5

ISSN 1373-6787

July 2000

NICONET Newsletter

Distributed by: **Working Group on Software WGS**

Contents: **Page**

Contact address of the WGS: Mrs. Ida Tassens, Secretary of WGS
Katholieke Universiteit Leuven
Dept. of Electrical Engineering (ESAT-SISTA/COSIC)
Kardinaal Mercierlaan 94
3001 Leuven-Heverlee, Belgium
email: ida.tassens@esat.kuleuven.ac.be
phone: + 32 16 32 17 09 and fax: + 32 16 32 19 70

1 Editorial

Welcome to the fifth issue of the NICONET newsletter which informs you about the evolution of the SLICOT library and its integration in user-friendly environments such as `Scilab` and MATLAB, as well as about other NICONET activities related to CACSD software developments.

Sections 3 to 7 present as usual the new updates of the SLICOT library in subfields of systems and control. In Section 8, Pierre-Olivier Malaterre describes his experience in using SLICOT for polynomial factorization of large MIMO systems. We are grateful to him and hope that other readers will follow his example by sending us a report on their own experience with SLICOT. Smaller reports or short notes will be published in our NICONET E-letter while longer reports, such as the report of P.-O. Malaterre, will enter our newsletter.

In Section 9, the importance of SLICOT in industrial control is discussed. In particular, Dr. Ralph Paul from DaimlerChrysler Aerospace describes how the newly developed SLICOT model reduction toolbox can be integrated into the Xmath/MatrixX environment and what the benefits are for control system design and analysis in his company. Section 10 gives more details about the newest additions to the SLICOT library, new reports and forthcoming events. Finally, Section 11 includes our the announcement and first call for papers of the third NICONET workshop to be held on January 19, 2001, in hotel Le Relais Mercure, Louvain-la-Neuve, in Belgium.

I hope you enjoy reading this newsletter.

Sabine Van Huffel
NICONET coordinator

2 The SLICOT benchmark library

The SLICOT benchmark library, an important tool for the development, analysis and testing of numerical methods and codes for the solution of control problems is constantly improved and updated. Currently there are 6 major collections, 3 for continous-time and 3 for discrete-time problems.

Release 2.0 of the benchmark collections for Riccati equations (BB01AD, BB02AD) has recently been issued. For details see the recent working notes SLWN1999-14, SLWN1999-16 by Jörn Abels and Peter Benner.

Currently a collection for benchmark examples for the control and simulation of robots is under development. In addition, a NICONET report describing benchmarks for subspace identification will appear very soon.

Volker Mehrmann

3 Basic numerical SLICOT tools for control

3.1 Task I.A : Standard and generalized state space systems and transfer matrix factorizations

This was the first task of the NICONET project to be completed and its final report can be found in the previous newsletter. It was explained there that despite the termination of this task, future extensions would be considered depending on the needs of the other NICONET tasks. The following user-callable routines were added to the basic software tools :

AB07ND: compute the inverse (Ai,Bi,Ci,Di) of a given system (A,B,C,D).
MB02VD: solve X*A = B or X*A' = B.
SB02PD: solve continuous-time Riccati equations by the matrix sign function and estimates the reciprocal condition number and forward error bound.
SB04PD: solve continuous-time or discrete-time Sylvester equations using a Schur form-based method.
SB04QD: solve discrete-time Sylvester equations using the Hessenberg-Schur method.

In addition 12 lower level routines were fully documented and standardized. They are used for the solution of Sylvester equations and systems of equations with matrices with few sub-diagonals.

3.2 Task I.B : Structured matrix decompositions and perturbations

Structured matrices refer here to matrices with Toeplitz, Hankel, Hamiltonian, symplectic or cyclic structure. In systems and control of dynamical systems these are encountered in three areas : identification (decompositions of Hankel and Toeplitz matrices), analysis and design (Hamiltonian and symplectic eigenvalue problems) and periodic systems (eigenvalue problems with cyclic structure). Periodic systems and Hamiltonian and symplectic eigenvalue problems were already handled in Task I.A. We therefore focus on Hankel and Toeplitz matrices. SLICOT Working Note SLWN2000-2 describes an updated selection of routines for this subtask, which we briefly survey below.

Let T be an $n \times n$ positive definite Toeplitz matrix with blocks of size $k \times k$ and let Z be a block shift matrix of compatible dimension :

$$T = \begin{bmatrix} T_0 & T_1 & \dots & T_{n-1} \\ T_1^T & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & T_1 \\ T_{n-1}^T & \dots & T_1^T & T_0 \end{bmatrix}, \quad Z_k = \begin{bmatrix} 0 & I_k & & \\ & \ddots & \ddots & \\ & & \ddots & I_k \\ & & & 0 \end{bmatrix}. \tag{1}$$

The displacement of T is defined as $\nabla T = T - Z^T T Z$ and this matrix has a symmetric factorization of the type

$$\nabla T = G^T \Sigma G, \quad \Sigma \doteq \begin{bmatrix} I_p & 0 \\ 0 & -I_q \end{bmatrix}, \tag{2}$$

where rank$G \doteq p + q \leq 2k$ is small compared to nk. The low rank matrix G is also called the *generator* of T. The generalized Schur algorithm computes the Cholesky factorization $T = U^T.U$ in $O(n^2k^3)$ flops rather than $O(n^3k^3)$ and it is even more economical when T is banded or of low rank. It can also be adapted to compute an incomplete factorization of embedded matrices of the type :

$$\begin{bmatrix} T & I \\ I & 0 \end{bmatrix} = \begin{bmatrix} U^T \\ L^T \end{bmatrix} \begin{bmatrix} U & L \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & -T^{-1} \end{bmatrix}.$$

This constructs both factorizations $T = U^T.U$ and $T^{-1} = L^T.L$ as well as the generator for T^{-1}. If applying the same ideas to the non-symmetric matrix

$$\begin{bmatrix} T & -b \\ I & 0 \end{bmatrix},$$

one easily checks that the Schur complement is $T^{-1}b$, i.e. the solution of the system of equations $Tx = b$. A MATLAB routine TOEPI implementing these ideas was applied to several block Toeplitz matrices with normally distributed random entries ($\sim N(0,1)$). The following errors were computed

$$\begin{aligned} e_U &= \|U^T U - T\| / \|T\|, \\ e_L &= \|LTL^T - I\|, \\ e_I &= \|T_i T - I\|, \quad (T_i \text{ is the inverse of } T \text{ computed from its generator}), \\ e_{\text{chol}} &= \|U^T U - T\| / \|T\|, \quad (\text{using } \texttt{chol} \text{ of MATLAB© to compute } U). \end{aligned}$$

Furthermore, we give the execution time of TOEPI in comparison with the LAPACK routine DPOTRF.

k	n	e_U	e_L	e_I	e_{chol}	t_{TOEPI}	t_{DPOTRF}
1	1000	1.14e-13	4.68e-15	5.53e-15	1.13e-15	0.72s	4.05s
2	500	1.07e-13	4.32e-15	2.01e-14	2.03e-15	0.74s	4.05s
20	50	5.17e-13	3.22e-15	1.48e-14	2.50e-14	1.13s	4.04s
50	20	1.32e-12	4.89e-15	3.14e-14	4.90e-14	1.70s	4.06s

A related question is that of finding the QR factorization $T = QR$ of an $m \times n$ block Toeplitz matrix T with blocks of size $k \times l$ (the block Hankel case is similar) :

$$T = \begin{bmatrix} T_0 & T_1 & \dots & T_{n-1} \\ T_{-1} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & T_1 \\ T_{-m+n} & & \ddots & T_0 \\ \vdots & \ddots & & T_1 \\ \vdots & & \ddots & \vdots \\ T_{-m+1} & \dots & \dots & T_{-m+n} \end{bmatrix}.$$

