$\mathcal{H}_\infty$ and $\mathcal{H}_2$ Optimization Toolbox in SLICOT [1]

Da-Wei Gu[2], Petko Hr. Petkov[3] and Mihail M. Konstantinov[4]

September 1999

[1]This paper 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/pub/WGS/REPORTS/SLWN1999-12.ps.Z*

[2]Control Systems Research, Department of Engineering, University of Leicester, Leicester LE1 7RH, U.K.

[3]Department of Automatics, Technical University of Sofia, 1756 Sofia, Bulgaria

[4]University of Architecture & Civil Engineering, 1 Hr. Smirnenski Blv., 1421 Sofia, Bulgaria

Abstract

This report summarizes the progress made in the sub-task Task IV A of the *NICONET* project. Selected routines to implement $\mathcal{H}_\infty$ and $\mathcal{H}_2$ (sub) optimization syntheses are listed, which have all been standardized and included in the *SLICOT* package. The integration of those routines in MATLAB has also been completed; the *mex* files are attached in the appendices. This report discusses the selection and testing of benchmark problems with regard to the developed routines, and the comparisons made between these routines and others available in MATLAB. In particular, two industrial benchmark case studies, namely the controller design of a Bell 205 helicopter and a distillation column design, are introduced and the design results, obtained using the developed routines, are analysed

Key Words: Robust control systems design, $\mathcal{H}_\infty$ and $\mathcal{H}_2$ optimization methods, Continuous-Time and Discrete-Time systems, SLICOT

Contents

1 Introduction

Robustness is of crucial importance in control system design because real engineering systems are vulnerable to external disturbance and measurement noise and there are always differences between mathematical models used for design and the actual system. Typically a control engineer is required to design a controller which will stabilize a plant and satisfy certain performance levels in the presence of disturbance signals, noise interference, unmodelled plant dynamics and plant parameter variations.

The last two decades have witnessed a rapid development in robust control systems design, in both linear and nonlinear systems. A core achievement is the so-called $\mathcal{H}_\infty$ optimization approach and its related methods. These methods are theoretically elegant and promising in practice. They have attracted considerable interest amongst control systems researchers. Commercially available software packages which implement such methods have also emerged. Despite the convenience in using those packages such as MATLAB [1], more and more people have experienced computational difficulties in solving industrial control systems design problems with those methods. The complex mathematics of the advanced methods requires complex computations. Numerical errors become outstanding when the size of the underlying problem are large and the involved data more realistic. It is thus apparent that computationally reliable, efficient and effective software packages are still in high demand. That is the major purpose of the *Niconet* project and so is the guidelines of implementing Task IV A.

The Task IV A concentrates on the development of $\mathcal{H}_\infty$ and $\mathcal{H}_2$ parts in the package *SLICOT*, in both continuous-time case and discrete-time case. Its objectives include the development of subroutines, integration of subroutines in user-friendly environments, benchmark problems testing and performance comparison, and test applications of industrial problems, as will be addressed respectively in the following sections. There are also six reports [1]– [6] completed in Task IV A. They are available at the ftp site
ftp://wgs.esat.kuleuven.ac.be/pub/WGS/REPORTS/ and will be frequently referred in the current report.

2 Selection and collation of subroutines for standardization

The basics of $\mathcal{H}_\infty$ and $\mathcal{H}_2$ optimization approaches are introduced in the *Niconet* reports [3], [2] and [4], and will not be repeated here. Most subroutines, implementing or related to these two

[1]MATLAB is a trademark of The Mathworks, Inc

approaches, are newly programmed in the *Niconet* project and are then included in the *SLICOT* package. The algebraic Riccati equation solver routine, which uses the Schur method and was in the *SLICOT*, has also been modified with scaling and error/accuracy estimation functionalities. The coding of the subroutines employs extensively the linear algebra standard package LAPACK [7] and is implemented according to the *SLICOT* standardization criteria.

2.1 Selection of algorithms

The numerical properties of the available $\mathcal{H}_\infty/\mathcal{H}_2$ design methods are still not very well studied which requires further investigation and comparison of the various algorithms. In the case of $\mathcal{H}_\infty$ design of continuous-time systems the solution of the standard problem is usually found by the algorithm of Glover and Doyle [9]. The advantage of this algorithm is that all computations are done in the state-space which makes possible to use the high quality software for linear algebra computations available in LAPACK package [7]. The main computational task in this algorithm is the solution of two matrix algebraic Riccati equations which can be done reliably by the existing methods. A disadvantage of the algorithm of Glover and Doyle is that it requires a preliminary reduction to standard form which may require ill-conditioned transformations. In addition, one (or both) of the Riccati equations becomes more and more ill-conditioned when approaching the optimum which may deteriorate the accuracy of solution. That is why in the routines for $\mathcal{H}_\infty$ design we compute the reciprocal condition numbers of the input and output transformations and implement Riccati solver which produces condition and error estimates so that the conditioning of the corresponding Riccati equation can be monitored with the variation of the parameter γ. It is possible also to make some modifications to Glover-Doyle's algorithm, which allow to reduce the order of the matrices which are to be inverted. The modifications are fully described in [2]. Along with using LAPACK, these modifications make it possible to always obtain a more accurate solution than the solution produced by the function `hinfsyn` from μ-toolbox of MATLAB [8].

The $\mathcal{H}_2$ design of optimal controllers for continuous-time systems follows the formulas given in [12]. The corresponding algorithm also involves transformation to standard form and solution of two Riccati equations so that the $\mathcal{H}_2$ design subroutine developed produces estimates of the reciprocal condition numbers as in the case of $\mathcal{H}_\infty$ design.

In the case of $\mathcal{H}_\infty$ design of discrete-time controllers we choose the algorithm of Green and Limebeer [10]. This algorithm has the important advantage that it does not require a preliminary reduction to standard form. It involves the solution of two discrete-time Riccati equations which also may become ill-conditioned when approaching the optimum. The corresponding subroutine produces estimates of the reciprocal condition numbers of these equations along with condition estimates of the matrices which have to be inverted in the course of deriving the controller.

The $\mathcal{H}_2$ design of optimal controllers for discrete-time systems follows by the formulas given in [12] and involves transformation to standard form and solution of two discrete-time Riccati equations.

2.2 Standardization of software for $\mathcal{H}_\infty/\mathcal{H}_2$ design

The following tables list all the user-callable routines available in the *SLICOT*, concerning the $\mathcal{H}_\infty$ and $\mathcal{H}_2$ optimization approaches and related topics.

Table 1 lists the synthesis routines for $\mathcal{H}_\infty$ and $\mathcal{H}_2$ controllers in the continuous-time case.

Table 1

Name	**Function**
SB10HD	Top-level subroutine for $\mathcal{H}_2$ optimal controller design
SB10UD	$\mathcal{H}_2$ normalization transformation routine with rank condition checking
SB10VD	Computation of $\mathcal{H}_2$ optimal state feedback and output injection matrices
SB10WD	Calculation of $\mathcal{H}_2$ optimal controller state-space matrices
SB10FD	Top-level subroutine for continuous-time $\mathcal{H}_\infty$ sub-optimal controller design
SB10PD	$\mathcal{H}_\infty$ normalization transformation routine with rank condition checking
SB10QD	Computation of $\mathcal{H}_\infty$ state feedback and output injection matrices
SB10RD	Calculation of the $\mathcal{H}_\infty$ controller state-space matrices

Table 2 lists the synthesis routines for $\mathcal{H}_\infty$ and $\mathcal{H}_2$ controllers in the discrete-time case.

Table 2

Name	**Function**
SB10ED	Top-level subroutine for discrete-time $\mathcal{H}_2$ optimal controller design
SB10SD	Computation of the discrete-time $\mathcal{H}_2$ optimal controller for the standard system
SB10TD	Computation of the discrete-time $\mathcal{H}_2$ optimal controller for the general system
SB10DD	Top-level subroutine for discrete-time $\mathcal{H}_\infty$ sub-optimal controller design

Table 3 lists the related subroutines in Task IV A.

Table 3

Name	**Function**
AB13BD	Computation of the $\mathcal{H}_2$ or $\mathcal{L}_2$ norm of a continuous-time system
AB13CD	Computation of the $\mathcal{H}_\infty$ norm of a continuous-time system
SB01DD	Pole and eigenstructure assignment of a multi-input system
SB02QD	Estimation of the condition number of a continuous-time Riccati equation and estimation of the forward error
SB02RD	Solution of a continuous-time algebraic Riccati equation with condition and accuracy estimates
SB03QD	Estimation of the condition number of a continuous-time Lyapunov equation and estimation of the forward error
SB03TD	Solution of a continuous-time Lyapunov equation with condition and accuracy estimates
SB02OD	Solution of a discrete-time algebraic Riccati equation
SB02SD	Estimation of the condition number of a discrete-time Riccati equation and estimation of the forward error
SB03SD	Estimation of the condition number of a discrete-time Lyapunov equation and estimation of the forward error
SB03UD	Solution of a discrete-time Lyapunov equations with condition and accuracy estimates
SB10LD	Calculation of closed-loop system state-space matrices

3 Integration of subroutines in user-friendly environments

3.1 Integration in MATLAB

The SLICOT routines described in the previous section can be called from MATLAB. The interface is conducted through three *mex*-functions:

1. `conhin` - Using this function it is possible to design $\mathcal{H}_\infty$ (sub)optimal or $\mathcal{H}_2$ optimal controllers for continuous-time systems. This is done by calling the subroutine SB10FD in the first case or the function SB10HD in the second one. Apart from the controller matrices it is also

possible to obtain the reciprocal condition numbers of the input and output transformation matrices and the reciprocal condition numbers of the two Riccati equations involved in the design. This is useful in monitoring the reliability of the code in deriving solutions of large scale and/or ill-conditioned problems.

The MATLAB help function for `conhin` is listed in Appendix A.

2. `dishin` - This function is the discrete-time counterpart of `conhin` and allows the user to design $\mathcal{H}_\infty$ (sub)optimal or $\mathcal{H}_2$ optimal controllers for discrete-time systems. It calls the subroutines SB10DD or SB10ED. It is also possible to obtain the reciprocal condition numbers of the matrices which have to be inverted during the design as well as the condition numbers of the two Riccati equations involved in the design.

   The MATLAB help function for `dishin` is listed in Appendix B.

3. `hinorm` - This function computes the H_∞ norm of a continuous-time stable system subject to a given tolerance *tol*. It calls the double precision function AB13CD.

   The MATLAB help function for `hinorm` is given in Appendix C.

4 Selection of benchmark problems, testing and performance comparisons

The most important computational problem arising in the $\mathcal{H}_2$ and $\mathcal{H}_\infty$ design of linear control systems is the solution of the corresponding matrix algebraic Riccati equations. That is why in Subsection 4.1 we present results from the comparison of five solvers all coded for numerical solutions of the continuous-time matrix algebraic Riccati equations. The solvers include four MATLAB functions from different toolboxes and the Fortran 77 solver provided in SLICOT. A benchmark problem is employed in the comparison which comprises 1600 6-th order Riccati equations with known solutions. For each solver and each equation we compute the relative forward error and, for the SLICOT solver, we investigate the accuracy of condition and error estimates.

In the next subsections we describe the benchmark problems used in the testing of the routines for $\mathcal{H}_\infty/\mathcal{H}_2$ design and the routine for computation of the $\mathcal{H}_\infty$ norm of a continuous-time systems.

