discretizations of two-point boundary value problems. October 1995.

95_30 S. V. Nepomnyaschikh. Domain decomposition and multilevel techniques for preconditioning operators. November 1995.

95_31 H. Matthes. Parallel preconditioners for plate problems. November 1995.

95_32 V. Mehrmann, H. Xu. An analysis of the pole placement problem. I. The single input case. November 1995.

95_33 Th. Apel. SPC-PM Po 3D — User's manual. December 1995.

95_34 Th. Apel, F. Milde, M. Theß. SPC-PM Po 3D — Programmer's manual. December 1995.

95_35 S. A. Ivanov, V. G. Korneev. On the preconditioning in the domain decomposition technique for the p-version finite element method. Part I. December 1995.

95_36 S. A. Ivanov, V. G. Korneev. On the preconditioning in the domain decomposition technique for the p-version finite element method. Part II. December 1995.

95_37 V. Mehrmann, N. K. Nichols. Mixed output feedback for descriptor systems. December 1995.

Some papers can be accessed via anonymous ftp from server `ftp.tu-chemnitz.de`,
directory `pub/Local/mathematik/SPC`. (Note the capital L in Local!)
The complete list of current and former preprints is available via
`http://www.tu-chemnitz.de/~pester/sfb/spc95pr.html`.