It is known that the matrix $T^T.T$ has a low displacement rank as well, and hence we can apply the Schur algorithm to it. Since $T^T.T = R^T.Q^T.Q.R = R^T.R$, this would only yield the factor R. In order to get both factors one considers the low rank factorization of the bordered matrix

$$\begin{bmatrix} T^T T & T^T \\ T & I \end{bmatrix} = \begin{bmatrix} T^T \\ I \end{bmatrix} \begin{bmatrix} T & I \end{bmatrix} = \begin{bmatrix} R^T \\ Q \end{bmatrix} \begin{bmatrix} R & Q^T \end{bmatrix}.$$

which can be shown to have displacement rank bounded by $2(l+k)$ when using the shift matrix $Z \doteq Z_l \oplus Z_k$. A MATLAB subroutine TOEPQR implementing this was applied to several block Toeplitz matrices with normally distributed random entries ($\sim N(0,1)$). The following errors were computed

$$\begin{aligned} e_R &= \|T^TT - R^TR\|/\|T^TT\|, \\ e_{QR} &= \|T - QR\|/\|T\|, \\ e_Q &= \|I - Q^TQ\|, \\ e_{\text{qr}} &= \|T - QR\|/\|T\|, \quad \text{(using } \texttt{qr} \text{ of MATLAB© to compute } Q \text{ and } R). \end{aligned}$$

Furthermore, we give the execution time of TOEPQR in comparison with the LAPACK routines DGEQRF and DORG2R.

k	l	m	n	e_R	e_{QR}	e_Q	e_{qr}	t_{TOEPQR}	t_{LAPACK}
1	1	1000	1000	5.38e-15	3.07e-15	1.99e-09	3.19e-15	3.36s	56.67s
10	100	100	10	1.62e-15	2.28e-15	5.00e-09	1.57e-15	20.40s	55.39s
100	10	10	100	3.87e-15	1.33e-15	5.94e-08	3.39e-15	28.47s	56.49s
100	100	10	10	3.14e-15	2.83e-15	2.07e-10	2.83e-15	39.62s	56.65s

Another variant of this uses the incomplete factorization of the embedding

$$\begin{bmatrix} T^TT & T^T \\ T & 0 \end{bmatrix} = \begin{bmatrix} R^T \\ Q \end{bmatrix} \begin{bmatrix} R & Q^T \end{bmatrix} - \begin{bmatrix} 0 & 0 \\ 0 & -Q.Q^T \end{bmatrix}.$$

The generator of this has a displacement rank that is slightly higher than the one proposed above, but it can be used to solve systems of least squares solutions since

$$\min_x \|Tx - b\|_2$$

is equivalent to the embedded system of equations

$$\begin{bmatrix} T^TT & T^T \\ T & 0 \end{bmatrix} \begin{bmatrix} x \\ -b \end{bmatrix} = \begin{bmatrix} 0 \\ b \end{bmatrix}.$$

Paul Van Dooren

4 SLICOT tools for controller reduction

High quality numerical software implementing LQG and H_∞-synthesis procedures has been recently developed by NICONET and included in SLICOT. However, these sophisticated controller design approaches lead often to high order controllers, with orders comparable to those of the plants. From a practical point of view simple controllers are preferred over complex ones. The main advantages of simpler controllers are their lower computational complexity allowing higher sampling rates in real-time implementation, and an easier maintenance (e.g. easier bug fix).

Since controllers, seen as systems, can be unstable, methods for reduction of unstable systems can be in principle used also for controller reduction. For example, the modal separation approach in conjunction with balancing related techniques as well as the coprime factorization based approach can be used to compute lower order controllers using the model reduction software available in SLICOT [9]. The main difficulty using these model reduction based approaches is that the presence of the plant in the control loop is completely ignored and the resulting reduced controllers can lead to unsatisfactory closed-loop performance and even to the loss of closed-loop stability.

It follows that the controller reduction problems must be handled distinctly from the open-loop model reduction problems because of the presence of the plant. What is finally important in controller reduction is the preservation of the overall closed-loop performance achieved with the original controller. Two method classes appropriate for controller reduction are considered as basis for robust numerical software for SLICOT: frequency weighting methods [3, 2] and methods using a fractional representation of the controller [4]. A particular case of frequency weighted reduction is the class of relative error methods [5]. Both method classes can address the controller reduction problem trying to preserve the closed-loop stability, the closed-loop performance, and the closed-loop transfer function.

The basis for standardization of the controller reduction routines in SLICOT forms a small set of routines available in the RASP-MODRED library [6] for relative error methods and frequency weighted reduction. However, for some methods of potential interest for practical applications (e.g., the frequency-weighted reduction using Enns's approach [2]), no reliable Fortran software exists. Thus the implementation of several new routines is necessary. The new implementations will rely on the already existing routines in the Model Reduction Toolbox prepared for SLICOT within Task II.A [9].

Task II.B for implementing controller reduction software just started in January 2000. In the first subtask II.B.1 we performed the selection of the routines to be standardized. A first report was delivered that proposed a selection of basic routines for each of the relevant methods (Working Note SLWN1999-18). Here we list the main routines which are presently under implementation.

Several general purpose routines, namely AB09HD, AB09ID, AB09JD and AB09KD, are intended to perform a frequency-weighted model reduction with general weights (one- or double-sided) and for relative error methods. They complement and extend the collection of already available general purpose model reduction routines available in SLICOT.

Additionally, the controller reduction routines SB16AD and SB16BD will be implemented which address explicitly the controller reduction problem using a frequency-weighted model reduction framework with special one- or double-sided weights:

AB09HD	stochastic balancing approaches using the balancing-free square-root approach [10]
AB09ID	frequency-weighted approach using Enns' method [2]
AB09JD	frequency-weighted approach using the improved Enns' method [11]
AB09KD	frequency-weighted Hankel-norm method using the Latham-Anderson approach [3]
SB16AD	controller reduction using frequency-weighted methods of Enns [2] for three special controller reduction problems
SB16BD	state-feedback/full-order estimator based controller reduction using coprime factorization with B&T or SPA methods [1]

References

[1] B. D. O. Anderson and Y. Liu. Controller reduction: concepts and approaches. *IEEE Trans. Autom. Control*, AC-34:802–812, 1989.

[2] D. Enns. *Model Reduction for Control Systems Design.* PhD thesis, Dept. Aeronaut. Astronaut., Stanford Univ., Stanford, CA, 1984.

[3] G. A. Latham and B. D. O. Anderson. Frequency-weighted optimal Hankel norm approximation of stable transfer functions. *Systems & Control Lett.*, 5:229–236, 1985.

[4] Y. Liu and B. D. O. Anderson. Controller reduction via stable factorization and balancing. *Int. J. Control*, 44:507–531, 1986.

[5] M. G. Safonov and R. Y. Chiang. Model reduction for robust control: a Schur relative error method. *Int. J. Adapt. Contr.&Sign. Proc.*, 2:259–272, 1988.

[6] A. Varga. *RASP Model Order Reduction Programs.* University of Bochum and DLR-Oberpfaffenhofen, TR R88-92, August 1992.

[7] A. Varga. Coprime factors model reduction based on accuracy enhancing techniques. *Systems Analysis Modelling and Simulation*, 11:303–311, 1993.

[8] A. Varga. Explicit formulas for an efficient implementation of the frequency-weighted model reduction approach. In *Proc. 1993 European Control Conference, Groningen, NL*, pp. 693–696, 1993.

[9] A. Varga. Model Reduction Routines for SLICOT. NICONET Report 1999-8, June 1999. `ftp://wgs.esat.kuleuven.ac.be/pub/WGS /REPORTS/nic1999-8.ps.Z`.

[10] A. Varga and K. H. Fasol. A new square-root balancing-free stochastic truncation model reduction algorithm. In *Prepr. of 12th IFAC World Congress, Sydney, Australia*, vol. 7, pp. 153–156, 1993.

[11] G. Wang, V. Sreeram, and W. Q. Liu. A new frequency-weighted balanced truncation method and error bound. *IEEE Trans. Autom. Control*, AC-44:1734–1737, 1999.