4.1 Accuracy of Riccati equation solutions

In this subsection we present the results of several numerical experiments which show the behaviour of some of the available solvers in computing the solutions of well- or ill-conditioned Riccati equations. Specifically, we compare the accuracy of the following Riccati equation solvers:

1. The function `are` from Control Systems Toolbox of MATLAB.
2. The function `aresolv` from Robust Control Toolbox of MATLAB.
3. The function `care` from Control Systems Toolbox of MATLAB.
4. The function `ricschr` from mu-toolbox of MATLAB.
5. An implementation of the Schur method with a block–scaling increasing the numerical reliability, included in SLICOT.

For this purpose we use a benchmark collection problem comprasing of 1600 6-th order Riccati equations with different conditioning. All experiments are done on PC with Pentium II processor on 333 MHz using MATLAB v. 5.3.0.10183 (R11) with relative precision $\varepsilon = 2.22 \times 10^{-16}$.

We consider the continuous–time matrix algebraic Riccati equation

$$A^T X + XA + C - XDX = 0 \tag{4.1}$$

where $A \in \mathcal{R}^{n\times n}$ and the matrices $C,\ D,\ X \in \mathcal{R}^{n\times n}$ are symmetric. We assume that the matrices of the equation are such that there exists a non–negative definite solution X which stabilises $A - DX$.

In order to have a closed form solution, the test matrices in the Riccati equation are chosen as

$$\begin{aligned} A &= TA_0T^{-1}, \\ C &= T^{-T}C_0T^{-1}, \\ D &= TD_0T^T, \end{aligned}$$

where $A_0,\ C_0,\ D_0$ are diagonal matrices and T is a nonsingular transformation matrix. The solution of (4.1) is then given by

$$X = T^{-T}X_0T^{-1}$$

where X_0 is a diagonal matrix whose elements are determined simply from the elements of $A_0,\ C_0,\ D_0$. To avoid large rounding errors in constructing and inverting T this matrix is chosen as

$$T = H_2SH_1$$

where H_1 and H_2 are elementary reflectors and S is a diagonal matrix,

$$\begin{aligned} H_1 &= I_n - 2ee^T/n, \quad e = [1, 1, ..., 1]^T \\ H_2 &= I_n - 2ff^T/n, \quad f = [1, -1, 1, ..., (-1)^{n-1}]^T, \\ S &= \mathrm{diag}(1, s, s^2, ..., s^{n-1}), \quad s > 1. \end{aligned}$$

Using different values of the scalar s, it is possible to change the condition number of the matrix T with respect to inversion,

$$\mathrm{cond}_2(T) = s^{n-1}.$$

Taking into account the form of T we obtain that

$$\begin{aligned} A &= H_2 S H_1 A_0 H_1 S^{-1} H_2, \\ C &= H_2 S^{-1} H_1 C_0 H_1 S^{-1} H_2, \\ D &= H_2 S H_1 D_0 H_1 S H_2. \end{aligned}$$

These matrices are computed easily with relative precision of order ε. Apart from the simplicity of these Riccati equations, their numerical solution presents a difficult task for the available methods for solving the Riccati equation since the diagonal structure of the equations is not recognized by these methods. On the other hand, the use of such equations in testing the corresponding numerical methods allows to check easily the solution accuracy.

In the given case we take

$$\begin{aligned} A_0 &= \mathrm{diag}(A_1, A_1), \\ C_0 &= \mathrm{diag}(C_1, C_1), \\ D_0 &= \mathrm{diag}(D_1, D_1), \end{aligned}$$

where

$$\begin{aligned} A_1 &= \mathrm{diag}(1 \times 10^k, 2 \times 10^k, 3 \times 10^k), \\ C_1 &= \mathrm{diag}(1 \times 10^{-k}, 1, 1 \times 10^k), \\ D_1 &= \mathrm{diag}(10^{-k}, 10^{-k}, 10^{-k}). \end{aligned}$$

The solution X_0 consists of two copies of diagonal blocks given by

$$\begin{aligned} X_1 &= \mathrm{diag}(x_1, x_2, x_3), \\ x_i &= (a_{ii} + \sqrt{a_{ii}^2 + c_{ii} d_{ii}})/d_{ii} \end{aligned}$$

where $a_{ii}, c_{ii}, d_{ii},\ i = 1, 2, 3$ are the corresponding diagonal elements of A_1, C_1, D_1, respectively. These equations are well conditioned (K_F is of order 1) for small s but in the unscaled version of the Schur method the difference between the norms of the blocks of Hamiltonian matrix increases quickly with k which introduces large errors in the solution.

In Fig. 1 we show the conditioning of the corresponding Riccati equations for the different values of k and s. The condition number is computed by the function `cndricc` given in Appendix

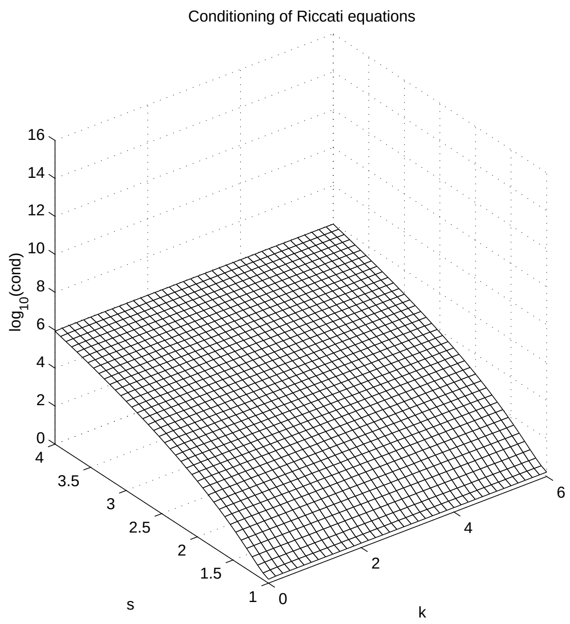

Figure 1: Conditioning of the Riccati equations

D. Practically, the conditioning does not depend on k and the maximum value of K_F is of order 10^6.

In Fig. 2 we show the forward errors for the function `are` from MATLAB. The forward errors increase with s which is explained with the increasing of the condition number of the Riccati equation. However, the forward error increases also with the increase of k which is a signal of potential numerical instability. In fact, for $k = 6$ and $s = 4$ the forward error is of order 10^{-3}, but from the sensitivity analysis we can expect errors of order only $K_F\varepsilon \approx 10^{-10}$. This is a result from the usage of the unscaled Schur method which is potentially numerically unstable.

The forward error for the function `aresolv`, obtained by using of the Schur method option, is given in Fig. 3. This error is slightly smaller than in the case of the function `are` but this function is obviously also numerically unstable.

The numerical behaviour of the function `care`, as illustrated in Fig. 4, is much better. The forward errors are slightly greater than expected from the conditioning, but the corresponding implementation could be accepted as conditionally stable. This property of the `care` are due to the block-scaling of the Hamiltonian matrix which allows to avoid the instability of the Schur method.

In Fig. 5 we present the results of using the function `ricschr` from mu-toolbox of MATLAB. They are very close to the results obtained by `aresolv` and demonstrate again the numerical

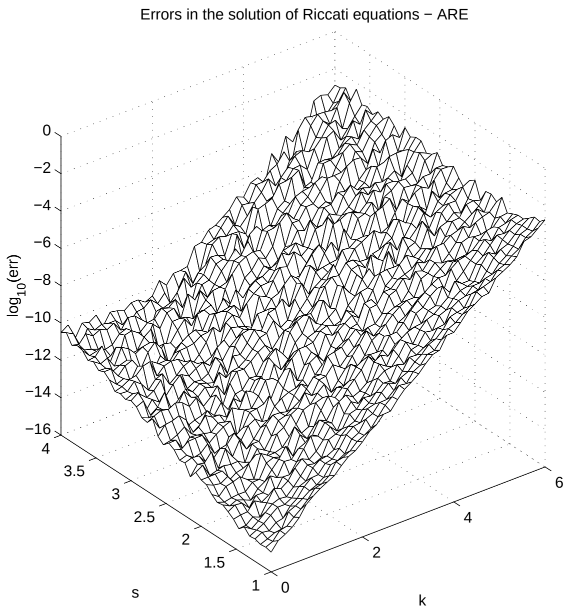

Figure 2: Forward errors for `are`

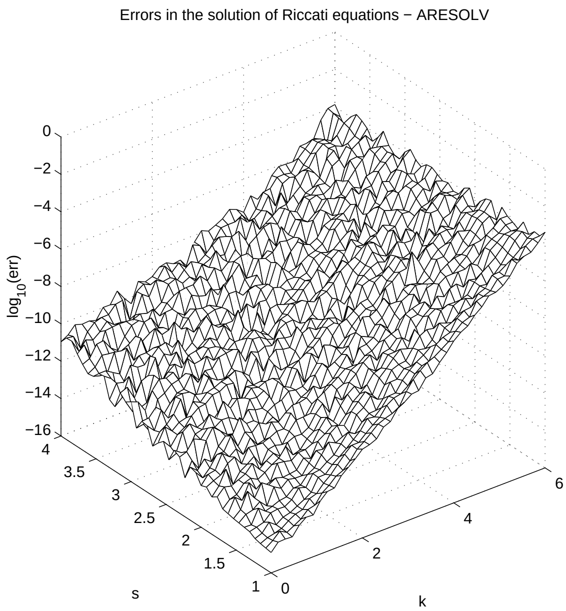

Figure 3: Forward errors for `aresolv`

instability of the unscaled Schur method.

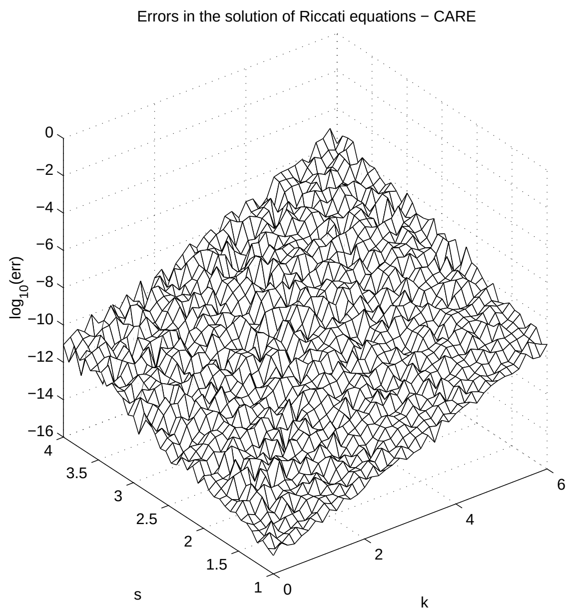

Figure 4: Forward errors for `care`

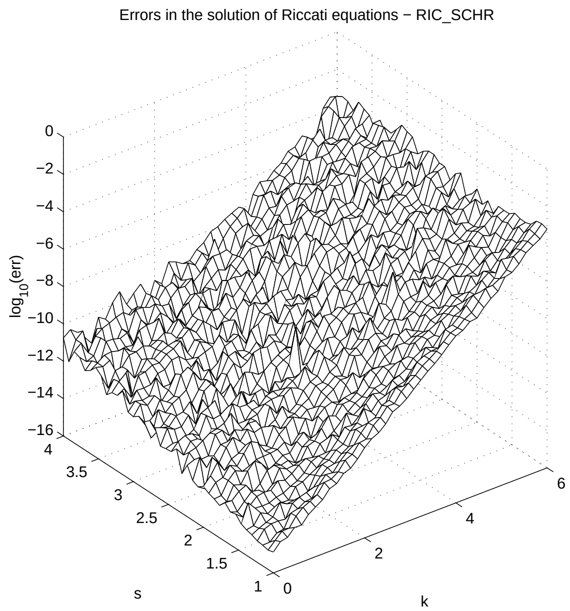

Figure 5: Forward errors for `ricschr`

The numerical behaviour of the solver `ricc` implementing the Schur method with block scaling is illustrated in Fig. 6. As in the case of the function `care` the forward error is close to the bound, predicted by the conditioning, so that the corresponding implementation may be considered as conditionally stable. This solver, however, produces also estimates of the condition number and bound on the forward error which increases significantly the reliability of the implementation.

