6

ISSN 1373-6787

February 2001

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)
Kasteelpark Arenberg 10
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 sixth 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 6 present as usual the new updates of the SLICOT library in subfields of systems and control. In Section 7, a Web computing environment for the SLICOT Library is described, enabling remote users to easily access some basic computational tools from the library. Section 9 gives more details about the newest additions to the SLICOT library, new reports and forthcoming events. Finally, Section 10 includes the announcement of our SLICOT training course in Bremen, Germany, in September 2001.

The NICONET Group organized a special poster session at the UKACC International Control 2000 Conference, held in September at Cambridge, UK. Six papers were displayed and well received.

I hope you enjoy reading this newsletter.

Sabine Van Huffel
NICONET coordinator

2 Basic numerical SLICOT tools for control

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

The final report of Task I.A can be found in Newsletter 4, but as explained there we decided to continue to add new routines if such a need would be justified by the other tasks. The following basic routines were added for Task II, Model/Controller Reduction.

Name	Function
MB01WD	$\bar{R} = \alpha\big(\operatorname{op}(A)^T \operatorname{op}(T)^T \operatorname{op}(T) + \operatorname{op}(T)^T \operatorname{op}(T) \operatorname{op}(A)\big) + \beta R$, or $\bar{R} = \alpha\big(\operatorname{op}(A)^T \operatorname{op}(T)^T \operatorname{op}(T) \operatorname{op}(A) - \operatorname{op}(T)^T \operatorname{op}(T)\big) + \beta R$, where α and β are scalars, R, and $\bar{R}$ are symmetric matrices, T is a triangular matrix, A is a general or Hessenberg matrix, and $\operatorname{op}(M) = M$ or $\operatorname{op}(M) = M^T$.
MB01XD	to compute the matrix product U^TU or LL^T, where U and L are upper and lower triangular matrices, respectively, stored in the corresponding upper or lower triangular part of an array (blocked algorithms).
MB01XY	to compute the matrix product U^TU or LL^T, where U and L are upper and lower triangular matrices, respectively, stored in the corresponding upper or lower triangular part of an array (unblocked algorithms).
MB01YD	to compute the symmetric rank k operations $C \leftarrow \alpha \operatorname{op}(A) \operatorname{op}(A)^T + \beta C$, where α and β are scalars, C is an $n \times n$ symmetric matrix, $\operatorname{op}(A)$ is an $n \times k$ matrix and the matrix A has l nonzero upper or lower codiagonals.
MB01ZD	to compute the matrix product $H \leftarrow \alpha \operatorname{op}(T)H$ or $H \leftarrow \alpha H \operatorname{op}(T)$, where α is a scalar, H is an $m \times n$ upper or lower Hessenberg-like matrix (with l nonzero subdiagonals or superdiagonals, respectively), and T is a unit, or non-unit, upper or lower triangular matrix.

Andras Varga and Paul Van Dooren

2.2 Task I.B : Structured matrix decompositions and perturbations

Task I.B focuses mainly on Hankel and Toeplitz matrices since matrices with cyclic, Hamiltonian and symplectic structure were handled in Task I.A. The following routines were implemented as basic routines :

Name	Function
MB02CD	to compute the Cholesky factor and the generator and/or the Cholesky factor of the inverse of a symmetric positive definite block Toeplitz matrix, defined by either its first block row, or its first block column.
MB02CX	to bring the first blocks of a generator in proper form.
MB02CY	to apply the transformations created by the SLICOT Library routine MB02CX on other columns or rows of the generator.
MB02DD	to update the Cholesky factor and the generator and/or the Cholesky factor of the inverse of a symmetric positive definite block Toeplitz matrix, given the information from a previous factorization and additional blocks of its first block row, or its first block column.
MB02ED	to solve a system of linear equations $TX = B$ or $XT = B$ with a symmetric positive definite block Toeplitz matrix T, defined by its first block row or its first block column.

These Fortran programs are accessed in the interactive environment Matlab or Scilab via the following mex and m-files :

Name	Function
fstoep.f	a mexfile to access all structured matrix routines
fstchol.m	computes the Cholesky factor of a symmetric positive definite (block) Toeplitz matrix and its inverse
fstgen.m	factors a symmetric positive definite (block) Toeplitz matrix and computes the generator of its inverse.
fstsol.m	solves a linear system $XT = B$ or $TX = B$, with T a symmetric positive definite (block) Toeplitz matrix.
fstupd.m	updates a factorization of a symmetric positive definite (block) Toeplitz matrix or solves the associated linear system

The associated test functions and a Matlab 5.3 demonstration package are available via the NICONET homepage. A final report is being prepared to describe the functionality of these routines. An updated version of the routines and interface is expected later on this year.

Daniel Kressner, Vasile Sima and Paul Van Dooren

3 SLICOT tools for model reduction

3.1 SLICOT tools for controller reduction

We are pleased to announce that recently developed model and controller reduction routines have been included in the latest release of the SLICOT library. The development of new routines has been done within Subtask II.B.1 and was based on a preliminary selection of routines (see Working Note SLWN1999-18). The following user callable model and controller reduction routines have been implemented:

Name	Function
AB09HD	stochastic balancing related approaches
AB09ID	frequency-weighted balancing approach
AB09KD	frequency-weighted Hankel-norm approximation method
SB16AD	controller reduction using frequency-weighted balancing methods for three closed-loop performance preserving problems defined by appropriate frequency weights
SB16BD	state-feedback/full-order estimator based controller reduction using coprime factorization in conjunction with balancing related methods
SB16CD	state-feedback/full-order estimator based controller reduction using frequency-weighted coprime factorization in conjunction with the balanced truncation method

Several new model reduction routines have been implemented, which complement and extend the collection of already available general purpose model reduction routines available in SLICOT. Two general purpose routines, **AB09ID** and **AB09KD**, have been implemented to perform frequency-weighted model reduction with general weights (one- or double-sided) using balancing [2, 4] and Hankel-norm approximation based approaches [3, 10]. Both of these routines implement several enhancements proposed recently in [8, 9] of the original approaches. The routine **AB09HD** has been implemented for relative error methods based on the stochastic balancing approach [6, 7]. For controller reduction, three new routines **SB16AD**, **SB16BD**, and **SB16CD**, have been implemented. They address explicitly the controller reduction problems using either the frequency-weighted model reduction framework with special one- or double-sided weights [1, 9], or the coprime factorization based controller reduction [5]. Several supporting routines called by the model and controller reduction routines have also been implemented and standardized: **AB09HX**, **AB09HY**, **AB09IX**, **AB09IY**, **AB09KX**, and **SB16CY**. In particular, **AB09HY**, **AB09IY**, and **SB16CY** are used for the computation of Cholesky factors of the problem dependent observability and controllability grammians.