Andras Varga

5 SLICOT tools for subspace identification

Standard software for linear, time-invariant state space model identification

The scheduled standardization of the planned subspace identification routines for open-loop identification of linear, time-invariant systems has been completed. A list of the routines and a brief description of their functionality is given below. Currently, these routines are being tested on industrial data sets supplied by the industrial partners LMS and IPCOS.

Name	Functionality
IB01AD	possibly sequential triangular compression automatic order detection via SVD
IB01BD	estimate $\begin{bmatrix} A & B \\ C & D \end{bmatrix}$ and $\begin{bmatrix} Q & S \\ S^T & R \end{bmatrix}$ MOESP and N4SID
IB01MD	possibly sequential triangular compression (QR or Cholesky)
IB01MY	possibly sequential triangular compression (fast QR)
IB01ND	SVD of triangular factor for order detection
IB01OD	Order detection based on SVD
IB01OY	Ask for user confirmation in selecting order
IB01PD	Estimate $\begin{bmatrix} A & B \\ C & D \end{bmatrix}$
IB01PX	Calculation of $\begin{bmatrix} B \\ D \end{bmatrix}$ from triangular factor (not recommended)
IB01PY	Calculation of $\begin{bmatrix} B \\ D \end{bmatrix}$ from data (preferred)
IB01QD	Estimate initial state

The documentation of the use of the developed routines, their integration into MATLAB and `Scilab` via mex files will be documented in a SLICOT Working Note which will appear in July, 2000. Also the standardization of the benchmark examples used for evaluating system identification routines will be completed by the same time.

Michel Verhaegen

6 SLICOT tools for robust control

In this period of six months, the robust control sub-group has been concentrating on the selection and standardization of subroutines for the structured singular value $\mu - analysis$ and $\mathcal{H}_\infty$ Loop Shaping Design Procedure (LSDP) controller synthesis in the discrete-time case. An additional subroutine for the computation of solutions to algebraic Riccati equations (ARE) using the matrix sign function method has also been developed. The following table summarises the user-callable routines developed so far concerning the above sub-tasks.

Name	**Function**
SB10KD	Driver subroutine for discrete-time $\mathcal{H}_\infty$ LSDP controller
AB13MD	Computation of an upper bound of the structured singular value μ

The following gateway function routine has also been developed.

Name	**Function**
MUCOMP	Gateway function for computing μ

In addition, another algebraic Riccati equation solver has been developed.

Name	**Function**
SB02PD	Computation of the algebraic Riccati equation solution using the matrix sign function method

The above subroutines have been tested with numerical examples. Comparison on numerical features with MATLAB routines has been made. The newly developed Slicot routines perform well in the efficiency and effectiveness. During development, some bugs were found in the MATLAB routines.

Furthermore, a glass tube process control system design problem is being under investigation using the discrete-time $\mathcal{H}_\infty$ loop shaping design procedure routines.

Professors Petko Hr. Petkov and Mihail M. Konstantinov visited the INRIA in April 2000. Together with INRIA and Leicester partners, they developed H_∞ loop shaping design procedure routines for the discrete-time case and μ-calculation routines. Professor Mihail M. Konstantinov and Dr D.-W. Gu attended the Niconet meeting at the DLR, Germany, in the late May. Dr. D.-W. Gu also attended the IFAC Conference on Control Systems Design, held in Bratislava, June 18-20, 2000, where he presented a paper on condition number and error estimations on matrix equations in control systems design which were incorporated in the SLICOT routines.

Da-Wei Gu

7 SLICOT tools for nonlinear systems in robotics

One of the objectives of the NICONET project is to provide the SLICOT numerical software library with tools for nonlinear control systems. In this sense, the objective of this task is to implement a standard interface to the most used integrator packages (ODEPACK, DASSL, DASPK, RADAU5, DGELDA).

SLICOT will deal with the simulation of non-linear control systems which can be described in terms of *ordinary differential equations* (ODEs):

$$\left.\begin{array}{r}\dot{x}(t) = f(x(t), u(t), p, t)\\ y(t) = g(x(t), u(t), p, t)\end{array}\right\}$$

or DAEs,

$$\left.\begin{array}{r}f(\dot{x}(t), x(t), u(t), p, t) = 0\\ y(t) = g(\dot{x}(t), x(t), u(t), p, t)\end{array}\right\}$$

where $x(t)$ is the state vector, $u(t)$ is the input vector, $y(t)$ is the output vector, p is the parameter vector.

As it was presented on previous newsletters, an interface has been designed to compile all the integrator packages on a single entry point. This standard interface has been implemented in both Fortran and MATLAB systems.

For the demonstration of the system, a test-battery has been included, using cases coming from the packages and from the IVPTestSet (http://www.cwi.nl/cwi/projects/IVPtestset/) and the integrator packages. The use of a standardised interface has allowed to compare different integrators by just changing the code number of the integrator package. The selected cases selected were:

6 Ordinary Differential Equations Examples:

Description	Dimension
- Chemikal Akzo Nobel Problem.	6
- Chemical Kinetics LSODE Problem.	3
- Medical Akzo Nobel Problem.	400
- High Irradiance Response Problem.	8
- Ring Modulator Problem.	15

5 Differential Algebraic Equations Examples:

Description	Dimension	Order
- Andrews' Squeezing Mechanism.	2	3
- Car Axis Problem.	10	3
- Transistor Amplifier Problem.	8	1
- LSODI Example Problem.	3	1
- GELDA Example Problem	2	1 (Time-varing coefficients)

A software package with all the examples was created to easily check all the possiblities. The problems are coded in both Fortran77 and MATLAB to test both interface implementations. The results obtained with and without the interface differ around the machine precission.

To provide with results that could be attractive for the industry, a set of robot benchmarks were selected for testing the interface. The examples came from:

- PUMA model developed at the Universidad Politécnica de Valencia. It is a system with six degrees of freedom.
- Robot model coming from the Tetrabot code provided by the Leicester University. This is a 6 DOF robot with parallel links provided by GEC.
- Robot models coming from the robot toolbox[1]. Three industrial robot models are included in this toolbox, two models of a PUMA560 robot and one model of a Stanford manipulator, both with 6 degrees of freedom.
- A three-link window cleaning robot provided by Chemnitz University, obtained from: Iou and and P. Müller, "LW and tracking control of descriptor systems with application to constrained manipulator." Technical report, Sicherheitstechnische Tegelungs- und Messtechnik, Universität Wuppertal, Germany 1994.

The problem solved is the simulation of the dynamical behaviour of the robot arm. A state equation of the robot models determines a differential equation that can be solved with the system.

Vicente Hernandez and Ignacio Blanquer-Espert

[1] see http://www.brb.dmt.csiro.au/dmt/programs/autom/pic/matlab.html

8 Polynomial factorization of large MIMO systems using SLICOT

Abstract
The aim of this note is to present an application of the SLICOT TB03AD routine for polynomial factorization of large MIMO systems. The corresponding polynomial matrix fractions have been calculated for the H, U and V matrix transfer functions coming from the Youla parametrization of a large MIMO system. The H (resp. U and V) matrix transfer function has 5 inputs, 10 outputs and 130 states (resp. 5, 10, 65 for U and 5, 5, 65 for V). The results proved to be very good in terms of singular values matching between the original and the factorized system (polynomial matrix fraction). Scaling has been introduced, reducing considerably the size of the polynomial coefficients, without losing precision for the singular values.