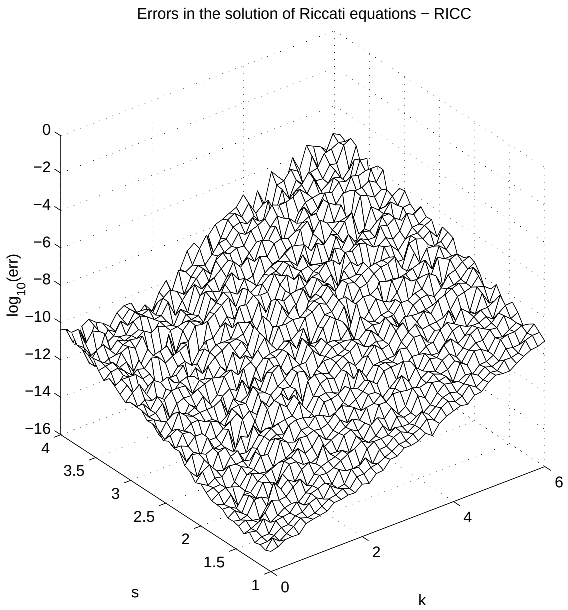

Figure 6: Forward errors for `ricc`

In Fig. 7 we show for the same example the ratio of the condition number estimate and the true condition number. It is seen that the difference between these quantities is less than half a decimal digit for all values of k and s. Experiments with other examples confirm the high reliability of the condition number estimator which makes use of the LAPACK estimator `DLACON`. The behaviour of the forward error estimator implemented in `ricc` is illustrated in Fig. 8 where we show the ratio of the true forward error and the forward error bound as a function of k and s. It is seen from the figure that the bound is always greater than the magnitude of the actual error (as one should expect) and that the bound becomes more pessimistic with the increasing of s. For this example the pessimism of the estimate does not exceed 3 decimal digits for any k and s which is fully acceptable.

A similar evaluation of the solvers of the discrete-time Riccati equation will be done after the inclusion in SLICOT of a stabilized solver for this equation.

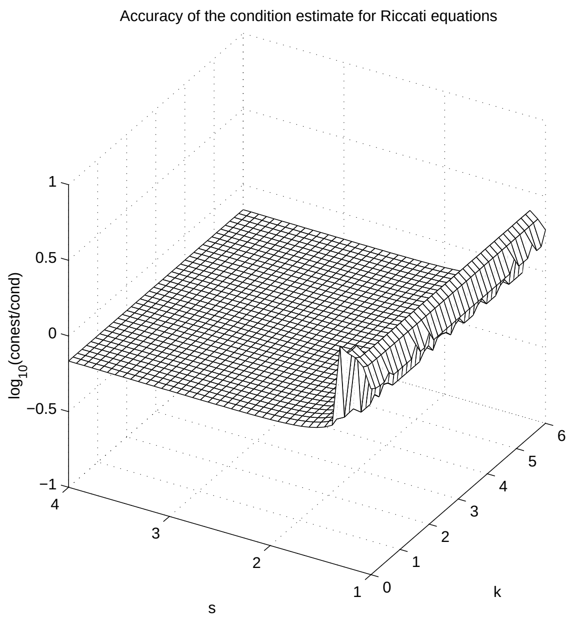

Figure 7: Accuracy of the condition estimate for `ricc`

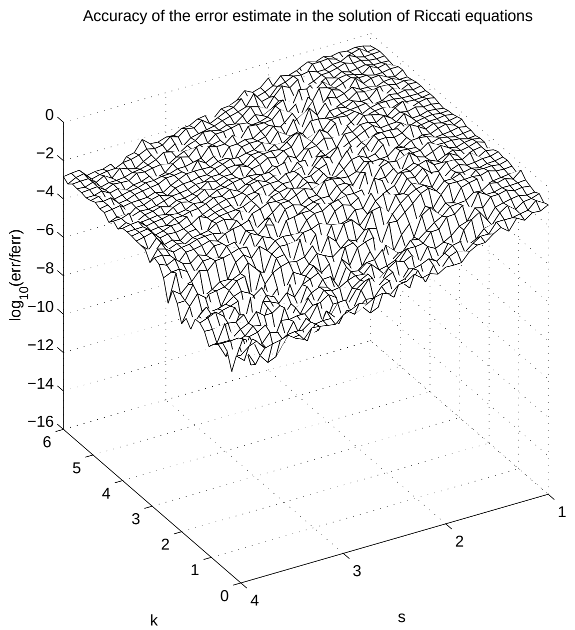

Figure 8: Accuracy of the forward error estimate for `ricc`

4.2 Design of $\mathcal{H}_\infty$ suboptimal controllers for continuous-time systems

As a benchmark problem for $\mathcal{H}_\infty$ design we consider the computation of suboptimal controllers for a sixth order system with $m = 5, p = 5$ and $m_2 = 2,\ p_2 = 2$ and matrices

$$A = \begin{bmatrix} -1 & 0 & 4 & 5 & -3 & -2 \\ -2 & 4 & -7 & -2 & 0 & 3 \\ -6 & 9 & -5 & 0 & 2 & -1 \\ -8 & 4 & 7 & -1 & -3 & 0 \\ 2 & 5 & 8 & -9 & 1 & -4 \\ 3 & -5 & 8 & 0 & 2 & -6 \end{bmatrix}, B = \begin{bmatrix} -3 & -4 & -2 & 1 & 0 \\ 2 & 0 & 1 & -5 & 2 \\ -5 & -7 & 0 & 7 & -2 \\ 4 & -6 & 1 & 1 & -2 \\ -3 & 9 & -8 & 0 & 5 \\ 1 & -2 & 3 & -6 & -2 \end{bmatrix},$$

$$C = \begin{bmatrix} 1 & -1 & 2 & -4 & 0 & -3 \\ -3 & 0 & 5 & -1 & 1 & 1 \\ -7 & 5 & 0 & -8 & 2 & -2 \\ 9 & -3 & 4 & 0 & 3 & 7 \\ 0 & 1 & -2 & 1 & -6 & -2 \end{bmatrix}, D = \begin{bmatrix} 1 & -2 & -3 & 0 & 0 \\ 0 & 4 & 0 & 1 & 0 \\ 5 & -3 & -4 & 0 & 1 \\ 0 & 1 & 0 & 1 & -3 \\ 0 & 0 & 1 & 7 & 1 \end{bmatrix}.$$

Using the mex-function `conhin` which makes use of the SLICOT subroutine SB10FD it was found by some trial and error that the optimal value of γ for this system is $\gamma_{opt} = 10.18425636157899$. The controller matrices in this case are (up to four digits)

$$A_k = 10^9 \begin{bmatrix} -0.6113 & 3.3477 & 2.7572 & 1.3328 & 1.6416 & 1.4269 \\ -0.2162 & 1.1839 & 0.9751 & 0.4713 & 0.5805 & 0.5046 \\ -0.6637 & 3.6348 & 2.9936 & 1.4471 & 1.7823 & 1.5493 \\ -0.2620 & 1.4346 & 1.1815 & 0.5711 & 0.7034 & 0.6115 \\ -0.9298 & 5.0922 & 4.1939 & 2.0273 & 2.4970 & 2.1705 \\ 0.2313 & -1.2669 & -1.0434 & -0.5044 & -0.6212 & -0.5400 \end{bmatrix},$$

$$B_k = \begin{bmatrix} -0.2225 & -0.1085 \\ -0.8518 & -0.6521 \\ 0.8173 & 0.5794 \\ 0.0843 & 0.0100 \\ -0.5632 & -0.2479 \\ 0.0068 & -0.7619 \end{bmatrix},$$

$$C_k = 10^8 \begin{bmatrix} -0.0170 & 0.0928 & 0.0765 & 0.0370 & 0.0455 & 0.0396 \\ -0.3556 & 1.9476 & 1.6041 & 0.7754 & 0.9550 & 0.8302 \end{bmatrix}, D_k = \begin{bmatrix} 0.0552 & 0.1334 \\ -0.3195 & 0.0333 \end{bmatrix}.$$

This controller is characterised by very high gains.

The estimates of the condition numbers of the X- and Y-Riccati equations in this case are $\mathrm{cond}_X = 40.7$ and $\mathrm{cond}_Y = 7625.7$, respectively, which shows that the Y-Riccati equation is relatively ill-conditioned.

In order to obtain controller with lower gains, we repeat the design using $\gamma = 15$. In this case one obtains

$$A_k = \begin{bmatrix} -2.8043 & 14.7367 & 4.6658 & 8.1596 & 0.0848 & 2.5290 \\ 4.6609 & 3.2756 & -3.5754 & -2.8941 & 0.2393 & 8.2920 \\ -15.3127 & 23.5592 & -7.1229 & 2.7599 & 5.9775 & -2.0285 \\ -22.0691 & 16.4758 & 12.5523 & -16.3602 & 4.4300 & -3.3168 \\ 30.6789 & -3.9026 & -1.3868 & 26.2357 & -8.8267 & 10.4860 \\ -5.7429 & 0.0577 & 10.8216 & -11.2275 & 1.5074 & -10.7244 \end{bmatrix},$$

$$B_k = \begin{bmatrix} -0.1581 & -0.0793 \\ -0.9237 & -0.5718 \\ 0.7984 & 0.6627 \\ 0.1145 & 0.1496 \\ -0.6743 & -0.2376 \\ 0.0196 & -0.7598 \end{bmatrix},$$

$$C_k = \begin{bmatrix} -0.2480 & -0.1713 & -0.0880 & 0.1534 & 0.5016 & -0.0730 \\ 2.8810 & -0.3658 & 1.3007 & 0.3945 & 1.2244 & 2.5690 \end{bmatrix}, D_k = \begin{bmatrix} 0.0554 & 0.1334 \\ -0.3195 & 0.0333 \end{bmatrix}.$$

This suboptimal controller is relatively easy to implement due to the low gains. In the given case the estimate of the condition numbers of the Riccati equations are $\text{cond}_X = 101.8$ and $\text{cond}_Y = 1424.6$, so that the conditioning of these equations is improved.

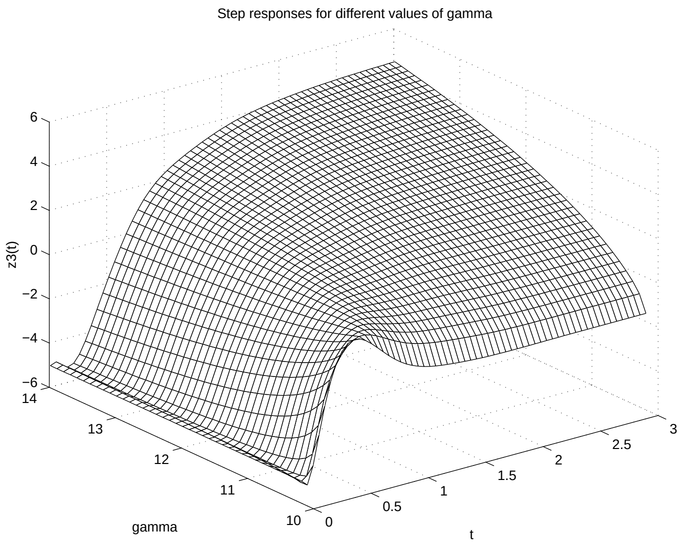

Figure 9: Step responses for different values of γ

In Figure 9 we show a family of closed-loop step responses of the component $z_3(t)$ for (sub)optimal controllers computed for several values of γ from $\gamma = \gamma_{opt}$ to $\gamma = 14.0$ and for $w_1(t) = -1(t), w_2(t) = 0$. It is seen that with the decreasing of γ the step response becomes faster and the steady-state value of the response decreases.