Seven user callable basic routines and 10 auxiliary routines have been implemented for the special needs of controller reduction software to perform system inversion, to solve discrete Sylvester equations, or to compute special residuals for Lyapunov/Stein equations. These new basic routines enter in SLICOT as contributions to Task I.A (extended over the whole period of the project). Overall, 30 new routines have been implemented within subtask II.B.1.

In the next period, we will implement within Subtask II.B.2 user friendly interfaces to the newly developed model and controller reduction routines for the CACSD environments Matlab and Scilab.

References

[1] B. D. O. Anderson and Y. Liu. Controller reduction: concepts and approaches. *IEEE Trans. Autom. Control*, 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] C.-A. Lin and T.-Y. Chiu. Model reduction via frequency weighted balanced realization. *CONTROL - Theory and Advanced Technology*, 8:341–351, 1992.

[5] Y. Liu, B. D. O. Anderson, and U. L. Ly. Coprime factorization controller reduction with Bezout identity induced frequency weighting. *Automatica*, 26:233–249, 1990.

[6] M. Green and B.D.O. Anderson. Generalized balanced stochastic-truncation. In *Proc. 29th CDC*, pages 476–481, 1990.

[7] 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.

[8] A. Varga. Numerical approach for the frequency-weighted Hankel-norm approximation. *(submitted ECC'2001)*, 2001.

[9] A. Varga and B. D. O. Anderson. Square-root balancing-free methods for the frequency-weighted balancing related model reduction. *(submitted CDC'2001)*, 2001.

[10] K. Zhou. Frequency-weighted L_∞ norm and optimal Hankel norm model reduction. *IEEE Trans. Autom. Control*, 40:1687–1699, 1995.

Andras Varga

3.2 SLICOT tools for model reduction of high order systems

The standardization of the new routines is performed according to the *PSLICOT standard* (SLICOT Working Note 1998-1).

We pursue two approaches for the parallel implementation of absolute error methods for stable systems:

- *Direct solvers* based on computing Cholesky factors of the system Grammians with Hammarling's method. In this approach, the numerical kernels for the model reduction are being ported to a parallel environment. The approach based on the Hammarling's method requires three parallel kernels:
  - *Reduction to Schur form.* This kernel is not totally available in ScaLAPACK, since a post-processing of the diagonal elements is required and computation of transformation matrices from reflectors is not provided. The code has been developed and it is under the optimisation.
  - *Solver for Standard Lyapunov Equation.* This kernel has been totally implemented.

  - *SVD factorisation of a product of triangular matrices.* This kernel is under development.

  These three kernels will be totally available at the end of March 2001.

- *Iterative solvers* based on computing full-rank factors of the system Grammians using the sign function method for solving Lyapunov equation (continuous-time problems) or the squared Smith iteration for solving Stein (discrete Lyapunov) equations (discrete-time case). This approach has the advantage that for large-scale problems, the Grammians are of low (numerical) rank. This yields thin rectangular factors of the computed Grammians. Therefore, the subsequent computations for computing the reduced order model are often much cheaper than for direct methods.

  The following model reduction routines have been implemented. The functionality of the routines duplicates those of the SLICOT routines AB09AD, AB09BD, AB09CD, AB09DD.

  PAB09AX computes reduced (or minimal) order models using either the square-root (SR) or the balancing-free square-root (BFSR) balanced truncation method;

  PAB09BX computes reduced order models using either the BFSR or SR singular perturbation approximation method;

  PAB09CX computes reduced order models using the optimal Hankel norm approximation method based on a minimal realization of the system computed with the SR version of `PAB09AX`;

  PAB09DX applies the singular perturbation approximation formulae to a given system.

  These routines call several other subroutines needed to compute the factors of the Grammians, special singular value decompositions, etc. These subroutines are either provided by ScaLAPACK[1] or the Parallel Library in Control (PLiC) described in the NICONET Report 1999–1[2]. The PLiC subroutines will also be available as part of PSLICOT.

Vicente Hernandez and Peter Benner

[1] See `http://www.netlib.org/scalapack/index.html`.

[2] See `http://www.win.tue.nl/niconet/NIC2/reports.html`.

4 SLICOT tools for subspace identification

4.1 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.

Name	Functionality
IB01AD	possibly sequential triangular compression automatic order detection via SVD (driver)
IB01BD	estimate $\begin{bmatrix} A & B \\ C & D \end{bmatrix}$ and $\begin{bmatrix} Q & S \\ S^T & R \end{bmatrix}$ MOESP and N4SID (driver)
IB01CD	Estimate the initial state and/or $\begin{bmatrix} B \\ D \end{bmatrix}$ using input-output data (driver)
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 $\begin{bmatrix} B \\ D \end{bmatrix}$ using Kronecker products (preferred)
IB01PY	Calculation of $\begin{bmatrix} B \\ D \end{bmatrix}$ exploiting the structure (not recommended for N4SID method)
IB01QD	Estimate initial state and/or $\begin{bmatrix} B \\ D \end{bmatrix}$ using input-output data
IB01RD	Estimate initial state using input-output data

The documentation of the use of the developed routines, and their integration into MATLAB and Scilab via mex files is documented in the SLICOT Working Note 2000-4. A demonstration package for the linear state space model identification toolbox has been developed.

4.2 Standard software for nonlinear state space model identification

The development of identification tools for nonlinear dynamical systems has been started. An inventory of existing algorithms for the identification of MIMO nonlinear Wiener systems has been made. Nonlinear Wiener systems are a concatenation of a linear dynamic block followed by a static nonlinearity. Continuing on the state space framework selected in Task

III.A, the inventory focused on schemes that enable the linear part to be represented in state space form.

The inventory lead to the selection of a combination of a variant of the subspace identification method and a single layer neural network to model the static nonlinearity. A SLICOT standardization of the selected algorithms has been started.

Michel Verhaegen

5 SLICOT tools for robust control

In the last six months, the Topic IV group has been concentrating on the further development of subroutines for the $\mathcal{H}_\infty$ Loop Shaping Design Procedure (LSDP) controller synthesis in the discrete-time case and for the computation of structured singular values (μ). Also, improvements on the $\mathcal{H}_\infty$ norm computation routine developed previously has been made. The following table summarizes the user-callable routines developed concerning the above sub-tasks.

Name	**Function**
SB10KD	Subroutine for discrete-time $\mathcal{H}_\infty$ LSDP controller synthesis, with newly developed formulae
AB13MD	Computation of an upper bound of the structured singular value μ

The following gateway function routines have also been developed for the integration into MATLAB.

Name	**Function**
DLSDP	Gateway function for the discrete-time $\mathcal{H}_\infty$ LSDP controller synthesis
MUCOMP	Gateway function for computing μ

An existing routine `AB13CD` has been modified to improve the computation of $\mathcal{H}_\infty$ norm of a state-space model system.

Name	**Function**
AB13CD	A further modification for the $\mathcal{H}_\infty$ norm computation

The above subroutines have all been tested with numerical examples.