8.1 Introduction

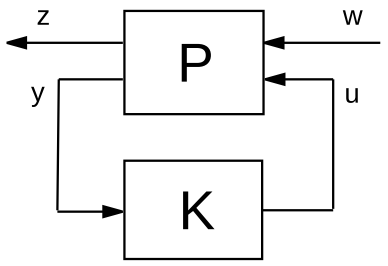

Figure 1: Standard framework

In many control design problems, the objective is to find a stabilizing linear time-invariant (LTI) discrete-time controller K which, in addition to stabilizing the closed loop system, minimizes some norm(s) of the transfer matrix $\Phi : w \to z$ (Figure 1). If the considered norm is the $\mathcal{H}_2$ or $\mathcal{H}_\infty$ norm of the transfer matrix Φ in the frequency domain, then a state-space solution exists for the controller. But in the case of the ℓ_1 norm of the impulse response, no state-space solution exists for the moment, and the problem must be solved by a linear programming approach (see [1]). This can be stated as solving:

$$\gamma^{opt} = \inf_{\text{K stabilizing}} \|F_l(P,K)\|_1 \tag{3}$$

where P represents the LTI discrete-time generalized plant (Figure 1), K the LTI discrete-time controller, $F_l(P,K) = \Phi$ the lower linear fractional transformation of P by K. We assume the dimensions of w, z, u, and y are n_w, n_z, n_u, and n_y respectively. It can be shown (see [1]), that this problem can be formulated as that of finding:

$$\gamma^{opt} = \inf_{Q \in \ell_1^{n_u \times n_y}} \|H - U * Q * V\|_1 \tag{4}$$

	without scaling	with scaling
from state-space	2.8160 10^{-10}	2.8160 10^{-10}
left factorization	9.5064 10^{-6}	9.1206 10^{-6}
right factorization	9.5899 10^{-9}	9.5920 10^{-9}

Table 1: TB03AD: Maximum difference of singular values to the ones obtained with the MatLab “Sigma” function, for U

	without scaling	with scaling	order
N_{ul}	1.7894 10^{13}	0.2767	7
D_{ul}	6.4676 10^{13}	1.0	7
N_{ur}	5.1159 10^{11}	0.6223	13
D_{ur}	8.2215 10^{11}	1.0	13

Table 2: TB03AD: Maximum absolute value of coefficients of the obtained polynomial matrices, and order of the polynomials, for U

where $*$ denotes convolution, $H \in \ell_1^{n_z \times n_w}$, $U \in \ell_1^{n_z \times n_u}$, and $V \in \ell_1^{n_y \times n_w}$ are fixed and depend on the problem data: P, n_w, n_z, n_u, and n_y.

In order to transform this problem into a Finite Dimensional Linear Programing problem (FD LP) one solution is to compute Finite Impulse Response (FIR) approximations of H, U and V, and to look for the optimal Q with a finite support of length n. It is proved (see [2]), that this approach with one additional constraint provides converging upper and lower bounds for the original problem, when $n \to \infty$.

8.2 Incentive for polynomial factorization

The numerical approach used to solve the previous problem can lead to a large number of linear constraints in the LP problem if the FIR approximations of H, U and V are very long. Another approach can be to compute a right (resp. left) polynomial factorization of U (resp. V): $U = N_{ur}.D_{ur}^{-1}$ (resp. $V = D_{vl}^{-1}.N_{vl}$). A similar work can be done with H but will not be detailed in this note. The original problem is then transformed into:

$$\gamma^{opt} = \inf_{Q \in \ell_1^{nu \times ny}} \|H - N_{ur} * D_{ur}^{-1} * Q * D_{vl}^{-1} * N_{vl}\|_1 \tag{5}$$

$$= \inf_{\widetilde{Q} \in \ell_1^{nu \times ny}} \|H - N_{ur} * \widetilde{Q} * N_{vl}\|_1 \tag{6}$$

with $\widetilde{Q} = D_{ur}^{-1} * Q * D_{vl}^{-1}$
Proof: U and V are stable $\Rightarrow$ the λ-Transform $\widehat{D}_{ur}$ and $\widehat{D}_{vl}$ have no unstable zeros $\Rightarrow$ $\widehat{D}_{ur}^{-1}$ and $\widetilde{\widehat{D}_{vl}^{-1}} \in \ell_1$ (Weiner Theorem) and therefore $Q \in \ell_1 \Leftrightarrow \widetilde{Q} \in \ell_1$, which means that searching over $\widetilde{Q}$ is equivalent to searching over Q.

8.3 Results of the polynomial factorization of U

The left and right polynomial factorization $U = D_{ul}^{-1}.N_{ul} = N_{ur}.D_{ur}^{-1}$ have been computed using the SLICOT TB03AD subroutine ([3]). The U system has 5 inputs, 10 outputs and 65 states. The order of the N_{ul} (resp. D_{ul}, N_{ur}, D_{ur}) matrix polynomial is 7 (resp. 7, 13, 13). The singular values of the original system are compared, over a large range of frequencies, to the ones of the left and right polynomial factorizations (Figure 2, Table 1). The matching is very good (the maximum deviation is 9.5064 10^{-6} for the left factorization). Nevertheless, one drawback of this factorization is that the coefficients of the polynomial matrices are very large in our example (Table 2). But this can be easily solved by scaling them (for example by dividing all coefficients by an appropriate scalar, which does not change the condition number of the matrices). The matching of the singular values is not affected (Table 1), and the maximum value of the coefficients of the polynomial matrices is greatly reduced (Table 2). This is important if these coefficients are then used in a Linear Programming software.

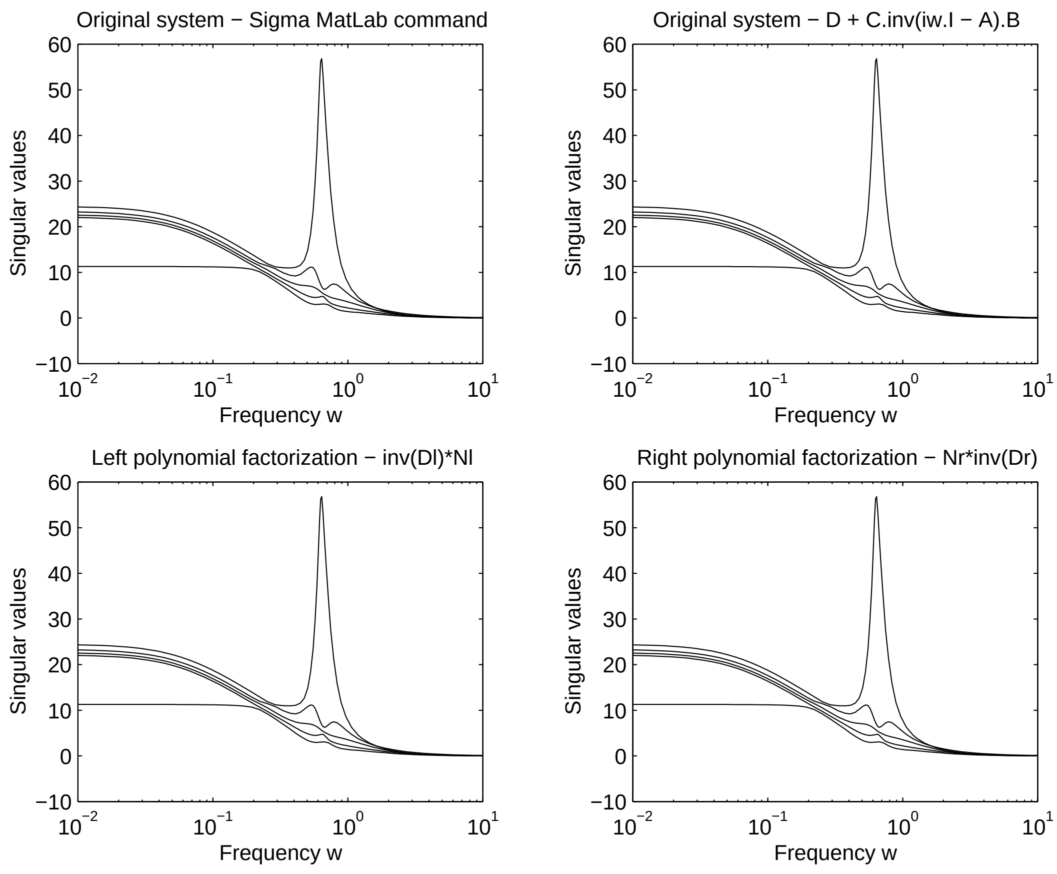

Figure 2: Comparison of the singular values for U

8.4 Comparison with the PolyX ToolBox

The results obtained with the SLICOT TB03AD routine are compared to the ones obtained with the MatLab Polynomial ToolBox PolyX (Tables 3, 4, 5, 6). These results have been

	without scaling	with scaling
from state-space	$2.8160\ 10^{-10}$	$2.8160\ 10^{-10}$
left factorization	$2.7644\ 10^{-6}$	$2.9720\ 10^{-6}$
right factorization	$9.8655\ 10^{-9}$	$9.7450\ 10^{-9}$