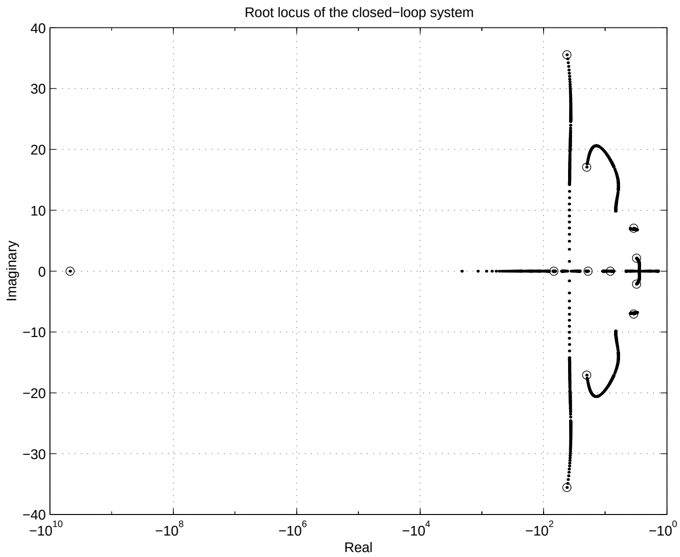

Figure 10: Root loci of the closed-loop system for different values of γ

In Figure 10 we show the closed-loop system root loci for the same family of (sub)optimal controllers. The circles denote the location of the closed-loop poles for the optimal controller.

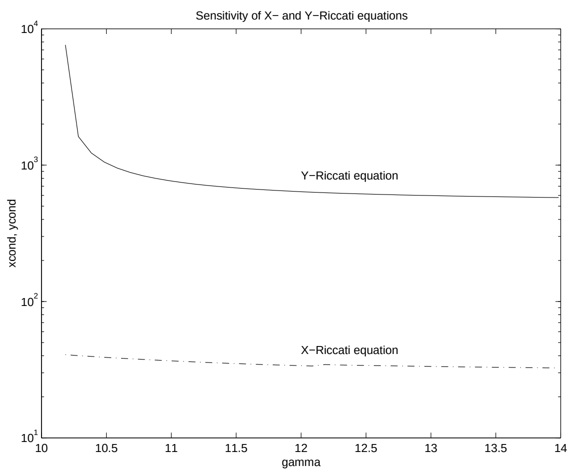

Figure 11: Sensitivity of X- and Y-Riccati equations for different values of γ

The dependence of the condition numbers of X- and Y-Riccati equations on γ is shown in Figure 11. It is seen, that with the decrease of γ the conditioning of both equations deteriorates.

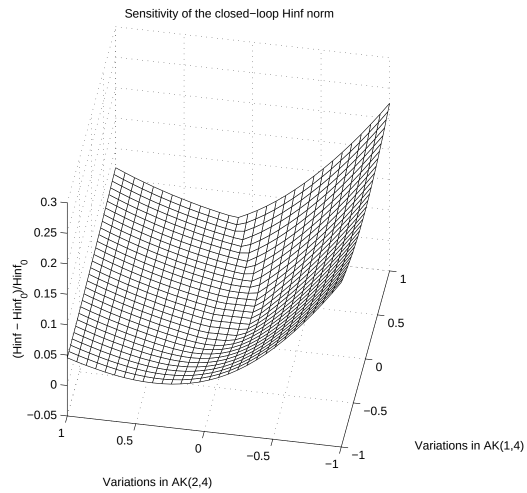

Figure 12: Sensitivity of the closed-loop $\mathcal{H}_\infty$ norm for $\gamma = 10.5$

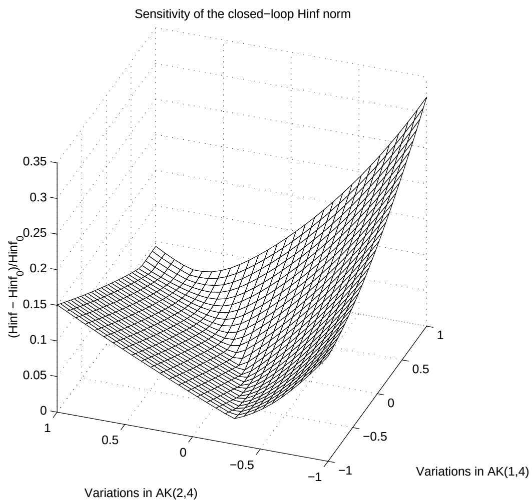

Figure 13: Sensitivity of the closed-loop $\mathcal{H}_\infty$ norm for $\gamma = 10.18426$

In Figures 12 and 13 we show the sensitivity of the $\mathcal{H}_\infty$ norm of the closed-loop system for two values of γ and relatively large variations in the elements A_{k14} and A_{k24} of the controller.

4.3 Design of $\mathcal{H}_2$ optimal controllers for continuous-time systems

As a benchmark problem for $\mathcal{H}_2$ design we consider the computation of the optimal controller for a sixth order system with $m = 5, p = 5$ and $m_2 = 2,\ p_2 = 2$ and matrices

$$A = \begin{bmatrix} -1 & 0 & 4 & 5 & -3 & -2 \\ -2 & 4 & -7 & -2 & 0 & 3 \\ -6 & 9 & -5 & 0 & 2 & -1 \\ -8 & 4 & 7 & -1 & -3 & 0 \\ 2 & 5 & 8 & -9 & 1 & -4 \\ 3 & -5 & 8 & 0 & 2 & -6 \end{bmatrix}, B = \begin{bmatrix} -3 & -4 & -2 & 1 & 0 \\ 2 & 0 & 1 & -5 & 2 \\ -5 & -7 & 0 & 7 & -2 \\ 4 & -6 & 1 & 1 & -2 \\ -3 & 9 & -8 & 0 & 5 \\ 1 & -2 & 3 & -6 & -2 \end{bmatrix},$$

$$C = \begin{bmatrix} 1 & -1 & 2 & -4 & 0 & -3 \\ -3 & 0 & 5 & -1 & 1 & 1 \\ -7 & 5 & 0 & -8 & 2 & -2 \\ 9 & -3 & 4 & 0 & 3 & 7 \\ 0 & 1 & -2 & 1 & -6 & -2 \end{bmatrix}, D = \begin{bmatrix} 0 & 0 & 0 & -4 & -1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & -0 & 0 & 1 \\ 3 & 1 & 0 & 1 & -3 \\ -2 & 0 & 1 & 7 & 1 \end{bmatrix}.$$

(Note that the blok D_{11} is zero.)

Using the mex-function `conhin` we obtain the following controller

$$A_k = \begin{bmatrix} 88.0015 & -145.7298 & -46.2424 & 82.2168 & -45.2996 & -31.1407 \\ 25.7489 & -31.4642 & -12.4198 & 9.4625 & -3.5182 & 2.7056 \\ 54.3008 & -102.4013 & -41.4968 & 50.8412 & -20.1286 & -26.7191 \\ 108.1006 & -198.0785 & -45.4333 & 70.3962 & -25.8591 & -37.2741 \\ -115.8900 & 226.1843 & 47.2549 & -47.8435 & -12.5004 & 34.7474 \\ 59.0362 & -101.8471 & -20.1052 & 36.7834 & -16.1063 & -26.4309 \end{bmatrix},$$

$$B_k = \begin{bmatrix} 3.7345 & 3.4758 \\ -0.3020 & 0.6530 \\ 3.4735 & 4.0499 \\ 4.3198 & 7.2755 \\ -3.9424 & -10.5942 \\ 2.1784 & 2.5048 \end{bmatrix},$$

$$C_k = \begin{bmatrix} -2.3346 & 3.2556 & 0.7150 & -0.9724 & 0.6962 & 0.4074 \\ 7.6899 & -8.4558 & -2.9642 & 7.0365 & -4.2844 & 0.1390 \end{bmatrix}$$

(In this case the controller is strictly proper, i.e. $D_k = 0$).

In Figure 14 we show the sensitivity of the $\mathcal{H}_2$ norm of the closed loop system for the same variations in the elements A_{k14} and A_{k24}.

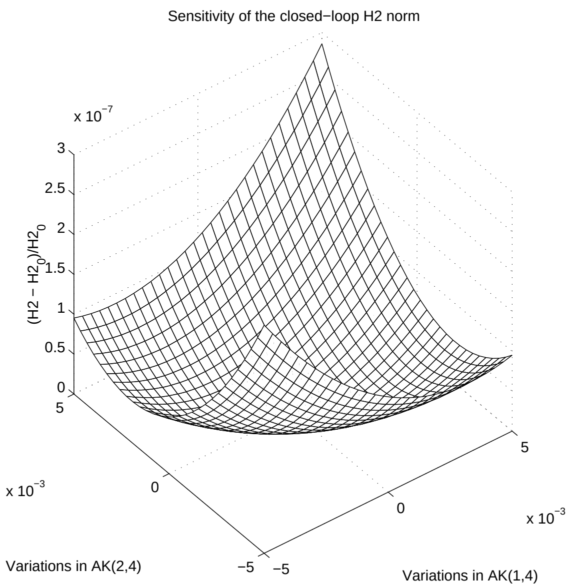

Figure 14: Sensitivity of the closed-loop $\mathcal{H}_2$ norm

4.4 Design of $\mathcal{H}_\infty$ suboptimal controllers for discrete-time systems

As a benchmark problem for discrete-time $\mathcal{H}_\infty$ design we consider the computation of optimal controller for a sixth order system with $m = 5,\ p = 5$ and $m_2 = 2,\ p_2 = 2$ and matrices

$$A = \begin{bmatrix} -0.7 & 0 & 0.3 & 0 & -0.5 & -0.1 \\ -0.6 & 0.2 & -0.4 & -0.3 & 0 & 0 \\ -0.5 & 0.7 & -0.1 & 0 & 0 & -0.8 \\ -0.7 & 0 & 0 & -0.5 & -1.0 & 0 \\ 0 & 0.3 & 0.6 & -0.9 & 0.1 & -0.4 \\ 0.5 & -0.8 & 0 & 0 & 0.2 & -0.9 \end{bmatrix}, B = \begin{bmatrix} -1 & -2 & -2 & 1 & 0 \\ 1 & 0 & 1 & -2 & 1 \\ -3 & -4 & 0 & 2 & -2 \\ 1 & -2 & 1 & 0 & -1 \\ 0 & 1 & -2 & 0 & 3 \\ 1 & 0 & 3 & -1 & -2 \end{bmatrix},$$

$$C = \begin{bmatrix} 1 & -1 & 2 & -2 & 0 & -3 \\ -3 & 0 & 1 & -1 & 1 & 0 \\ 0 & 2 & 0 & -4 & 0 & -2 \\ 1 & -3 & 0 & 0 & 3 & 1 \\ 0 & 1 & -2 & 1 & 0 & -2 \end{bmatrix}, D = \begin{bmatrix} 1 & -1 & -2 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 2 & -1 & -3 & 0 & 1 \\ 0 & 1 & 0 & 1 & -1 \\ 0 & 0 & 1 & 2 & 1 \end{bmatrix}.$$

Using the mex-function **dishin** which calls the subroutine SB10DD it was found by some trial and error that the optimal value of γ for this system is $\gamma_{opt} = 111.2931936924534$. The controller matrices in this case are (up to four digits)

$$A_k = \begin{bmatrix} -17.9945 & 52.0119 & 26.0705 & -0.4270 & -40.8826 & 18.0767 \\ 18.8109 & -57.5960 & -29.0798 & 0.5870 & 45.3091 & -19.8544 \\ -26.5868 & 77.9314 & 39.0182 & -1.4019 & -60.0839 & 26.6776 \\ -21.4062 & 62.1415 & 30.7357 & -0.9200 & -48.5988 & 21.8244 \\ -0.8905 & 4.2767 & 2.3276 & -0.2424 & -3.0361 & 1.2162 \\ -5.3260 & 16.1877 & 8.4786 & -0.2489 & -12.2288 & 5.1562 \end{bmatrix},$$

$$B_k = \begin{bmatrix} 16.9705 & 14.1579 \\ -18.9123 & -15.6649 \\ 25.1924 & 21.2745 \\ 20.1024 & 16.8240 \\ 1.4098 & 1.2034 \\ 5.3156 & 4.5127 \end{bmatrix},$$

$$C_k = \begin{bmatrix} -9.1896 & 27.5030 & 13.7297 & -0.3638 & -21.5879 & 9.5978 \\ 3.6473 & -10.6143 & -5.2747 & 0.2431 & 8.1069 & -3.6275 \end{bmatrix},$$

$$D_k = \begin{bmatrix} 9.0273 & 7.5311 \\ -3.3990 & -2.8205 \end{bmatrix}.$$

The estimates of the condition numbers of the X- and Z-Riccati equations in this case are $\text{cond}_X = 3.1763 \times 10^3$ and $\text{cond}_Z = 2.6594 \times 10^{13}$, respectively, which shows that the Z-Riccati equation is extremally ill-conditioned.