Another major activity during the period was the benchmark designs and industrial case studies using the SLICOT routines. Those include

- an IFAC '93 benchmark problem;
- a mass-spring-damper system (ACC 92 benchmark);
- a two-cart robust control system design;
- a triple inverted pendulum system robust design;
- a glass tube production process control system design (Philips Lighting).

Successful results have been achieved. They can be well served as worked examples in short courses on transferring the advanced technology and SLICOT package to the industry in the following year's schedule.

In addition, the following two technical reports were produced during the period.

1. Da-Wei Gu, Petko Petkov and Mihail Konstantinov
   On Discrete $\mathcal{H}_\infty$ Loop Shaping Design Procedure Routines,
   SLICOT Working Note SLWN2000-6: December 2000.

2. Petko Petkov, Da-Wei Gu and Mihail Konstantinov
μ-synthesis of a robust control system for a triple inverted pendulum,
SLICOT Working Note SLWN2001-1: January 2001.

Da-Wei Gu

6 SLICOT tools for nonlinear systems in robotics

6.1 Nonlinear Systems Simulation

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 has been to implement a standard interface to the most widely used integrator packages (ODEPACK, DASSL, DASPK, RADAU5, DGELDA).

SLICOT can deal now 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 in previous reports, an interface has been implemented 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[3] 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.

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

- PUMA model developed at the Universidad Politécnica de Valencia. It is a manipulator 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[4]. A model of a PUMA560 robot manipulator with 6 degrees of freedom has been obtained from this public toolbox.

- A three-link window cleaning robot provided by Chemnitz University, obtained from: Iou 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. This problem is directly linked with task V.B, as described below.

[3] See `http://www.cwi.nl/cwi/projects/IVPtestset/`

[4] See `http://www.brb.dmt.csiro.au/dmt/programs/autom/pic/matlab.html`

A deliverable for Task V.A[5] has been prepared with extensive information on the work developed. Finally, a Web-based deliverable was made available at URL: `http://www.win.tue.nl/niconet/NIC2/NICtask5A.html`. This Web-page contains a description of the task, links to the different reports and source code, MATLAB Gateway, benchmarks and industrial problems, as well as execution plots.

6.2 Nonlinear Systems Control

The work has been concentrated on the implementation of routines for the computation of the coefficients of a general robot manipulator, the computation of control actions for trajectory tracking and the implementation of templates for a quick use of the integrator packages implemented in Task V.A.

The result has been the following driver routines:

Routine	Purpose
ROBFOR	This routine computes the inertial, coriolis, centripetal and gravitational forces matrices, needed for the computation of the coefficients of the robot dynamic equation.
ROBCEF	This routine computes the coefficients for the dynamic equation using the force matrices.
ROBOCT	This routine computes the input control action for trajectory tracking using optimal control.
ODEDER	Template routine for the integration of the robot model with any of the Ordinary Differential Equation (ODE) Packages interfaced on Task V.A.
DAEDEC DAEDAC DAEDFC	Template routines for the integration of the robot model with the type 3 (See Task V.A deliverable) Differential Algebraic Equations (DAE) Package (GELDA) using the interface implemented on task V.A.
DAEDAB DAEDFB	Template routines for the integration of the robot model with the type 2 (See Task V.A deliverable) DAE Package (ODEPACK or RADAU5) using the interface implemented on task V.A.
DAEDFA	Template routine for the integration of the robot model with the type 1 (See Task V.A deliverable) DAE Package (DASSL or DASPK) using the interface implemented on task V.A.
ODEOUT	Template routine for the computation of the output of the system using the ODE integrator package.
DAEOUT	Template routine for the computation of the output of the system using the DAE integrator package.

The optimal control approach selected requires certain properties on the weight matrices that reduce the problem, avoiding to compute a Riccati Differential Equation, and requiring to solve a Lyapunov Equation instead.

Vicente Hernandez and Ignacio Blanquer-Espert

[5] *Nonlinear Control Systems Simulation Toolbox in SLICOT*, SLICOT Working Note 2000-5. This report is available by anonymous ftp from `wgs.esat.kuleuven.ac.be` in the directory `pub/WGS/REPORTS/SLWN2000-5.ps.Z`.

7 A Web Computing Environment for the SLICOT Library

Abstract
A prototype web computing environment for computations related to the design and analysis of control systems using the SLICOT software library is presented. The web interface can be accessed from a standard world wide web browser with no need for additional software installations on the local machine. The environment provides user-friendly access to SLICOT routines where run-time options are specified by mouse clicks on appropriate buttons. Input data can be entered directly into the web interface by the user or uploaded from a local computer in a standard text format or in MATLAB binary format. Output data is presented in the web browser window and possible to download in a number of different formats, including MATLAB binary. The environment is ideal for testing the SLICOT software before performing a software installation or for performing a limited number of computations. It is also highly recommended for education as it is easy to use, and basically self-explanatory, with the users' guide integrated in the user interface.

7.1 Introduction

Today, many computing services are made available via the internet and the world wide web (www). Slowly, this is changing peoples way to think about computing and their expectations on computing services. For example, many people already find it quite natural to use their computers to pay bills, service bank accounts, or to receive professional brokerage service, instead of visiting their bank's local branch. Many banks provide these services through a standard www interface with no requirements for installing any specific software on the local computer. This greatly reduces the work needed the first time a person tries to use this service. It also makes it possible for one person to use the same service from virtually any computer the person may have access to. Ideally, all instructions needed are also included in the web interface.

In the world of scientific computing, the scene is quite different. A user that is familiar with the problem to be solved and knows where to find a software to solve it may still have to go through a whole series of tasks before she or he actually can try to solve the problem. Normally, the process starts by acquiring and installing the software. Then, there are manuals to read and programs to write. After compiling and linking, the actual computations can take place and an analysis of the results may be performed. In a worst case scenario, one would then find that the software was not appropriate for the problem, and one may start looking for another software and repeat the same procedure.

So, does it have to be like this? Of course not! In order to demonstrate the power of an easy-to-use interface we have developed a prototype for a web computing environment for the SLICOT library [2, 6]. SLICOT is a software library with numerical algorithms for computations in systems and control theory. Based on numerical linear algebra routines from high-performance BLAS [3, 4] and LAPACK [1] libraries, SLICOT provides methods for the design and analysis of control systems.

The use of the web computing service does not require any installation of software on the local computer, nor does it require any documentation besides the one that is integrated in the user interface. The user prerequisites are mainly to know what type of problems, i.e., what equations, that are to be solved. The software requirements are limited to a standard web browser.

This contribution provides a description of this environment that currently includes a small subset of SLICOT. The outline is as follows. Section 2 gives a brief description of the web computing environment, mainly from a user's point of view. This includes presentation of the major functionalities and a step-by-step illustration of a computing session. In Section 3, we give some details about the internal design and implementation. Conclusions and future work are presented in Section 4, mainly to illustrate that this environment is one piece of a larger concept with the common goal to give easy access to state-of-the-art software and different computing resources.