Table 3: PolyX: Maximum difference of singular values to the ones obtained with the MatLab "Sigma" function, for U

	without scaling	with scaling	order
N_{ul}	$1.7575\ 10^{13}$	0.2680	7
D_{ul}	$6.5584\ 10^{13}$	1.0	7
N_{ur}	$3.2054\ 10^{11}$	0.4997	13
D_{ur}	$6.4140\ 10^{11}$	1.0	13

Table 4: PolyX: Maximum absolute value of coefficients of the obtained polynomial matrices, for U

kindly provided by PolyX, inc ([4]). The results are very similar, as far as the singular values and the maximum polynomial coefficients are concerned. Except in one case (right factorization for U), PolyX gives slightly better results. Also, with both packages the computation time is about 1 to 2 seconds on a Pentium II, 350 Mhz PC.

8.5 Conclusion

The SLICOT TB03AD subroutine proved to be very useful and numerically efficient, even on large systems. The results are close to the ones obtained using the PolyX MatLab ToolBox, although slightly not as good on 3 cases out of 4. Part of the losses in the numerical precision is maybe due to the way this SLICOT subroutine has been used, i.e. using .mat files and an external .exe program compiled with the PowerStation 4.0 Fortran Compiler. Using a .mex file compiled with the Digital Fortran Compiler is probably a better alternative, in terms of computation time and numerical precision.

The coefficients of the obtained polynomials proved to be very large. This is an important problem since these coefficients are then planned to be used into a Linear Programing software. But, this could be easily solved using a scaling factor. Neither the precision of the polynomial factorization nor the condition number of the matrices have been altered.

	without scaling	with scaling
from state-space	$8.7041\ 10^{-14}$	$8.7041\ 10^{-14}$
left factorization	$2.9665\ 10^{-13}$	$2.9488\ 10^{-13}$
right factorization	$6.8212\ 10^{-13}$	$6.8301\ 10^{-13}$

Table 5: PolyX: Maximum difference of singular values to the ones obtained with the MatLab "Sigma" function, for V

	without scaling	with scaling	order
N_{vl}	$2.1523\ 10^8$	0.0263	13
D_{vl}	$8.1682\ 10^9$	1.0	13
N_{vr}	$5.6355\ 10^6$	0.0264	13
D_{vr}	$2.1380\ 10^8$	1.0	13

Table 6: PolyX: Maximum absolute value of coefficients of the obtained polynomial matrices, for V

8.6 Acknowledgment

The material presented in this note was studied during my stay at ISU as a Visiting Scientist. I would like to express deep gratitude to Mustafa Khammash who kindly welcomed me in his research group. I would also like to thank Cemagref, Montpellier, France for its financial support and my colleagues there that accepted to do part of my share of work during this period.

I also want to thank Vasile Sima, from the Research Institute for Informatics, Bucharest, for his help in correcting bugs in the original version of the TB03AD SLICOT routine ([3]). I also want to thank Michael Sebek from PolyX ([4]), for his help in showing that this problem could be solved numerically at the time the bug in the original TB03AD SLICOT routine was not identified, and proving the quality of the PolyX MatLab Toolbox.

References

[1] M. A. Dahleh and I. J. Diaz-Bobillo, *Control of uncertain systems: a linear programming approach.* Prentice-Hall, 1995.

[2] M. Khammash, "A new approach to the solution of the ℓ_1 control problem: the Scaled-Q method," *IEEE Transaction on Automatic Control*, vol. 45, pp. 180–187, February 2000.

[3] SLICOT homepage http://www.win.tue.nl/niconet/NIC2/slicot.html, 2000.

[4] H. Kwakernaak and M. Sebek, "Polynomial toolbox 2." PolyX Ltd, Prague, Czech Republic, www.polyx.com, www.polyx.cz, 2000.

Pierre-Olivier Malaterre[2]

Department of Electrical and Computer Engineering
Control Group, 3133 Coover Hall
Iowa State University, Ames, Iowa 50011-3060

8.7 Appendix 1 - Files provided

- TB03AD.m: .m file of the MatLab function computing the polynomial factorization (type "help TB03AD" for help on input and output arguments). This TB03AD.m file

[2]Research-Engineer from Cemagref, Montpellier, France and Visiting Scientist at ISU from May 1999 to August 2000, pom@iastate.edu

calls a MTB03AD.exe program compiled from the MTB03AD.f Fortran file described below.

- MTB03AD.f: Fortran source code allowing to generate the MTB03AD.exe program used from the TB03AD.m MatLab function above described. This source code has been compiled using the MicroSoft PowerStation 4.0 Fortran compiler, and linked to the SLICOT, LAPACK and BLAS libraries ([3]). Another approach, probably more efficient, would have been to generate a .mex file using the Digital Fortran Compiler.

- RWMAT.f: Fortran source code of 2 subroutines allowing to read or write a matrix on a .mat MatLab file in MatLab format 4.2.

- RWMAT5.f: Fortran source code of 2 subroutines allowing to read or write a matrix on a .mat MatLab file in MatLab format 5. We used this version on SGI (Unix). It can also be used on PC but only with the Digital Fortran Compiler. In this case, the above MTB03AD.f program has to be slightly modified since the headers of the 2 subroutines are not exactly the same.

- Toto5.m: .m file of the MatLab program used to compute the polynomial factorization of H, U or V and plot the figures given in this note.

- Toto6.m: .m file of a simple MatLab program calling the TB03AD.m function, to compute the polynomial factorization of H, U or V.

- Hsys.mat, Usys.mat and Vsys.mat: .mat files containing the state space representations of the H, U and V systems considered in this note.

8.8 Appendix 2 - Results for H

	without scaling	with scaling
from state-space	$4.7494\ 10^{-10}$	$4.7494\ 10^{-10}$
left factorization	$2.3334\ 10^{-8}$	$2.3342\ 10^{-8}$
right factorization	$1.2744\ 10^{-8}$	$1.2763\ 10^{-8}$

Table 7: TB03AD: Maximum difference of singular values to the ones obtained with the MatLab “Sigma” function, for H

	without scaling	with scaling	order
N_{hl}	$2.5911\ 10^{8}$	0.0028	13
D_{hl}	$9.1911\ 10^{10}$	1.0	13
N_{hr}	$1.5042\ 10^{21}$	0.0202	26
D_{hr}	$7.4527\ 10^{22}$	1.0	26

Table 8: TB03AD: Maximum absolute value of coefficients of the obtained polynomial matrices, and order of the polynomials, for H