In Figure 15 we show the closed-loop system response to a unit step disturbance w_1 obtained for the optimal controller.

In Figure 16 we show the closed-loop system root loci for a family of (sub)optimal controllers obtained for different values of $\gamma \in [\gamma_{opt}, 1000]$. The circles denote the location of the closed-loop poles for the optimal controller.

The dependence of the condition numbers of X- and Z-Riccati equations on γ is shown in Figure 17. It is seen, that with the decrease of γ the conditioning of the Z-Riccati equation deteriorates very much.

In Figure 18 we show the sensitivity of the $\mathcal{H}_\infty$ norm of the closed-loop system for $\gamma = \gamma_{opt}$ with respect to variations in the elements A_{k25}, A_{k41} of the controller.

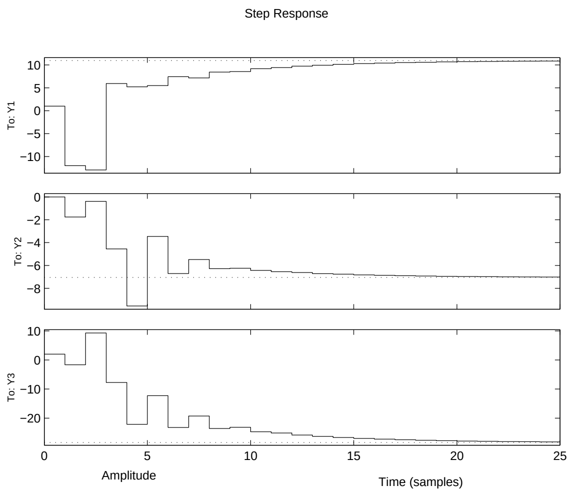

Figure 15: Step response of the H_∞-optimal closed-loop system

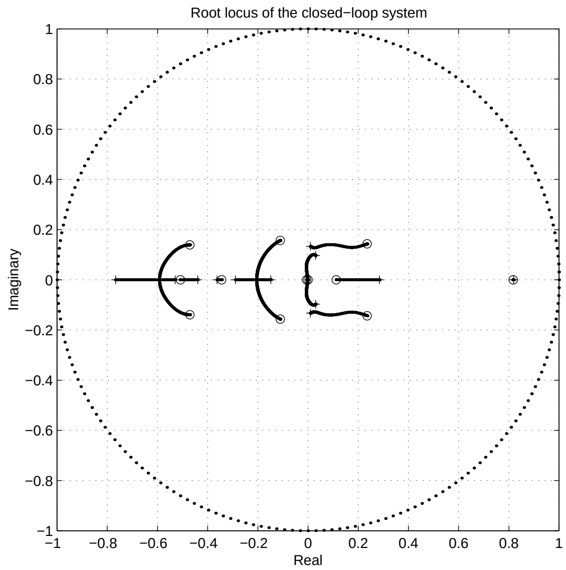

Figure 16: Root loci of the closed-loop system for different values of γ

4.5 Design of $\mathcal{H}_2$ optimal controllers for discrete-time systems

For the same system as in the previous subsection the mex-file `dishin`, which for task $= 2$ calls the subroutine SB10ED, produces the following optimal $\mathcal{H}_2$ controller,

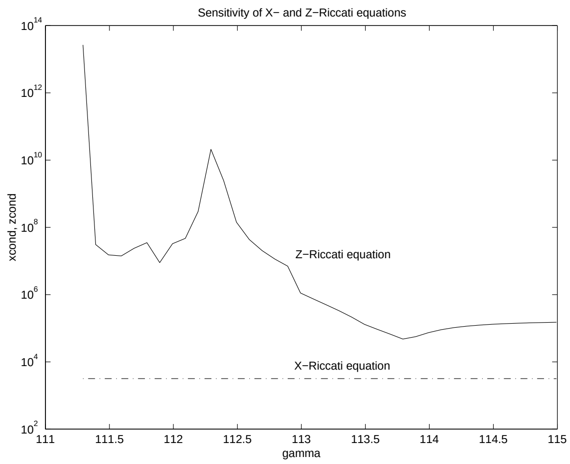

Figure 17: Sensitivity of X- and Z-Riccati equations for different values of γ

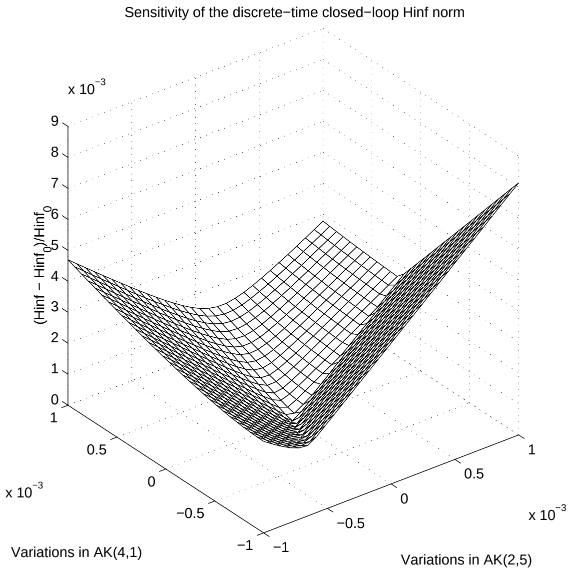

Figure 18: Sensitivity of the closed-loop $\mathcal{H}_\infty$ norm for $\gamma = \gamma_{opt}$

$$A_k = \begin{bmatrix} -0.0551 & -2.1891 & -0.6607 & -0.2532 & 0.6674 & -1.0044 \\ -1.0379 & 2.3804 & 0.5031 & 0.3960 & -0.6605 & 1.2673 \\ -0.0876 & -2.1320 & -0.4701 & -1.1461 & 1.2927 & -1.5116 \\ -0.1358 & -2.1237 & -0.9560 & -0.7144 & 0.6673 & -0.7957 \\ 0.4900 & 0.0895 & 0.2634 & -0.2354 & 0.1623 & -0.2663 \\ 0.1672 & -0.4163 & 0.2871 & -0.1983 & 0.4944 & -0.6967 \end{bmatrix},$$

$$B_k = \begin{bmatrix} -0.5985 & -0.5464 \\ 0.5285 & 0.6087 \\ -0.7600 & -0.4472 \\ -0.7288 & -0.6090 \\ 0.0532 & 0.0658 \\ -0.0663 & 0.0059 \end{bmatrix},$$

$$C_k = \begin{bmatrix} 0.2500 & -1.0200 & -0.3371 & -0.2733 & 0.2747 & -0.4444 \\ 0.0654 & 0.2095 & 0.0632 & 0.2089 & -0.1895 & 0.1834 \end{bmatrix},$$

$$D_k = \begin{bmatrix} -0.2181 & -0.2070 \\ 0.1094 & 0.1159 \end{bmatrix}.$$

(Note that the controller is not strictly proper, i.e. $D_k \neq 0$).

In Figure 19 we show the unit step response of the $\mathcal{H}_2$ optimal closed-loop system. In Figure 20 we show the sensitivity of the $\mathcal{H}_2$ norm of the closed loop system for the same variations in the elements A_{k25} and A_{k41} as in the case of $\mathcal{H}_\infty$ optimal design.

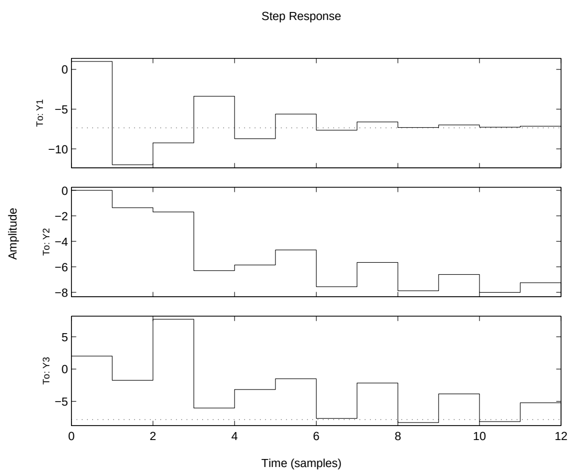

Figure 19: Step response of the H_2-optimal closed-loop system

4.6 Computation of the $\mathcal{H}_\infty$ norm of a continuous-time system

The benchmark problem for this task is a sixth-order single-input single-output system which is characterized by a very narrow and large peak of the singular value of the transfer function matrix

$$T(s) = C(sI - A)^{-1}B.$$

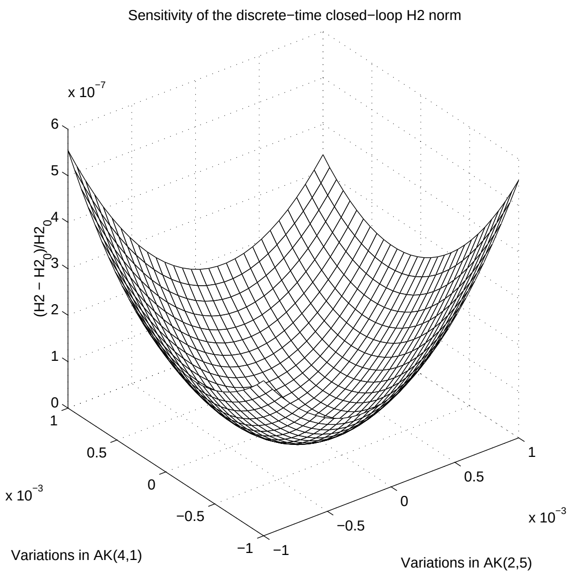

Figure 20: Sensitivity of the closed-loop $\mathcal{H}_2$ norm

(This makes the numerical computation of the $\mathcal{H}_\infty$ norm very difficult.) The matrices of the system are

$$A = \begin{bmatrix} 0 & 1.0 & 0 & 0 & 0 & 0 \\ -0.5 & -0.0002 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1.0 & 0 & 0 \\ 0 & 0 & -1.0 & -0.00002 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1.0 \\ 0 & 0 & 0 & 0 & -2.0 & -0.000002 \end{bmatrix}, B = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \end{bmatrix},$$

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

The singular value of the transfer function as a function of the frequency is shown in Fig. 21.