7.2 The SLICOT Web Computing Environment

The SLICOT web computing environment can be accessed from a standard www browser, with no need for additional software installations on the local computer. The main web page, presented in Figure 1, introduces the user to the computing services provided, which presently include a number of SLICOT routines and set of benchmark problems.

In the following, we illustrate how the software can be used to solve a sample problem, namely to compute the solution of continuous-time algebraic Riccati equations

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

The matrices defining the problem can be entered by the user in specific fields in the web interface. They can also be uploaded from MATLAB binary or Fortran data files.

The underlying routine `SB02MD` is an implementation of a Schur vectors method for solving algebraic Riccati equations [5, 8, 7]. The web interface provides full flexibility in specifying data and computational options. Figures 2 and 3 show the user interface for solving algebraic Riccati equations. Here, the matrices have already been entered and the user may select a number of different options, e.g., the type of equation to solve, preferred eigenvalue ordering, and scaling strategy. Of course, the available options depend on the type of problem to solve.

After pressing the "Compute" button in the window of Figure 3, the output is presented as in Figure 4. The solution is presented on the screen but also possible to download, e.g., as plain text files or as MATLAB binaries. By pressing the "Error Bound" button, an error bound of the computed solution is calculated.

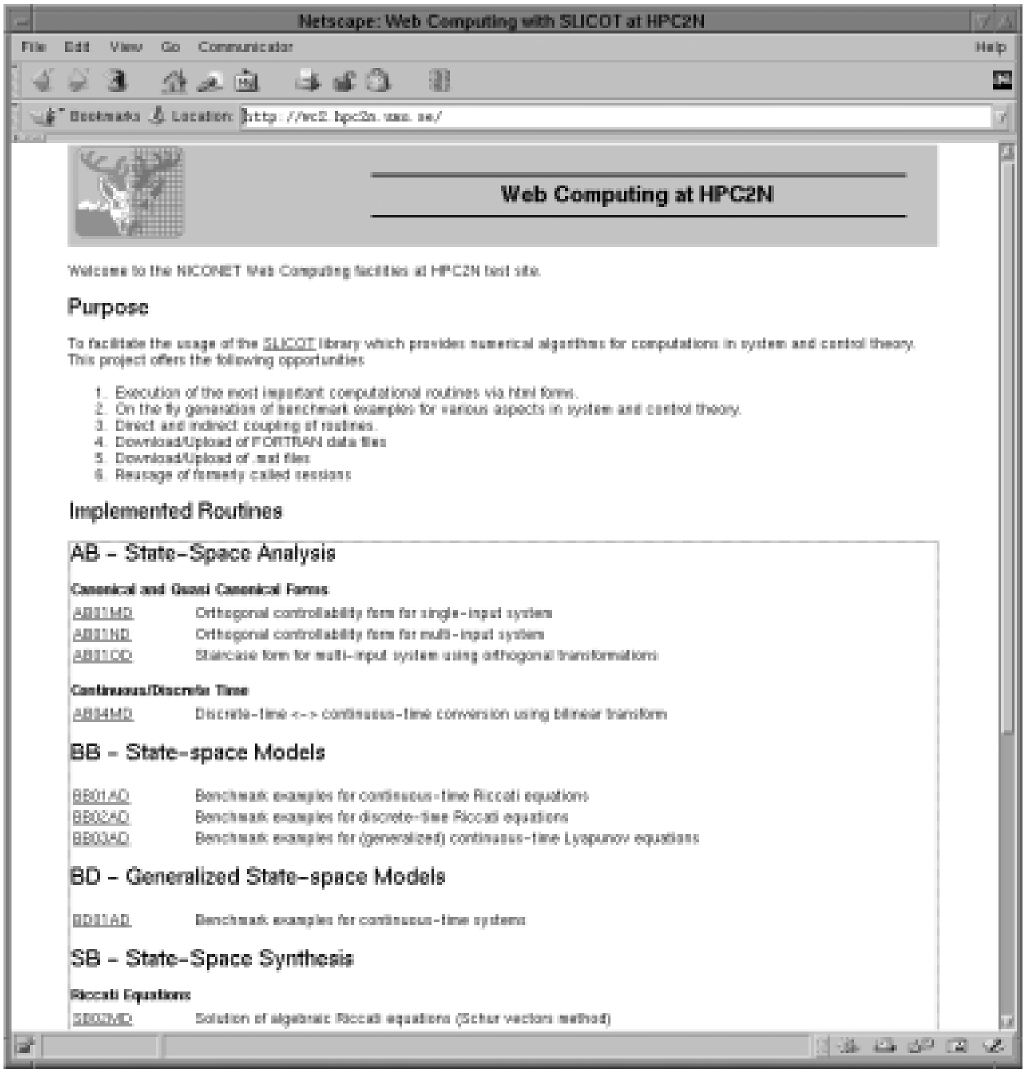
During the web computing session, matrices are stored in a workspace, making the result from the algebraic Riccati equation solver available as input in subsequent computations. The workspace manager serves as a container for all input and output matrices. A convenient interface enables administrating this workspace, as illustrated in Figure 5.

Additional features provided by the workspace manager include possibilities to download matrices in Fortran data format, LaTeX format, MATLAB and Scilab binary format on the local computer and to save complete sessions. The session data is stored on the remote machine so that it may be recovered at a different client machine.

In order to facilitate the test and evaluation of the SLICOT library through the web interface, the SLICOT benchmark collection is also provided. This collection provides relevant test problems for the different routines.

7.3 Implementation Details

Apart from making the web interface intuitive and easy to use, an important aspect in the design of the web computing environment is to facilitate the expansion of the system. Future



Netscape: Web Computing with SLICOT at HPC2N
File Edit View Go Communicator Help
Bookmarks Location: http://wc2.hpc2n.umu.se/

Web Computing at HPC2N

Welcome to the NICONET Web Computing facilities at HPC2N test site.