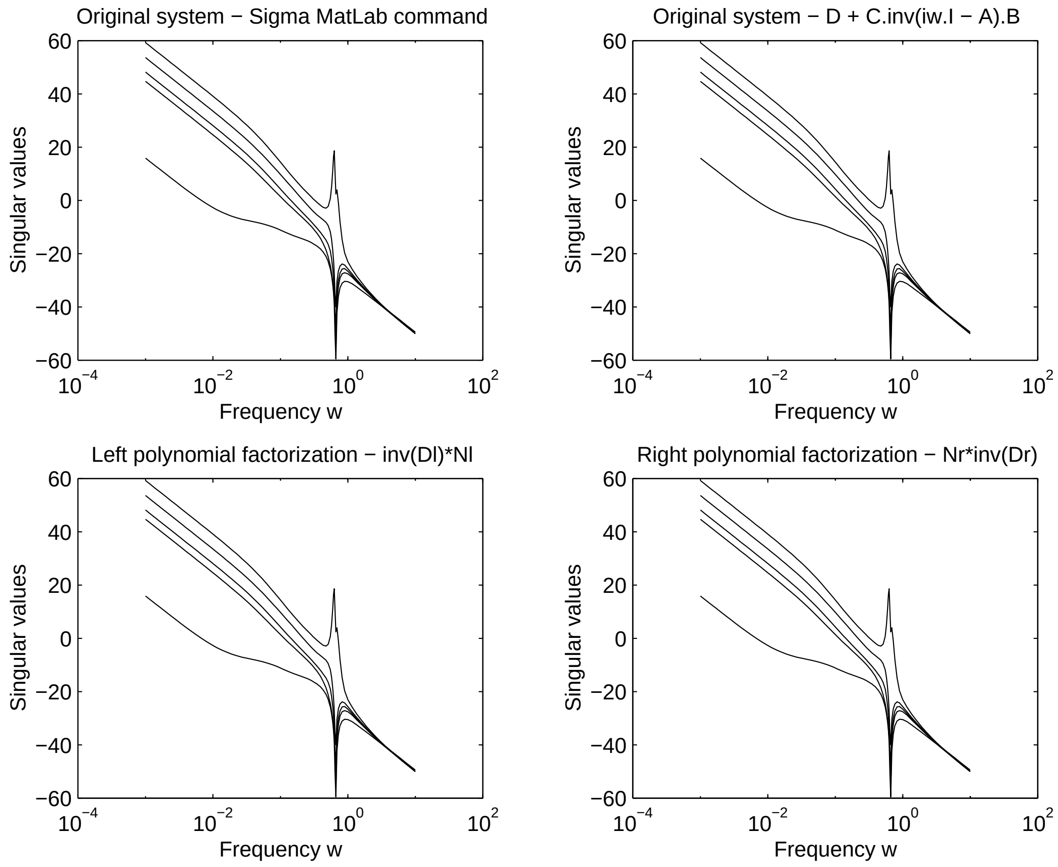

Figure 3: Comparison of the singular values for H

8.9 Appendix 3 - Results for V

	without scaling	with scaling
from state-space	$8.7041\ 10^{-14}$	$8.7041\ 10^{-14}$
left factorization	$1.4602\ 10^{-12}$	$1.4619\ 10^{-12}$
right factorization	$1.0090\ 10^{-12}$	$1.0036\ 10^{-12}$

Table 9: TB03AD: Maximum difference of singular values to the ones obtained with the MatLab "Sigma" function, for V

	without scaling	with scaling	order
N_{vl}	$1.7564\ 10^{8}$	0.0265	13
D_{vl}	$6.6233\ 10^{9}$	1.0	13
N_{vr}	$4.9486\ 10^{6}$	0.0265	13
D_{vr}	$1.8684\ 10^{8}$	1.0	13

Table 10: TB03AD: Maximum absolute value of coefficients of the obtained polynomial matrices, and order of the polynomials, for V

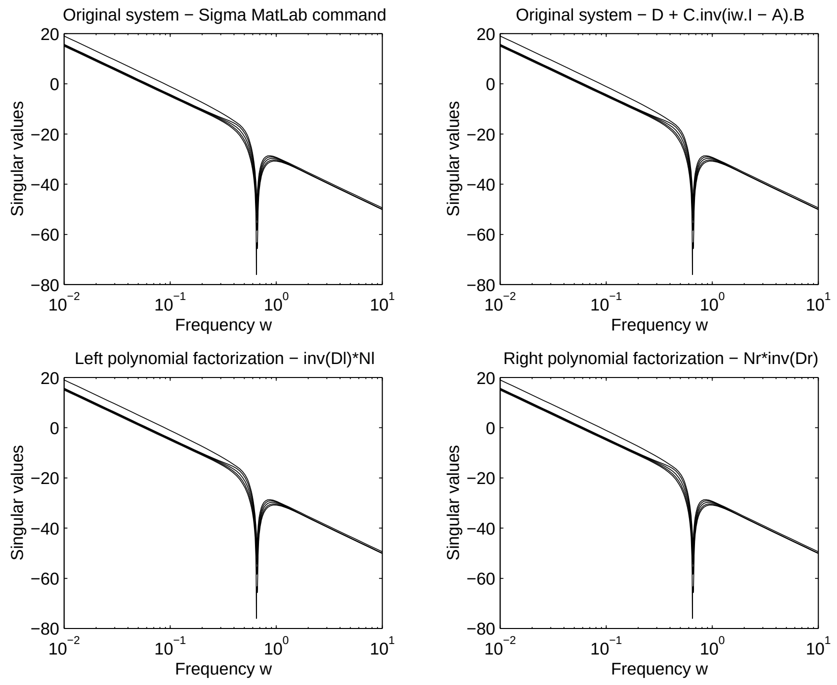

Figure 4: Comparison of the singular values for V

9 SLICOT: a useful tool in industry?

Porting the SLICOT Model Reduction Toolbox to Xmath/MatrixX

9.1 The Design of the Xmath/MatrixX Interface

For the Xmath/MatrixX interface design several constraints had to be accounted for :

- Modification to the SLICOT source code should be avoided or kept to a minimum
- The user should see the model reduction routines as 'just another' toolbox with a familiar interface and feel
- The starting time and memory penalty associated with the use of a Xmath 'LNX' executable (= Matlab MEX file) should be minimized
- The programming language and interface differences between C/C++ (=Xmath) and Fortran (=SLICOT, Matlab MEX Functions) such as 'call by value' ↔'call by reference' and array memory layout 'row order'↔'column order' need to be taken care of and hidden from the user
- The modifications necessary for future updates of the SLICOT toolbox should be simple.

Most of these requirements are already addressed in the existing 'SLICOT to Matlab' interface, therefore I decided to closely follow its layer and gateway design. As a result of this choice, only the layers depicted as gray boxes in figure 5 had to be created (Xmath to Mex Adapter) or adapted from the original Matlab versions.

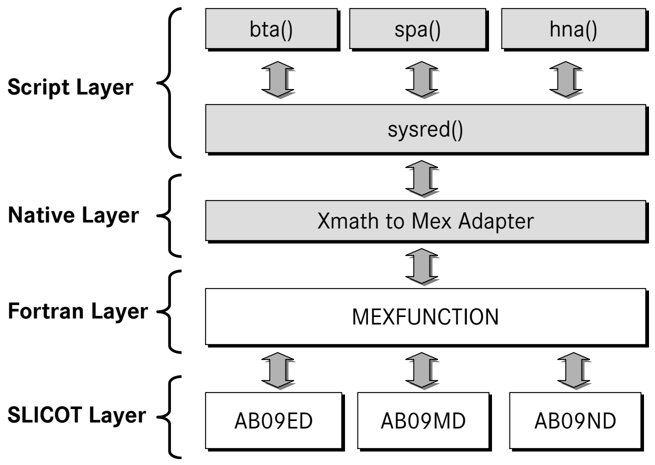

Figure 5: Layer Model of the Xmath/MatrixX Interface

The adaptation of the Matlab scripts turned out to be fairly straight forward and took about 1-2 hours of time. Semantic and/or syntactical differences were easily overcome. Most of this time was actually spent comparing the Matlab and the Xmath/MatrixX reference guides to determine the meaning of several overloaded functions (ex. `norm()`). The reference test results provided as a binary 'MAT' file were converted to 'pure' ASCII form using PC Matlab and some helper scripts.

The implementation of the 'Xmath to Mex' adapter was more difficult and took about 1.5 days. As a first step, a mapping between the Matlab MEX Interface functions and the corresponding Xmath/MatrixX counterparts was defined. For the second step, several implementation related differences had to be assessed and accounted for in the 'Xmath to Mex' Adapter :

- Programming language difference between C/C++ (Xmath) and Fortran (Matlab)
- Differences in internal data structures and external interfaces
- Compaq Tru64 is a 64 bit based operating system (PC = 32bit)

The 'Xmath to Mex' adapter is implemented as a small library written in ANSI-C.