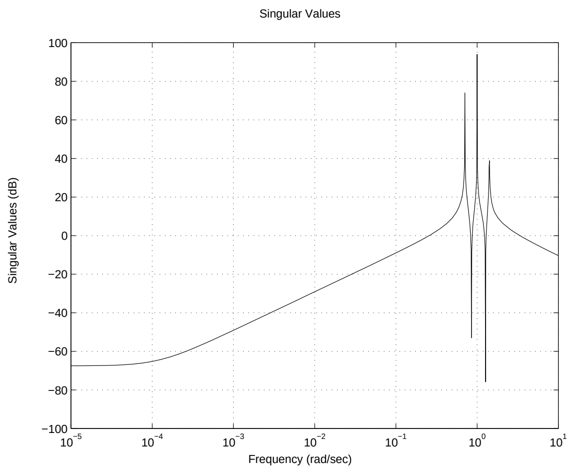

Figure 21: Singular value of the transfer function

For this system the computed value of the $\mathcal{H}_\infty$ norm depends very much on the threshold used. For $tol = 10^{-9}$ we obtain that $\|T(s)\|_\infty = 0.5000000006D + 06$.

5 Testing on industrial benchmark problems and comparisons

5.1 Design of $\mathcal{H}_\infty$ suboptimal controller for Bell 205A-1 helicopter

In this example we consider the design of a continuous-time $\mathcal{H}_\infty$ suboptimal controller for the Bell 201A-1 helicopter using the SLICOT subroutines SB10FD, SB10PD, SB10QD and SB10RD. The linearized helicopter dynamics is described by a 8-th order, 4 input, 6 output state space model

$$\begin{aligned} \dot{x} &= Ax + Bu, \\ y &= Cx + Du \end{aligned}$$

where the state vector

$$x = [u_H \; w \; q \; v \; p \; r \; \theta \; \phi \,]^T$$

consists of the variables

u_H	forward velocity
w	vertical velocity
q	pitch rate
v	lateral velocity
p	roll rate
r	yaw rate
θ	pitch angle
ϕ	roll angle

the input vector

$$u = [\delta_{coll} \; \delta_{long.cyclic} \; \delta_{lat.cyclic} \; \delta_{t.r.coll}]^T$$

consists of the variables

δ_{coll}	main rotor collective
$\delta_{long.cyclic}$	longitudinal cyclic
$\delta_{lat.cyclic}$	lateral cyclic
$\delta_{t.r.coll}$	tail rotor collective

and the output vector

$$y = [w \; \theta \; \phi \; r \; q \; p \,]^T$$

consists of the corresponding six state variables.

The model matrices $A,\ B,\ C,\ D$ are obtained as (velocity 60 knots at sea level)

$$A = \begin{bmatrix} -0.0046 & 0.0380 & 0.3259 & -0.0045 & -0.4020 & -0.0730 & -9.8100 & 0 \\ -0.1978 & -0.5667 & 0.3570 & -0.0378 & -0.2149 & 0.5683 & 0 & 0 \\ 0.0039 & -0.0029 & -0.2947 & 0.0070 & 0.2266 & 0.0148 & 0 & 0 \\ 0.0133 & -0.0014 & -0.4076 & -0.0654 & -0.4093 & 0.2674 & 0 & 9.81 \\ 0.0127 & -0.0100 & -0.8152 & -0.0397 & -0.8210 & 0.1442 & 0 & 0 \\ -0.0285 & -0.0232 & 0.1064 & 0.0709 & -0.2786 & -0.7396 & 0 & 0 \\ 0 & 0 & 1.0000 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1.0000 & 0 & 0 & 0 \end{bmatrix},$$

$$B = \begin{bmatrix} 0.0676 & 0.1221 & -0.0001 & -0.0016 \\ -1.1151 & 0.1055 & 0.0039 & 0.0035 \\ 0.0062 & -0.0682 & 0.0010 & -0.0035 \\ -0.0170 & 0.0049 & 0.1067 & 0.1692 \\ -0.0129 & 0.0106 & 0.2227 & 0.1430 \\ 0.1390 & 0.0059 & 0.0326 & -0.4070 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix},$$

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

Let

$$G(s) = C(sI - A)^{-1}B + D$$

be the transfer function of the open-loop system so that $y(s) = G(s)u(s)$.

In the given case the design aim is to find a controller with a transfer function matrix $K(s)$ which is a solution of the *mixed-sensitivity problem* (see for details the NICONET Report 1999-4 [3])

$$\left\| \begin{bmatrix} W_1 S \\ W_2 K S \end{bmatrix} \right\|_\infty < \gamma \tag{5.1}$$

for a prescribed value of $\gamma > 0$, where W_1, W_2 are suitably chosen weighting functions, which specify the requirements to the closed-loop system dynamics in the frequency domain.

The problem (5.1) can be represented in the standard form

$$\|F_\ell(P,K)\|_\infty < \gamma \tag{5.2}$$

where

$$F_\ell(P,K) = P_{11} + P_{12}K(I - P_{22}K)^{-1}P_{21}$$

and P is *the generalized plant* given by

$$P = \begin{bmatrix} W_1 & -W_1G \\ 0 & W_2 \\ I & -G \end{bmatrix}.$$

In (5.2) it should be fulfilled that

$$\gamma > \gamma_0 := \min_{K_{stabilizing}} \|F_\ell(P,K)\|_\infty$$

which means that we solve a $\mathcal{H}_\infty$ *sub-optimal problem.*

For the problem under consideration W_1 and W_2 are chosen as diagonal matrices, the frequency response characterictics of whose elements are shown in Figure 22 and Figure 23, respectively.

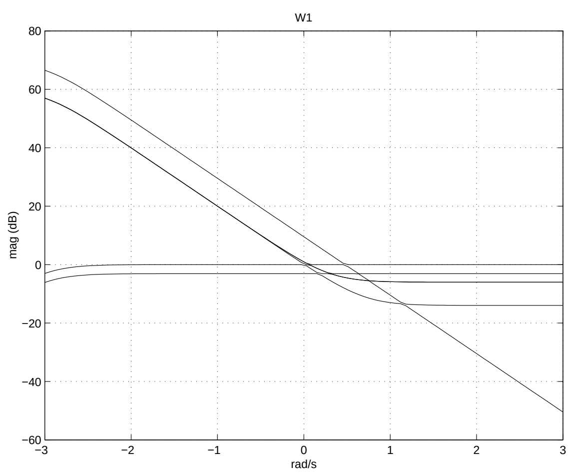

Figure 22: Singular values of W_1

The generalized plant is of 18-th order which leads to a controller of the same size. The standard problem (5.2) is solved by using the subroutines SB10FD, SB10PD, SB10QD and SB10RD using the algorithm described in NICONET Report 1998-8 [2]. This algorithm involves the solution of two matrix Riccati equations which in the given case are also of 18-th order. It should be stressed

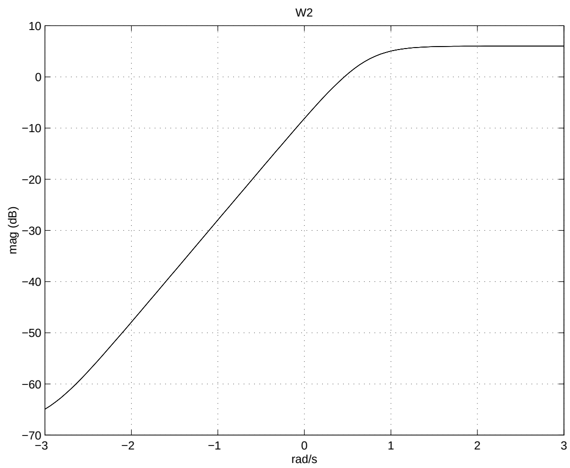

Figure 23: Singular values of W_2

that the subroutine SB10PD involves the input and output transformations T_u and T_y, respectively, to normalized form which are potentially unstable from a numerical point of view. The subroutine SB10PD produces the condition numbers of these transformations which is useful in monitoring the numerical behaviour of the algorithm. In the given case they are

$$\mathrm{cond}(T_u) = 1, \quad \mathrm{cond}(T_y) = 10$$

which shows that the transformations are very well conditioned. The standard problem (5.2) is solved for $\gamma = 2.875$ which was chosen to ensure a good compromise between the requirements to the system dynamics and the necessity to reduce the gains as much as possible. This value is close to the minimum possible one which is indicated by the condition numbers of the X- and Y-Riccati equations, produced by the subroutine SB10QD:

$$\mathrm{cond}_X = 2.3564e+06, \quad \mathrm{cond}_Y = 80.1560.$$

The condition number of the X-Riccati equation shows that approximately 6 digits are lost in the solution of this equation. Note that condition numbers larger than the reciprocal of the square root of the machine precision (about 10^8 for the double precision of most machines) may lead to serious numerical difficulties.

In Figure 24 we present the step-responses of the closed-loop system for unit steps on the four reference inputs. It is seen that all step response are satisfactory and the four channels are almost decoupled. It should be pointed out that this result is very diffucult to achieve by using the traditional design methods.

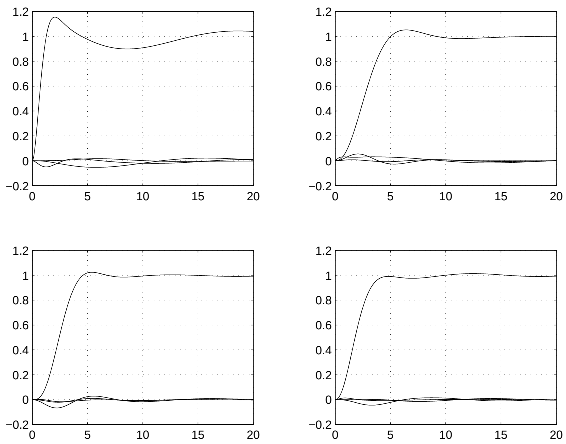

Figure 24: Step response of the closed-loop system

Figures 25, 26 and 27 represent important frequency-domain characteristics of the closed-loop system: the singular values of the sensitivity matrix $S = (I + GK)^{-1}$, the singular values of the complementary sensitivity matrix $T = GK(I + GK)^{-1}$ and the singular values of the matrix $KS = K(I + GK)^{-1}$, respectively, as functions of the frequency.