Purpose

To facilitate the usage of the SLICOT library which provides numerical algorithms for computations in system and control theory. This project offers the following opportunities

1. Execution of the most important computational routines via html forms.
2. On the fly generation of benchmark examples for various aspects in system and control theory.
3. Direct and indirect coupling of routines.
4. Download/Upload of FORTRAN data files
5. Download/Upload of .mat files
6. Reusage of formerly called sessions

Implemented Routines

AB – State–Space Analysis

Canonical and Quasi Canonical Forms

AB01MD	Orthogonal controllability form for single-input system
AB01ND	Orthogonal controllability form for multi-input system
AB01OD	Staircase form for multi-input system using orthogonal transformations

Continuous/Discrete Time

AB04MD	Discrete-time <--> continuous-time conversion using bilinear transform

BB – State–space Models

BB01AD	Benchmark examples for continuous-time Riccati equations
BB02AD	Benchmark examples for discrete-time Riccati equations
BB03AD	Benchmark examples for (generalized) continuous-time Lyapunov equations

BD – Generalized State–space Models

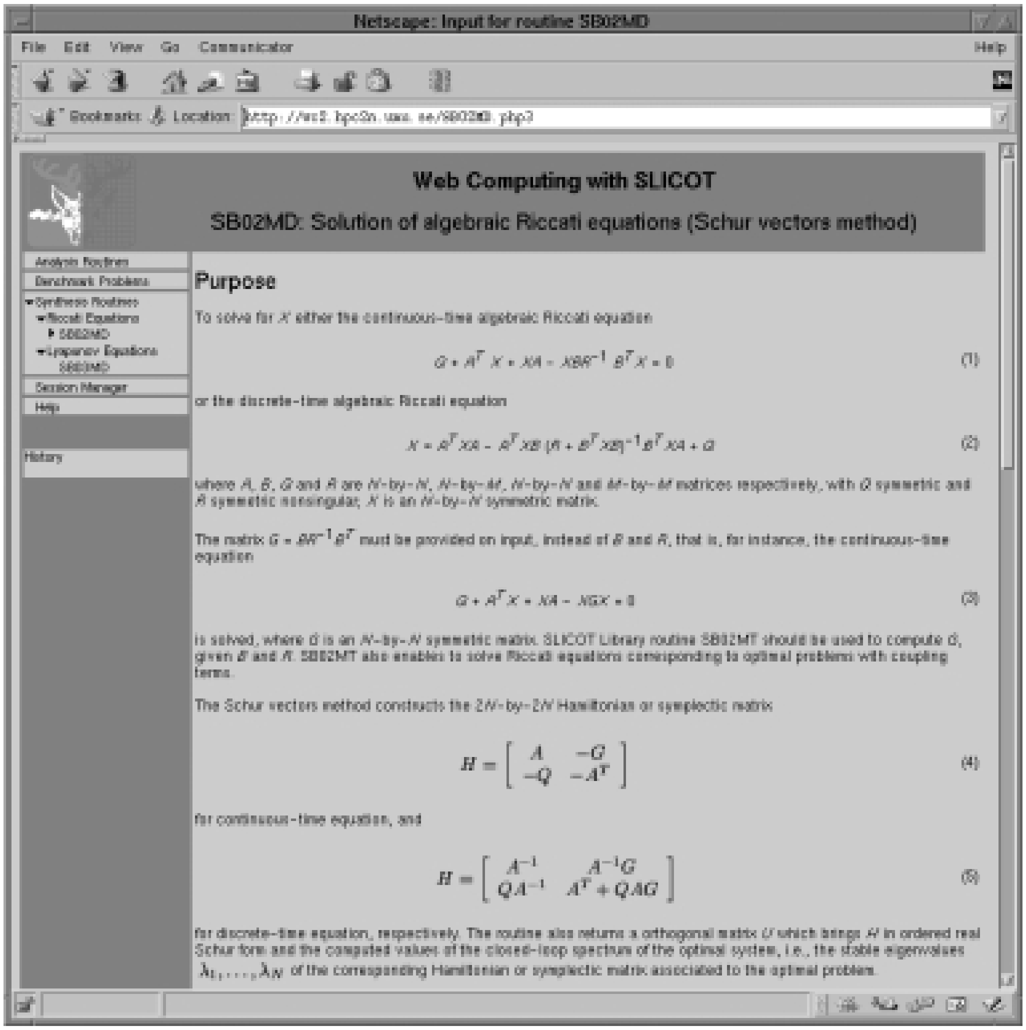
BD01AD	Benchmark examples for continuous-time systems

SB – State–Space Synthesis

Riccati Equations

SB02MD	Solution of algebraic Riccati equations (Schur vectors method)

Figure 1: The main page of the web computing service.



Web Computing with SLICOT

SB02MD: Solution of algebraic Riccati equations (Schur vectors method)

Purpose

To solve for X either the continuous-time algebraic Riccati equation

$$Q + A^T X + XA - XBR^{-1} B^T X = 0 \qquad (1)$$

or the discrete-time algebraic Riccati equation

$$X = A^T XA - A^T XB (R + B^T XB)^{-1} B^T XA + Q \qquad (2)$$

where A, B, Q and R are N-by-N, N-by-M, N-by-N and M-by-M matrices respectively, with Q symmetric and R symmetric nonsingular; X is an N-by-N symmetric matrix.

The matrix $G = BR^{-1}B^T$ must be provided on input, instead of B and R, that is, for instance, the continuous-time equation

$$Q + A^T X + XA - XGX = 0 \qquad (3)$$

is solved, where G is an N-by-N symmetric matrix. SLICOT Library routine SB02MT should be used to compute G, given B and R. SB02MT also enables to solve Riccati equations corresponding to optimal problems with coupling terms.

The Schur vectors method constructs the $2N$-by-$2N$ Hamiltonian or symplectic matrix

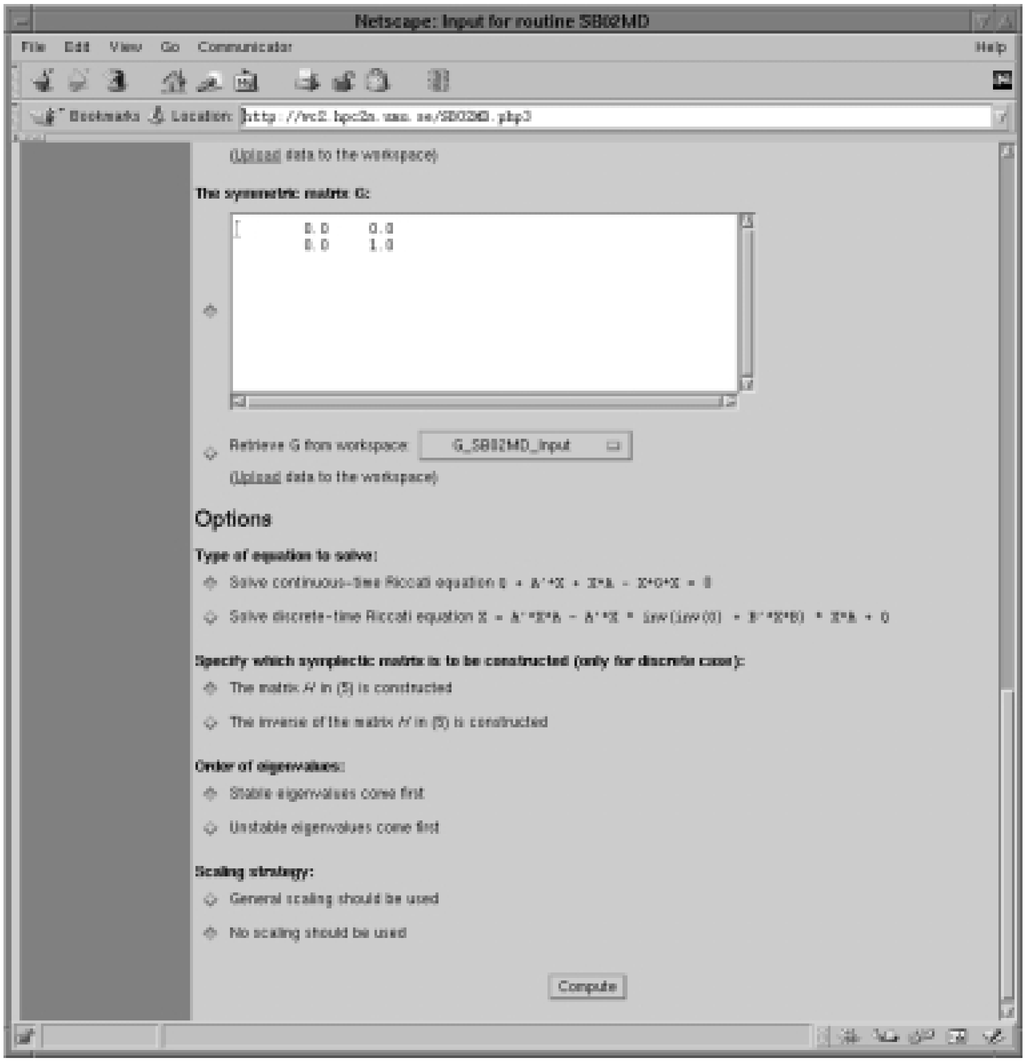
$$H = \left[\begin{array}{cc} A & -G \\ -Q & -A^T \end{array} \right] \qquad (4)$$