For a more detailed discussion of the 'Matlab to SLICOT' interface or the model reduction toolbox, please refer to references [1], [2].

9.2 Porting SLICOT to Compaq Tru64 UNIX

To be able to port the SLICOT model reduction library to Compaq Tru64 and the Xmath/MatrixX software, the following tools are necessary :

- Compaq Fortran 77, 90/95 and C compiler for Tru64 UNIX
- BLAS and LAPACK libraries Version 3.0 or above
- SLICOT Fortran source code
- Compaq C++ runtime used in the Xmath/MatrixX LNX interface
- The Xmath/MatrixX external interface library

The first step was to obtain a SLICOT compatible binary copy of the BLAS and LAPACK libraries. Unfortunately this turned to be more difficult than expected because the highly optimized versions made available by Compaq as part of their Compaq CXML high performance math library, only support the BLAS/LAPACK version 2.0+ interfaces. Therefore I had to compile BLAS and LAPACK myself using the source code available at 'www.netlib.org'.

After performing several performance tests with different BLAS/LAPACK implementations and CXML, I proceeded to extract the BLAS routines from CXML. The current Xmath SLICOT LNX therefore uses a mixture of the optimized CXML BLAS and the LAPACK routines from netlib.

Compiling and running SLICOT was simple and required about another day of work. The only part that required manual work, were the test cases (also known as 'examples') because in 29 out of 153 tests differences between the resulting output and the reference results were found.

A detailed description of the difference is beyond the scope of this report and can be obtained from the author of this report.

9.3 Preliminary Results

In this section the results of a preliminary evaluation of the SLICOT model reduction toolbox are briefly summarized.

- Numerical Results
  The results obtained so far on typical flight control design and analysis problems have been very promising.

- Ease of Use
  Using the SLICOT model reduction toolbox within Xmath is not different from using the regular Xmath toolbox of the same name. This means that most users will probably never notice that this toolbox is not an 'official' Xmath toolbox. The SLICOT toolbox is even more user friendly because it does not require the original base system to be stable.

- Speed and Performance
  The performance of the SLICOT model reduction routines is at least an order of magnitude better than the normal Xmath toolbox. In the case of the flight control test problems mentioned earlier, this significant performance advantage has resulted in substantial runtime savings for control design and analysis iterations.

- Multi Layer Interface Design
  The original 'SLICOT to Matlab' interface design of the model reduction toolbox is very good and represents the right compromise between resource efficiency (only 1 large binary) and functionality.

- Test and Example Files
  As mentioned in the last section the verification of the SLICOT port resulted in additional manual work. Overall 29 of 153 test results showed differences. Apparently this mostly due to the fact that some of the reference results have not been updated to reflect a LAPACK upgrade to version 3.0+. Therefore simple, tool based or visual comparisons by the user failed to dismiss the differences as insignificant.

- Matlab Script Files
  The process of translating the scripts from Matlab to Xmath syntax and commands could be even further simplified and possibly automated if the original scripts were created using some kind of standard text templates.

9.4 Concluding Remarks

Porting SLICOT to Xmath/MatrixX has been interesting, fun and rewarding. The results I have seen so far look really promising and I am hoping to be able to report more results in the future.

If other Xmath/MatrixX users are interested in using the model reduction toolbox or certain parts of the interface described in this report, please feel free to contact me by email. Hopefully we can then work together to further improve the Xmath/MatrixX support in SLICOT.

References

[1] A. Varga, *Model reduction routines for SLICOT*, NICONET Report 1999-8, December 1999.

[2] Volker Mehrmann, Vasile Sima, Andras Varga and Hongguo Xu, *A MATLAB MEX-file environment of SLICOT*, SLICOT Working Note 1999-11, August 1999.

Ralph Paul

DaimlerChrysler Aerospace GmbH
Military Aircraft
MT62 Flight Dynamics
E-Mail: ralph.paul@m.dasa.de

10 NICONET information corner

This section informs the reader on how to access the SLICOT library, the main product of the NICONET project, and how to retrieve its routines and documentation. Recent updates of the library are also described. In addition, information is provided on the newest NICONET reports, available via the NICONET website or ftp site, as well as information about upcoming workshops/conferences organized by NICONET or with a strong NICONET representation.

Additional information about the NICONET Thematic Network can be found from the NICONET homepage World Wide Web URL

`http://www.win.tue.nl/wgs/niconet.html`

10.1 Electronic Access to the Library

The SLICOT routines can be downloaded from the WGS ftp site,

`ftp://wgs.esat.kuleuven.ac.be`

(directory `pub/WGS/SLICOT/` and its subdirectories) in compressed (gzipped) tar files. On line `.html` documentation files are also provided there. It is possible to browse through the documentation on the WGS homepage at the World Wide Web URL

`http://www.win.tue.nl/wgs/`

after linking from there to the SLICOT web page and clicking on the `FTP site` link in the freeware SLICOT section. The SLICOT index is operational there. Each functional "module" can be copied to the user's current directory, by clicking on an appropriate location in the `.html` image. A "module" is a compressed (gzipped) tar file, which includes the following files: source code for the main routine and its example program, example data, execution results, the associated `.html` file, as well as the source code for the called SLICOT routines.

The entire library is contained in a file, called `slicot.tar.gz`, in the SLICOT root directory `/pub/WGS/SLICOT/`. The following Unix commands should be used for decompressing this file:

```
gzip -d slicot.tar
```

and

```
tar xvf slicot.tar
```

The created subdirectories and their contents is summarized below:

`slicot`	contains the files `libindex.html`, `make.inc`, `makefile`, and the following subdirectories:
`benchmark_data`	contains benchmark data files for Fortran benchmark routines (`.dat`);
`doc`	contains SLICOT documentation files for routines (`.html`);
`examples`	contains SLICOT example programs, data, and results (`.f`, `.dat`, `.res`), and `makefile`, for compiling, linking and executing these programs;
`src`	contains SLICOT source files for routines (`.f`), and `makefile`, for compiling all routines and creating an object library;

`SLTools`	contains Matlab `.m` files, mex files (optionally), and data `.mat` files (optionally);
`SLmex`	contains Fortran source codes for mexfiles (`.f`).

Another, similarly organized file, called `slicotPC.zip`, is available in the SLICOT root directory; it contains the MS-DOS version of the source codes of the SLICOT Library, and can be used on Windows 9x/2000 or NT platforms.

Included are several source makefiles, and executable Matlab files for the PC version (`.dll` files).

After downloading and decompressing the appropriate SLICOT archive, the user can then browse through the documentation on his local machine, starting from the index file `libindex.html` from `slicot` subdirectory.

10.2 SLICOT Library updates in the period January 2000— July 2000

There have been two major SLICOT Library updates during the period January 2000—July 2000: on February 18, and July 1. Another, intermediate update took place on April, when some updated files have been posted on the ftp site root directory (file `Erratum.gz`). These updated files have now been incorporated in the compressed library files. Known bugs have been corrected out on each update. Other routines have been improved. Details are given in the files `Release.Notes` and `Release.History`, located in the directory `pub/WGS/SLICOT/` of the ftp site. Several changes have also occured in the documenting comments of some routines.