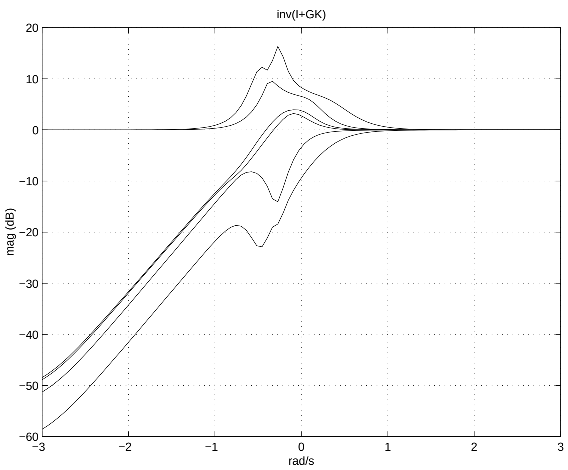

Figure 25: Singular values of S

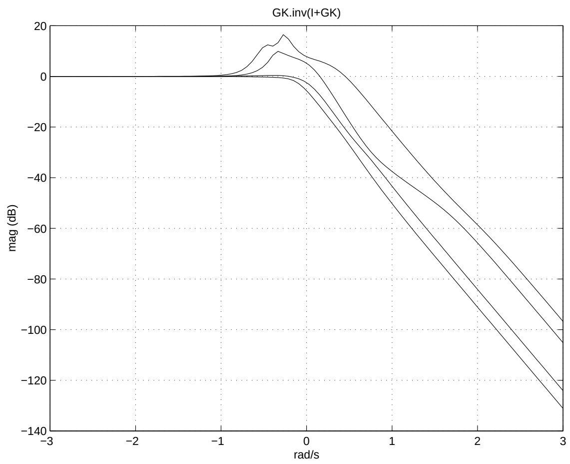

Figure 26: Singular values of T

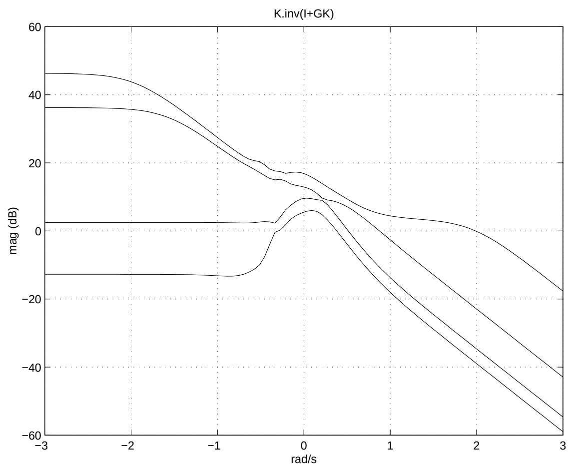

Figure 27: Singular values of KS

5.2 Design of $\mathcal{H}_\infty$ suboptimal controller for distillation column

In this example we consider the design of a continuous-time $\mathcal{H}_\infty$ suboptimal controller for a distillation column. The column is described by nonlinear differential equations of 82-nd order [11], which are linearized around the working point. Here we shall use a reduced order model of 17-th order whose dynamic behaviour is very close to the behaviour of the full model. The plant has six inputs and five outputs. The design is done using the mixed sensitivity formulation described in the previous example. The weighting matrices are diagonal and their elements are first order transfer functions. The singular values of the weighting matrices W_1 and W_2 are shown in Fig. 28 and Fig. 29, respectively.

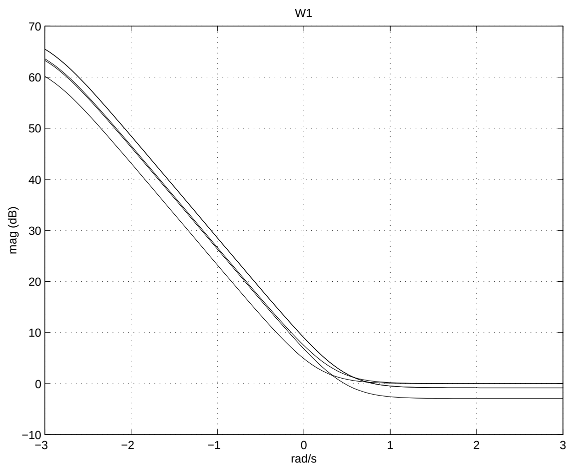

Figure 28: Singular values of W_1

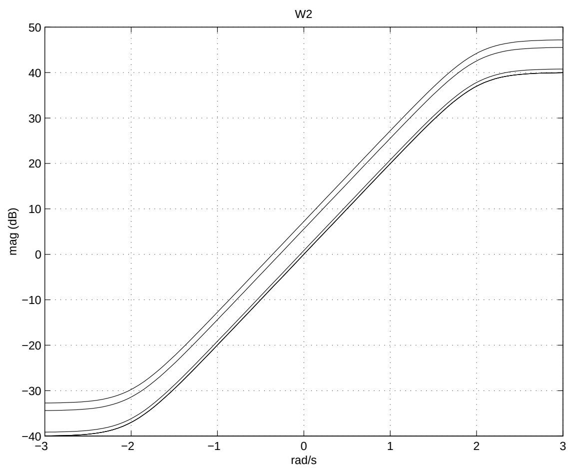

Figure 29: Singular values of W_2

The augmented model is of 28-th order which gives rise to a controller of the same order. The solution of the standard problem is obtained by the mex-function `conhin` for $gamma = 2.9$. It is interesting to note that the μ-toolbox of MATLAB was unable to produce a solution for this problem.

In Figure 30 we present the step-responses on the first four outputs of the closed-loop system.

(The behaviour of the fifth output is similar).

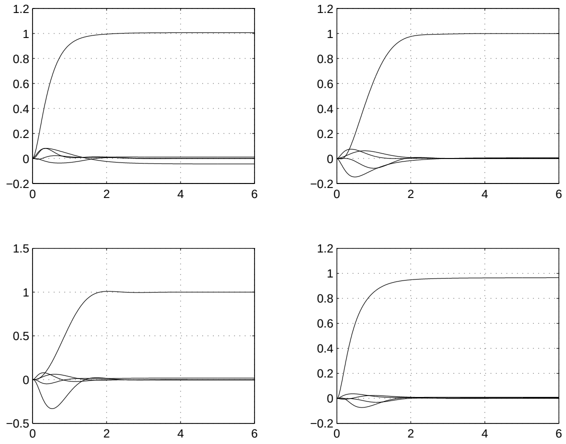

Figure 30: Step response of the closed-loop system

Figures 31, 32 and 33 represent the frequency-domain characteristics of the 45-th order closed-loop system: the singular values of the sensitivity matrix S, the complementary sensitivity matrix T and the matrix KS, respectively, as functions of the frequency. These characteristics show that the requirements of the closed-loop system dynamics are fulfilled.

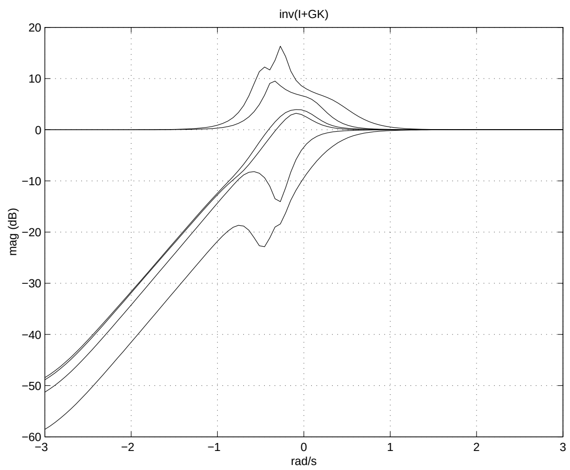

Figure 31: Singular values of S

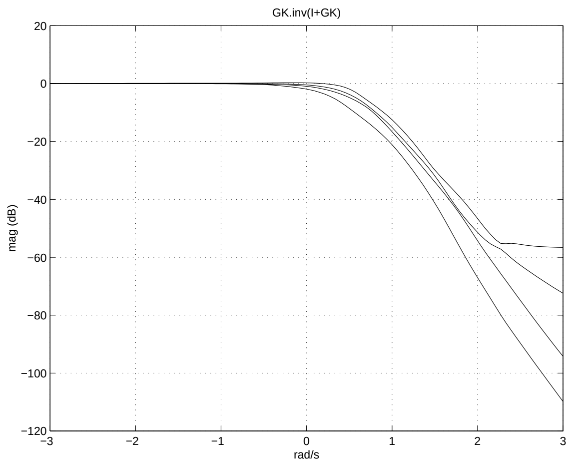

Figure 32: Singular values of T

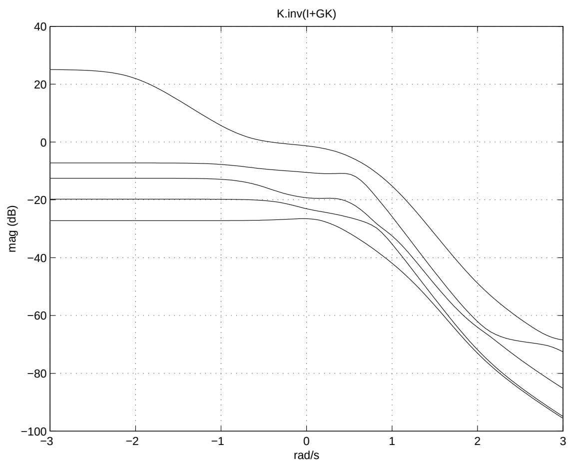

Figure 33: Singular values of KS

6 Summary of achieved results and perspectives

The $\mathcal{H}_\infty/\mathcal{H}_2$ design tools in SLICOT consist of a functionally complete collection of standardized Fortran 77 routines for the computation of (sub)optimal controllers for continuous-time and discrete-time systems. The implementation of the algorithms guarantees that these routines will always produce more accurate solutions and will be faster than the corresponding MATLAB functions from the μ-toolbox. In addition, the SLICOT routines for $\mathcal{H}_\infty/\mathcal{H}_2$ design produce estimates of the reciprocal condition numbers of the matrices whose inversion determines the accuracy of the results as well as estimates of the reciprocal condition numbers of the matrix algebraic Riccati equations involved in the design. In this way the user has the opportunity to monitor the numerical behaviour of the corresponding algorithms in the solution of the given problem and to obtain an idea what is going on "inside" the computations. All routines are carefully documented. The documentation is also available in *html* format. In addition, the documentation includes a test program example, test data and the corresponding results. The routines have been used in the design of several high-order industrial problems.

Apart from the standardized Fortran routines, the interface between the SLICOT $\mathcal{H}_\infty/\mathcal{H}_2$ tools and MATLAB is also provided. Three *mex*-functions, namely `conhin`, `dishin` and `hinorm` have been developed which allow the user to access the developed routines in the SLICOT package. The *mex*-functions are fully documented according to the MATLAB standard.

The further enhancement of the $\mathcal{H}_\infty/\mathcal{H}_2$ SLICOT tools will include development of routines for solving other important $\mathcal{H}_\infty$ design problems. Additional efforts will be devoted to the improvement of the efficiency and numerical reliability of the corresponding routines.

References

[1] D.-W. Gu, P.Hr. Petkov and M.M. Konstantinov. "Direct Formulae for the $\mathcal{H}_\infty$ Sub-Optimal Central Controller". *Niconet Report*, **No. 1998-7**, 1998.

[2] P.Hr. Petkov, D.-W. Gu and M.M. Konstantinov. "Fortran 77 Routines for $\mathcal{H}_\infty$ and $\mathcal{H}_2$ Design of Continuous-Time Linear Control Systems". *Niconet Report*, **No. 1998-8**, 1998.

[3] D.-W. Gu, P.Hr. Petkov and M.M. Konstantinov. "An Introduction to $\mathcal{H}_\infty$ Optimisation Designs". *Niconet Report*, **No. 1999-4**, 1999.

[4] P.Hr. Petkov, D.-W. Gu and M.M. Konstantinov. "Fortran 77 Routines for $\mathcal{H}_\infty$ and $\mathcal{H}_2$ Design of Linear Discrete-Time Control Systems". *Niconet Report*, **No. 1999-5**, 1999.

[5] P.Hr. Petkov, M.M. Konstantinov, D.-W. Gu and V. Mehrmann. "Numerical Solution of Matrix Riccati Equations: A Comparison of Six Solvers". *Niconet Report*, **No. 1999-10**, 1999.

[6] A.A. Stoorvogel. "Numerical Problems in Robust and $\mathcal{H}_\infty$ Optimal Control". *Niconet Report*, **No. 1999-13**, 1999.

[7] 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*. 2nd Edition. SIAM, Philadelphia, PA, 1995.

[8] G.J. Balas, J.C. Doyle, K. Glover, A. Packard and R. Smith. *μ-Analysis and Synthesis Toolbox User's Guide.* The MathWorks Inc., Natick, MA, 1998.

[9] K.Glover and J.C. Doyle. State-Space Formulae for all Stabilizing Controllers that Satisfy an H_∞ Norm Bound and Relations to Risk Sensitivity. *Syst. Contr. Lett.*, vol. 11, 1989, pp. 167–172.