for continuous-time equation, and

$$H = \left[\begin{array}{cc} A^{-1} & A^{-1}G \\ QA^{-1} & A^T + QAG \end{array} \right] \qquad (5)$$

for discrete-time equation, respectively. The routine also returns a orthogonal matrix U which brings H in ordered real Schur form and the computed values of the closed-loop spectrum of the optimal system, i.e., the stable eigenvalues $\lambda_1, \ldots, \lambda_N$ of the corresponding Hamiltonian or symplectic matrix associated to the optimal problem.

Figure 2: Introduction to solving algebraic Riccati equations.


Netscape: Input for routine SB02MD
File Edit View Go Communicator Help
Bookmarks Location: http://vc2.hpc2n.umu.se/SB02MD.php3

(Upload data to the workspace)

The symmetric matrix G:

●
```
        0.0     0.0
        0.0     1.0
```

○ Retrieve G from workspace: G_SB02MD_Input

(Upload data to the workspace)

Options

Type of equation to solve:

● Solve continuous-time Riccati equation Q + A'*X + X*A - X*G*X = 0

○ Solve discrete-time Riccati equation X = A'*X*A - A'*X * inv(inv(G) + B'*X*B) * X*A + Q

Specify which symplectic matrix is to be constructed (only for discrete case):

● The matrix H in (5) is constructed

○ The inverse of the matrix H in (5) is constructed

Order of eigenvalues:

● Stable eigenvalues come first

○ Unstable eigenvalues come first

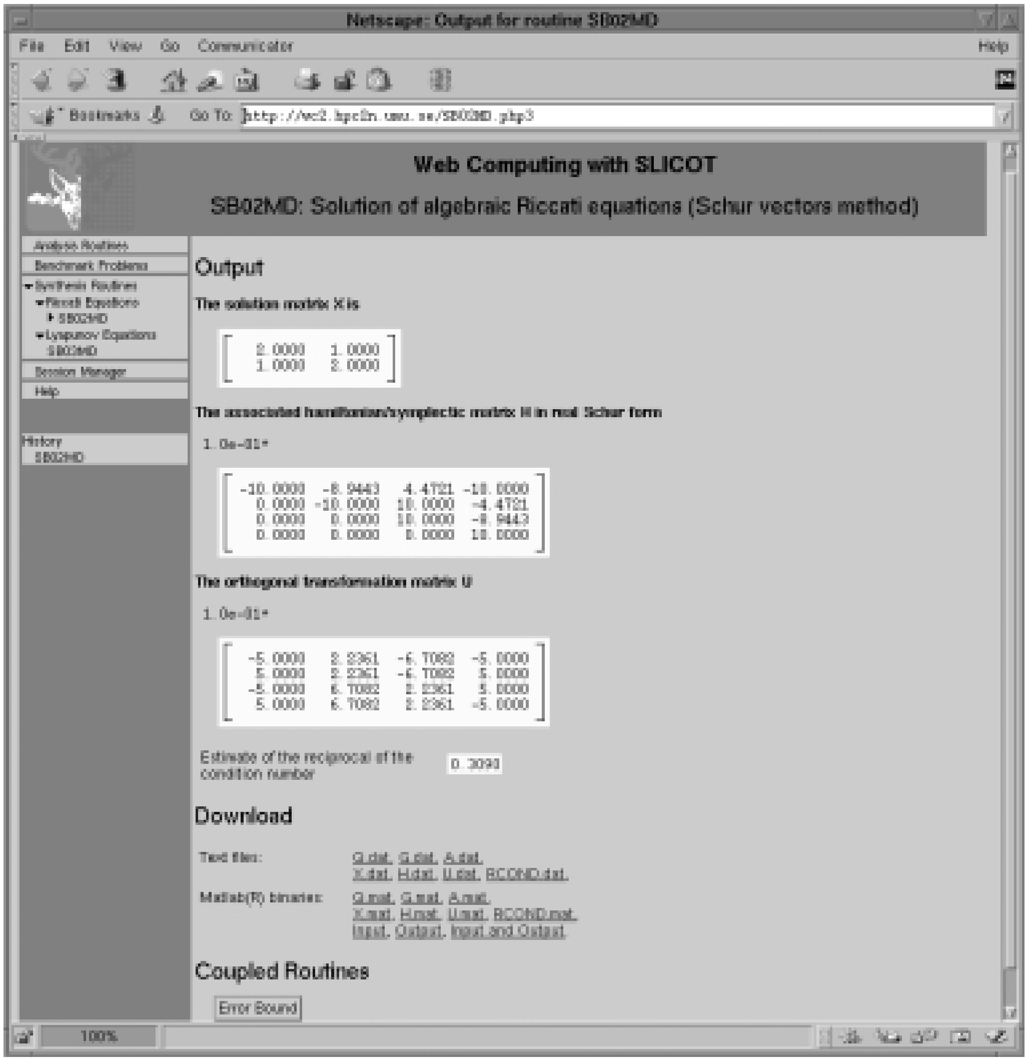
Scaling strategy:

○ General scaling should be used

● No scaling should be used

Compute

Figure 3: Solving algebraic Riccati equations.