Several new user-callable and computational routines for basic control problems, as well as for linear system identification have been made available on the ftp site in the period January 2000—July 2000. They include *Analysis Routines*, *Benchmark and Test Problems*, *Identification Routines*, *Mathematical Routines*, and *Synthesis Routines*, performing the following main computational tasks:

- inverse (A_i, B_i, C_i, D_i) of a given system (A, B, C, D);
- benchmark examples for the numerical solution of continuous-time and discrete-time algebraic Riccati equations;
- preprocessing the input-output data for estimating the matrices of a linear time-invariant dynamical system and finding an estimate of the system order; optionally, the input-output data can be processed sequentially;
- estimating the system matrices A, C, B, and D, the noise covariance matrices Q, R_y, and S, and the Kalman gain matrix K of a linear time-invariant state space model, using the processed triangular factor R of the concatenated block-Hankel matrices, provided by other SLICOT Library routines;
- estimating the initial state and, optionally, the system matrices B and D of a linear time-invariant discrete-time system, given the system matrices (A, B, C, D), or only the matrix pair (A, C), and the input and output trajectories of the system;
- upper triangular factor in the QR factorization of the block-Hankel matrix built from input-output data;

- singular value decomposition giving the system order, using the triangular factor of the concatenated block-Hankel matrices;
- system order, using the singular value decomposition;
- system matrices and covariance matrices of a linear time-invariant state space model;
- estimating B and D matrices using Kronecker products;
- estimating B and D matrices using a structure-exploiting technique;
- estimating the initial state and B and D matrices, given the matrix pair (A, C) and the input and output trajectories;
- estimating the initial state, given the system matrices (A, B, C, D) and the input and output trajectories;
- Kronecker product of two matrices;
- solution of $XA = B$ or $XA^T = B$;
- solution of continuous-time Riccati equations by the matrix sign function method (with condition number and forward error bound estimates);
- solution of either continuous-time or discrete-time general Sylvester equations, using the reduction to real Schur form;
- solution of discrete-time Sylvester equations with matrices in real Schur form (LAPACK-style codes);
- solution of discrete-time Sylvester equations using the Hessenberg-Schur method.

In addition, three new mexfiles and ten Matlab `.m` files have been added in the subdirectories `./SLmex` and `./SLTools`, respectively. They are useful for performing system identification using subspace techniques (MOESP, N4SID, or their combination).

10.3 New NICONET Reports

Recent NICONET reports (available after January 2000), that can be downloaded as compressed postscript files from the World Wide Web URL

`http://www.win.tue.nl/wgs/reports.html`

or from the WGS ftp site,

`ftp://wgs.esat.kuleuven.ac.be`

(directory `pub/WGS/REPORTS/`), are the following:

- Petko Petkov, Da-Wei Gu, Mihail M. Konstantinov, and Volker Mehrmann. *Condition and Error Estimates in the Solution of Lyapunov and Riccati Equations* (file `SLWN2000-1.ps.Z`).

The condition number estimation and the computation of residual based forward error estimates in the numerical solution of matrix algebraic continuous-time and discrete-time Lyapunov and Riccati equations is considered. The estimates implemented involve the solution of triangular Lyapunov equations along with usage of the LAPACK norm estimator. Results from numerical experiments demonstrating the performance of the estimates proposed are presented.

- Daniel Kressner and Paul Van Dooren. *Factorizations and linear system solvers for matrices with Toeplitz structure* (file `SLWN2000-2.ps.Z`).

  This report describes new routines for several factorizations of matrices with Toeplitz or block Toeplitz structure and shows how this can be used to solve the corresponding systems of equations or least squares systems of equations. Certain implementation details are given. Matrices of low rank or of low bandwidth are also considered.

Previous NICONET/WGS reports are also posted at the same address.

10.4 Forthcoming Conferences

Forthcoming Conferences related to the NICONET areas of interest, where NICONET partners submitted proposals for NICONET/SLICOT-related talks and papers, and disseminated or will disseminate information and promote SLICOT, include the following:

- UKACC International Conference CONTROL 2000, University of Cambridge, United Kingdom, September 4–7, 2000.
- IEEE International Symposium on Computer-Aided Control Systems Design, CACSD 2000, Anchorage, Alaska, USA, September 25–27, 2000.

Vasile Sima

11 Upcoming third NICONET workshop in Louvain-la-Neuve, Belgium

Third NICONET WORKSHOP ON

NUMERICAL CONTROL SOFTWARE
a useful tool in industry

Friday, January 19, 2001
Hotel Le Relais Mercure, Louvain-la-Neuve, Belgium.

First Announcement and Call for Posters

This workshop organized by the European Numerics in Control thematic Network (NICONET[3]) aims to bring together engineers, mathematicians, computer scientists and practitioners from industry and academia dealing with numerical software in systems and control and their implementation and use in industrial practice.

Recent advances on the use of numerical software libraries especially designed for solving systems and control engineering problems in a numerically reliable and efficient way will be discussed. The current status of the *freely available* SLICOT library will be extensively discussed, as well as industrial control applications and future extensions, comprising *parallel versions* and *practically oriented benchmarks*. SLICOT is a valuable tool for the reliable solution of many control problems, and for large-scale, computer-intensive control problems and real-time applications, SLICOT can lead to significant performance improvements.

Chairpersons: Paul Van Dooren (local organizer) and Sabine Van Huffel (project coordinator).

Organizing Committee: T. Backx, P. Benner, A. van den Boom, J. De Cuyper, F. Delebecque, D.W. Gu, S. Hammarling, V. Hernández, B. Kågström, M. Konstantinov, V. Mehrmann, A. Moner, P. Petkov, V. Sima, A. Stoorvogel, A. Varga, M. Verhaegen, R. Wohlgemuth.

Workshop Program:

- *Plenary session (9h-12h30):*
  - "Matching prediction error identification and robust control" by Prof. Michel Gevers (Dept. CESAME, Université Catholique de Louvain).
  - "Model reduction within control systems design" by Prof. Maarten Steinbuch (Eindhoven University of Technology).
  - Introductory presentations of the newly developed SLICOT toolboxes
    - "Structured matrix decompositions" presented by Prof. Paul Van Dooren (Université Catholique de Louvain)
    - "Subspace identification" presented by Prof. Michel Verhaegen (Twente University of Technology)
    - "Nonlinear systems for Robotics" presented by Prof. Vicente Hernández (Universidad Politecnica de Valencia).
- *Demo and poster session (14h-16h30)* on new developments and performance presentations of control software in engineering practice and industrial applications.
- *Closing discussion (16h30-17h)*

[3]http://www.win.tue.nl/wgs/niconet.html

The preliminary, as well as the final program, will be announced on the NICONET website[1].

Participants of the workshop receive:

- A copy of the workshop program and proceedings book
- A copy of the last version of the SLICOT Software Library and Toolboxes (upon request)
- Documentation on SLICOT and NICONET.

Prospective authors are invited to submit two copies of a camera-ready paper (1 to 6 pages long), describing the contents of the poster contribution, to the workshop secretariat (see address below) for review. Address and e-mail should be provided if possible. All accepted contributions, as well as documentation for the plenary session, will be published in the workshop proceedings.

Author's Schedule:

Submission of camera-ready paper (1-6 pages):	*December 1, 2000*
Notification of acceptance as poster presentation	*December 15, 2000*

Registration Information:

Please register to the workshop at the address below before *December 15, 2000.* The registration fee is 1500 BEF and covers the attendance to the workshop, the coffee breaks, the lunch, the workshop proceedings, the documentation and the software.

Third NICONET Workshop Registration Form, Friday January 19, 2001:

Name:
Title:
Company/Institution:
Area:
Address:
Phone:
Fax:
e-mail:

Rooms at the conference hotel are available at reduced rates when booked through the workshop secretariat : 2550 BEF for a single room, 2900 BEF for a double room.

Jan. 18 to 19 : single room ☐, double room ☐

Jan. 19 to 20 : single room ☐, double room ☐

to be sent to
Prof. Paul Van Dooren
Secr.: Mrs. Isabelle Hisette
Université Catholique de Louvain, Center for Systems Eng. & Applied Mechanics
4, Avenue Georges Lemaitre, 1348 Louvain-la-Neuve, BELGIUM
Fax: 32-10-47 21 80 and Phone: +32-10-478040 (or +32-10-478034)
URL: http://www.auto.ucl.ac.be/~vdooren/register.html
E-mail: vdooren@csam.ucl.ac.be (or hisette@csam.ucl.ac.be)