[10] M. Green and D.J.N. Limebeer. *Linear Robust Control*, Prentice-Hall, Englewood Cliffs, NJ, 1995.

[11] S. Skogestad. Dynamics and Control of Distillation Columns. A Tutorial Introduction. *Trans. IChemE*, vol.75, Part A, Sept. 1997.

[12] K. Zhou, J.C. Doyle and K. Glover. *Robust and Optimal Control.* Prentice-Hall, Upper Saddle River, NJ, 1996.

A MATLAB interface for $\mathcal{H}_\infty/\mathcal{H}_2$ design of (sub)optimal controllers for continuous-time systems

```
% CONHIN.F - Mex-function for H_infinity or H_2 design of
%            continuous-time systems based on SLICOT routines
%
%   [AK,BK,CK,DK,(RCOND)] = conhin(task,A,B,C,D,ncon,nmeas,(gamma))
%
%   task = 1 :  [AK,BK,CK,DK,(RCOND)] = conhin(1,A,B,C,D,ncon,nmeas,
%                                              gamma)
%     To compute the matrices of an H-infinity (sub)optimal n-state
%     controller
%
%               | AK | BK |
%           K = |----|----|,
%               | CK | DK |
%
%     for the continuous-time system
%
%               | A  | B1  B2  |   | A | B |
%           P = |----|---------| = |---|---|
%               | C1 | D11 D12 |   | C | D |
%               | C2 | D21 D22 |
%
%     and for a given value of gamma, where B2 has column size of the
%     number of control inputs (ncon) and C2 has row size of the number
%     of measurements (nmeas) being provided to the controller.
%
%   task = 2 :  [AK,BK,CK,DK,(RCOND)] = conhin(2,A,B,C,D,ncon,nmeas)
%
%     To compute the matrices of the H2 optimal n-state controller
%
%               | AK | BK |
%           K = |----|----|
%               | CK | DK |
%
%     for the continuous-time system
```

```
%
%              | A  | B1  B2  |   | A | B |
%          P = |----|---------| = |---|---|
%              | C1 |  0  D12 |   | C | D |
%              | C2 | D21 D22 |
%
%     where B2 has column size of the number of control inputs (ncon)
%     and C2 has row size of the number of measurements (nmeas) being
%     provided to the controller.
%
% Input parameters:
%   task  - integer option to determine the type of the design:
%           = 1 : H_infinity design;
%           = 2 : H_2 design;
%   A     - the n-by-n system state matrix A.
%   B     - the n-by-m system input matrix B.
%   C     - the p-by-n system output matrix C.
%   D     - the p-by-m system matrix D.
%   ncon  - the number of control inputs. m >= ncon >= 0,
%           m-nmeas >= ncon.
%   nmeas - the number of measurements. p >= nmeas >= 0,
%           m-ncon >= nmeas.
%   gamma - (task 1 only) the parameter gamma used in H_infinity design.
%           It is assumed that gamma is sufficiently large so that the
%           controller is admissible. gamma >= 0.
%
% Output parameters:
%   AK    - the n-by-n controller state matrix AK.
%   BK    - the n-by-nmeas controller input matrix BK.
%   CK    - the ncon-by-n controller output matrix CK.
%   DK    - the ncon-by-nmeas controller matrix DK.
%   RCOND - (optional) a vector containing estimates of the reciprocal
%           condition numbers of the matrices which are to be inverted
%           and estimates of the reciprocal condition numbers of the
%           Riccati equations which have to be solved during the
%           computation of the controller. (See the description of
%           the algorithm in [1].)
%           RCOND(1) contains the reciprocal condition number of the
```

```
%                    control transformation matrix,
%           RCOND(2) contains the reciprocal condition number of the
%                    measurement transformation matrix,
%           RCOND(3) contains an estimate of the reciprocal condition
%                    number of the X-Riccati equation,
%           RCOND(4) contains an estimate of the reciprocal condition
%                    number of the Y-Riccati equation.
%
% References
%   [1] P.Hr. Petkov, D.W. Gu and M.M. Konstantinov. Fortran 77 routines
%       for Hinf and H2 design of continuous-time linear control systems.
%       Report98-14, Department of Engineering, Leicester University,
%       April 1999.
```

B MATLAB interface for $\mathcal{H}_\infty/\mathcal{H}_2$ design of (sub)optimal controllers for discrete-time systems

```
% DISHIN.F - Gateway function for H_infinity or H_2 design of
%            discrete-time systems based on SLICOT routines
%
%   [AK,BK,CK,DK,(RCOND)] = dishin(task,A,B,C,D,ncon,nmeas,(gamma))
%
%   task = 1 :  [AK,BK,CK,DK,(RCOND)] = dishin(1,A,B,C,D,ncon,nmeas,
%                                              gamma)
%
%     To compute the matrices of an H-infinity (sub)optimal n-state
%     controller
%
%              | AK | BK |
%          K = |----|----|,
%              | CK | DK |
%
%     for the discrete-time system
%
%              | A  | B1  B2  |   | A | B |
%          P = |----|---------| = |---|---|
%              | C1 | D11 D12 |   | C | D |
%              | C2 | D21 D22 |
%
%     and for a given value of gamma, where B2 has column size of the
%     number of control inputs (ncon) and C2 has row size of the number
%     of measurements (nmeas) being provided to the controller.
%
%   task = 2 :  [AK,BK,CK,DK,(RCOND)] = dishin(2,A,B,C,D,ncon,nmeas)
%
%     To compute the matrices of the H2 optimal n-state controller
%
%              | AK | BK |
%          K = |----|----|
%              | CK | DK |
%
```

```
%     for the discrete-time system
%
%               | A  | B1  B2  |   | A | B |
%           P = |----|---------| = |---|---|
%               | C1 |  0  D12 |   | C | D |
%               | C2 | D21 D22 |
%
%     where B2 has column size of the number of control inputs (ncon)
%     and C2 has row size of the number of measurements (nmeas) being
%     provided to the controller.
%
% Input parameters:
%   task  - integer option to determine the type of the design:
%           = 1 : H_infinity design;
%           = 2 : H_2 design;
%   A     - the n-by-n system state matrix A.
%   B     - the n-by-m system input matrix B.
%   C     - the p-by-n system output matrix C.
%   D     - the p-by-m system matrix D.
%   ncon  - the number of control inputs. m >= ncon >= 0,
%           m-nmeas >= ncon.
%   nmeas - the number of measurements. p >= nmeas >= 0,
%           m-ncon >= nmeas.
%   gamma - (task 1 only) the parameter gamma used in H_infinity design.
%           It is assumed that gamma is sufficiently large so that the
%           controller is admissible. gamma >= 0.
%
% Output parameters:
%   AK    - the n-by-n controller state matrix AK.
%   BK    - the n-by-nmeas controller input matrix BK.
%   CK    - the ncon-by-n controller output matrix CK.
%   DK    - the ncon-by-nmeas controller matrix DK.
%   RCOND - (optional) a vector containing estimates of the reciprocal
%           condition numbers of the matrices which are to be inverted
%           and estimates of the reciprocal condition numbers of the
%           Riccati equations which have to be solved during the
%           computation of the controller. (See the description of
%           the algorithm in [1].)
```

```
%          If task = 1 then
%            RCOND(1) contains the reciprocal condition number of the
%            matrix R3,
%            RCOND(2) contains the reciprocal condition number of the
%            matrix R1 - R2'*inv(R3)*R2,
%            RCOND(3) contains the reciprocal condition number of the
%            matrix V21,
%            RCOND(4) contains the reciprocal condition number of the
%            matrix St3,
%            RCOND(5) contains the reciprocal condition number of the
%            matrix V12,
%            RCOND(6) contains the reciprocal condition number of the
%            matrix Im2 + DKHAT*D22
%            RCOND(7) contains the reciprocal condition number of the
%            X-Riccati equation,
%            RCOND(8) contains the reciprocal condition number of the
%            Z-Riccati equation.
%
%          If task = 2 then
%            RCOND(1) contains the reciprocal condition number of the
%            control transformation matrix TU,
%            RCOND(2) contains the reciprocal condition number of the
%            measurement transformation matrix TY.
%            RCOND(3) contains the reciprocal condition number of the
%            matrix Im2 + B2'*X2*B2,
%            RCOND(4) contains the reciprocal condition number of the
%            matrix Ip2 + C2*Y2*C2',
%            RCOND(5) contains the reciprocal condition number of the
%            X-Riccati equation,
%            RCOND(6) contains the reciprocal condition number of the
%            Y-Riccati equation,
%            RCOND(7) contains the reciprocal condition number of the
%            matrix Im2 + DKHAT*D22 .
%
% References
%   [1] P.Hr. Petkov, D.W. Gu and M.M. Konstantinov. Fortran 77 routines
%       for Hinf and H2 design of discrete-time linear control systems.
%       Report99-8, Department of Engineering, Leicester University,
```

```
%       April 1999.
```

C MATLAB interface for computation of the $\mathcal{H}_\infty$ norm of a continuous-time system

```
% HINORM.F - Gateway function for computation of the H_infinity norm of
%            a continuous-time system based on SLICOT routines
%
%   hfnorm = hinorm(A,B,C,D,(tol))
%
% Purpose:
%   To compute the H-infinity norm of the continuous-time system
%
%                          | A | B |
%                   G(s) = |---|---| .
%                          | C | D |
%
%   It is assumed that the system is asymptotically stable.
%
% Input parameters:
%   A      - the n-by-n system state matrix A.
%   B      - the n-by-m system input matrix B.
%   C      - the p-by-n system output matrix C.
%   D      - the p-by-m system matrix D.
%   tol    - (optional) tolerance used to set the accuracy in
%            determining the norm.
%            default: sqrt(epsilon_machine) where epsilon_machine is
%            the relative machine precision.
%
% Output parameter:
%   hfnorm - the value of the H_infinity norm of the system.
```

D MATLAB m-file for determining the exact condition number of a continuous-time Riccati equation

```
%function [cond,Omega,Theta,Pi] = cndricc(A,C,D,X)
%CNDRICC Quantities related to the conditioning of the
%        continuous-time matrix algebraic Riccati equation
```

```
%
%                    A'*X + X*A + C - X*D*X = 0.
%
%         The condition number of Riccati equation is given by
%
%         cond = norm([Theta*norm(A,'fro'), Omega*norm(C,'fro'),
%                      -Pi*norm(D,'fro')])/norm(X,'fro')
%
%         where Omega, Theta and Pi are defined by
%
%         Omega = inv(kron(eye(n),Ac') + kron(Ac',eye(n))),
%         Theta = Omega*(kron(eye(n),X) + kron(X,eye(n))*W),
%         Pi = Omega*kron(X,X), Ac = A-D*X
%
%         and W is the vec-permutation matrix.
%

%         01.01.1999
%

function [cond,Omega,Theta,Pi] = cndricc(A,C,D,X)
n = max(size(A));
nora = norm(A,'fro');
norc = norm(C,'fro');
nord = norm(D,'fro');

Ac = A - D*X;
M = kron(eye(n),Ac') + kron(Ac',eye(n));

Omega = inv(M);

W = 0*eye(n*n);
for i = 1:n,
    for j = 1:n,
    W(j+(i-1)*n,i+(j-1)*n) = 1.;
    end
end
```

```
Theta = M\(kron(eye(n),X) + kron(X,eye(n))*W);
Pi = M\kron(X,X);

D1 = norc*Omega;
D2 = nora*Theta;
D3 = nord*Pi;
norx = norm(X,'fro');
if norx > 0
   cond = norm([D1, D2, -D3]) / norx;
else
   cond = 0;
end
```