Web Computing with SLICOT

SB02MD: Solution of algebraic Riccati equations (Schur vectors method)

Output

The solution matrix X is

```
 2.0000    1.0000
 1.0000    2.0000
```

The associated hamiltonian/symplectic matrix H in real Schur form

```
1.0e-01*

-10.0000   -8.9443    4.4721  -10.0000
  0.0000  -10.0000   10.0000   -4.4721
  0.0000    0.0000   10.0000   -8.9443
  0.0000    0.0000    0.0000   10.0000
```

The orthogonal transformation matrix U

```
1.0e-01*

 -5.0000    2.2361   -6.7082   -5.0000
  5.0000    2.2361   -6.7082    5.0000
 -5.0000    6.7082    2.2361    5.0000
  5.0000    6.7082    2.2361   -5.0000
```

Estimate of the reciprocal of the condition number 0.3090

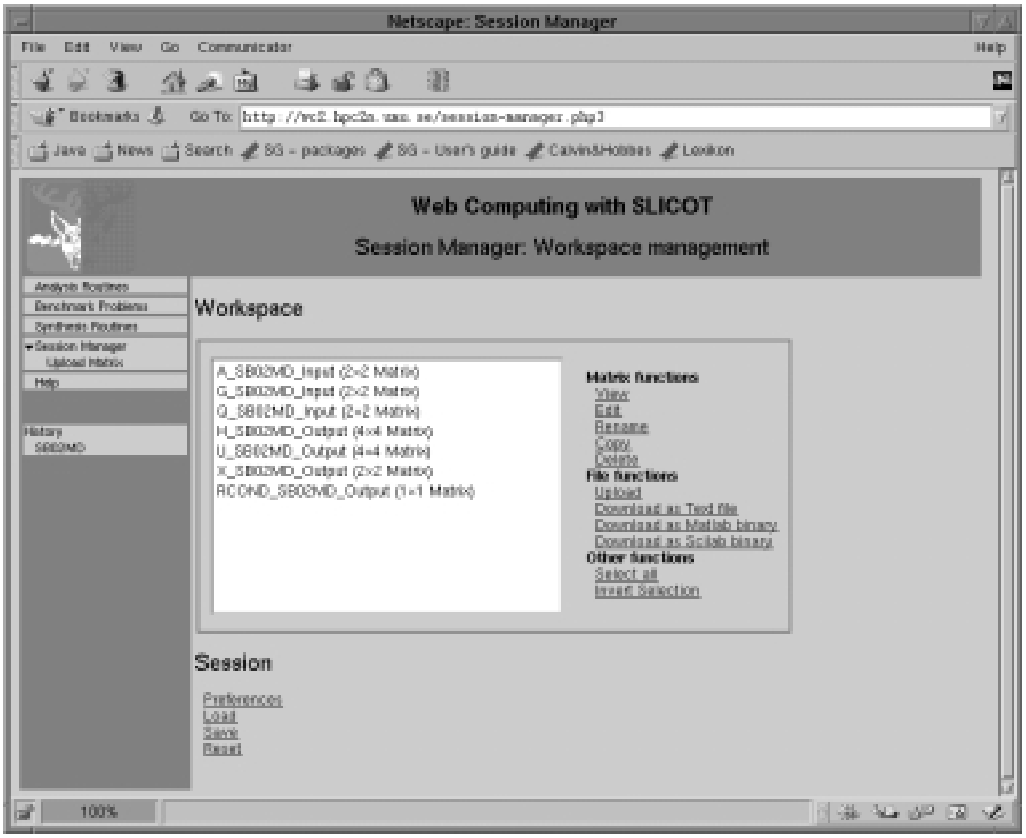
Download

Text files: Q.dat, G.dat, A.dat, X.dat, H.dat, U.dat, RCOND.dat,

Matlab(R) binaries: Q.mat, G.mat, A.mat, X.mat, H.mat, U.mat, RCOND.mat, Input, Output, Input and Output

Coupled Routines

Figure 4: Solution and coupled routines.



Web Computing with SLICOT

Session Manager: Workspace management

Workspace

A_SB02MD_Input (2×2 Matrix)
G_SB02MD_Input (2×2 Matrix)
Q_SB02MD_Input (2×2 Matrix)
H_SB02MD_Output (4×4 Matrix)
U_SB02MD_Output (4×4 Matrix)
X_SB02MD_Output (2×2 Matrix)
RCOND_SB02MD_Output (1×1 Matrix)

Session

Figure 5: Workspace manager.

versions will include a larger number of SLICOT routines, and possibly also other software that may be useful together with SLICOT, e.g., BLAS and other basic computational routines.

Figure 6 describes in a schematic way the overall design of the system.

As the top level programming language we use PHP[6], which is a server-side HTML-embedded scripting language. This means that based on the request of the user and the state of the system, a new web page is created and sent back to the user (PHP-scripts are visualized as a paper with a PHP-logo in Figure 6). In addition to the PHP-scripts, there are also the Fortran-based SLICOT-routines and some C++ based auxiliary routines for conversion and parsing (gear-wheel in Figure 6).

The process when a user first starts to use this interface with a browser is as follows. The input data and the computed results are stored for each individual user. These data are kept in a folder in the file system and maintained in a database. To identify different users, each user is handed a unique ID, stored by the user's browser in something called a cookie. Every time the user goes to one of the pages on the web computing server this ID is sent to the server. If there is no previous information stored about the ID, the server creates a new post in the database and a new folder where data can be stored.

When the user then uploads new data or enter data on the routine web page, the data is stored and accessible as long as the user's browser keeps the given ID. Notice that the identification with the cookie is handled by the interaction between the web server and the

[6] PHP: Hypertext Preprocessor (The first name was Personal Homepage).

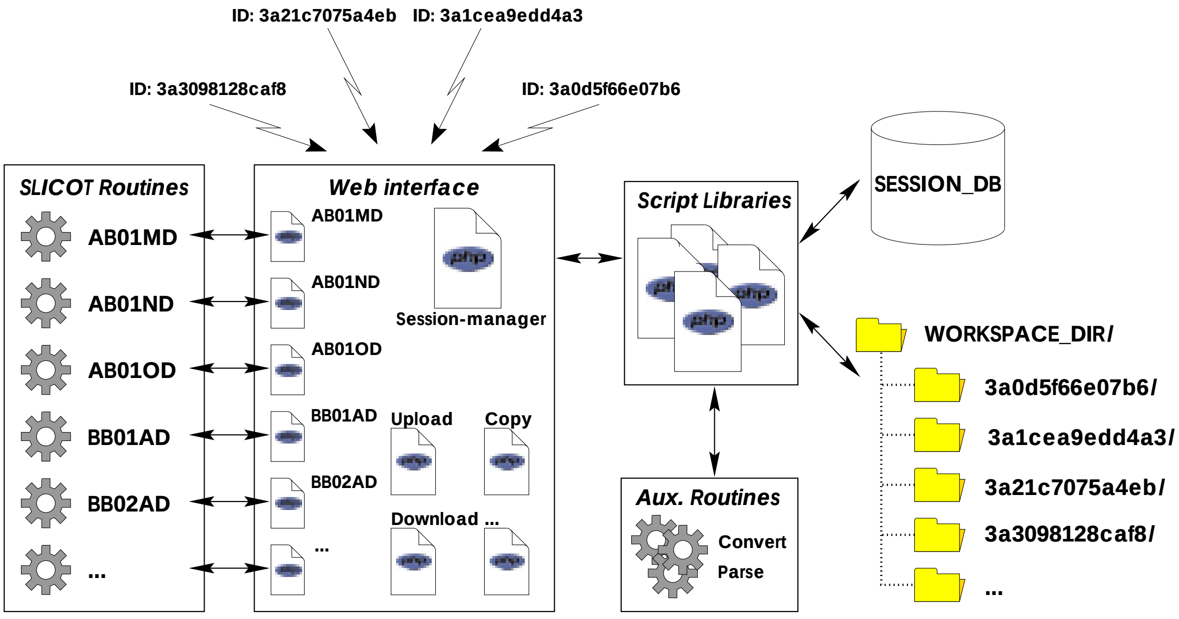


Figure 6: A schematic figure over the web computing system design.

user's browser and not something the user needs to do.

When one of the SLICOT routines is called, the input data is supplied to the SLICOT routine that will be executed on the server. The routine then computes some output data that is presented to the user and stored on the server for later use or download. All data on the server can also be converted to different formats, e.g., MATLAB binaries. Notice that all routines reside on the server and that the computation is performed there and therefore not dependent on the performance or the operating system of the user's machine.

The code in PHP-scripts for the routines and other pages are kept at a minimum. Instead, all code that handle, for example, the calls to routines, communication, parsing and layout, are gathered in libraries that can be used by the other scripts. This makes it easy to modify and add new routines to the system.

7.4 Conclusions and Future Work

The web computing environment presented provides test and use of SLICOT routines from a standard web browser. The routines can be executed on the remote web server with no need for any software installation on the local computer. Consequently, a user can test the SLICOT routines with very little effort, and he or she can connect to the same user interface from virtually any computer. Input matrices can either be entered directly by the user or uploaded from the local computer. Output matrices are presented on the screen and saved in a user workspace for future use, but they may also be downloaded to the local computer. A users' guide will be integrated with the user interface. Moreover, the prototype will be extended to include all SLICOT routines relevant to this type of environment.

Future directions for this project include the development of a computational grid environment. From a standard program, e.g., written in C or Fortran, the user will be able to execute SLICOT routines on more powerful remote computers. In order to do this, the user will only have to install a small software library for handling of the remote access. This library will be used instead of a local SLICOT library. From the point of view of a user, the compilation, execution and linking will be performed exactly as when running on the local computer. In practice, the SLICOT routines are run on some powerful remote computer instead of the local computer (where the other parts of the program are running). As an extension of this concept, the user will also be able to run the whole program on a computational grid of remote computers, where different parts of the execution may take place on different appropriate computers. Of course, the web interface described in this contribution will be able to direct the computations to this powerful grid.

References

[1] 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.* Society for Industrial and Applied Mathematics, Philadelphia, third edition, 1999.

[2] P. Benner, V. Mehrmann, V. Sima, S. Van Huffel, and A. Varga. SLICOT - A Subroutine Library in Systems and Control Theory. In Biswa N. Datta, editor, *Applied and Computational Control, Signal and Circuits*, volume 1, chapter 10, pages 499–539, 1999.

[3] B. Kågström, P. Ling, and C. Van Loan. GEMM-based level 3 BLAS: High-Performance Model Implementations and Performance Evaluation Benchmark. *ACM Trans. on Math. Software*, 24(3):268–302, 1998.

[4] B. Kågström, P. Ling, and C. Van Loan. GEMM-based level 3 BLAS: Portability and Optimization Issues. *ACM Trans. on Math. Software*, 24(3):303–316, 1998.

[5] A. Laub. A Schur method for solving algebraic Riccati equations. *IEEE Trans. Autom. Contr.*, AC-24:913–921, 1979.

[6] NICONET. Network for Development and Evaluation of Numerically Reliable Software in Control Engineering and its Implementation in Production Technologies. Brite/EuRam III RTD Thematic Network, Contract ERB BRRT-CT-97-5040, Coordinator Sabine Van Huffel, ESAT-SISTA, Katholieke Universiteit Leuven, Kardinaal Mercierlaan 94, B-3001 Leuven, Belgium, April, 1997. http://www.win.tue.nl/niconet/.

[7] V. Sima. *Algorithms for Linear-Quadratic Optimization*, volume 200 of *Pure and Applied Mathematics: A Series of Monographs and Textbooks.* Marcel Dekker, Inc., New York, 1996.

[8] W.M. Wonham. On a Matrix Riccati Equation of Stochastic Control. *SIAM J. Contr.*, 6:681–697, 1968.

Erik Elmroth, Pedher Johansson, Bo Kågström, and Daniel Kreßner

8 Highlights of the third NICONET workshop at Louvain-la-Neuve, Belgium

The third NICONET Workshop was organized in Louvain-la-Neuve, Belgium on January 19, 2001. The programme of the day was as follows :

08h30 - 09h00	Welcome coffee
09h00 - 10h00	Matching prediction error identification and robust control
	by Prof. M. GEVERS, CESAME, UCL
10h00 - 10h20	Niconet software for structured matrices
	by Prof. P. VAN DOOREN, CESAME, UCL
10h20 - 10h50	Coffee break
10h50 - 11h50	Model reduction within control systems design
	by Dr. P. WORTELBOER, Philips Research Eindhoven
11h50 - 12h10	Niconet software for subspace identification
	by Dr. V. SIMA, Res. Inst. for Informatics, Bucharest
12h10 - 12h30	Niconet software for nonlinear systems for robotics
	by Prof. V. HERNANDEZ, Univ. Polit. de Valencia
12h30 - 14h00	Lunch
14h00 - 14h45	The Polynomial Toolbox for Matlab
	by Dr. M. SEBEK, Czech Academy of Sciences and
	by Prof. D. HENRION, LAAS-CNRS, Toulouse
14h45 - 15h30	A Matlab Toolbox for Linear Matrix Inequalities
	by Dr. J. STURM, Tilburg University
15h30 - 16h00	Coffee break
16h00 - 17h30	Poster session
17h30	Closing remarks

In the meeting we managed to combine new theoretical developments, the presentation of three new demos of NICONET, two MATLAB toolboxes that were developed by other research activities, and several posters that focused on new algorithmic and computational aspects of systems and control. We had over 50 people attending the conference and had several very lively discussions on the state of the art of numerical methods for systems and control.

The invited presentations of M. Gevers and P. Wortelboer are recorded in the conference proceedings, which also contain the posters presented at the conference. These were :

PSLICOT routines for model reduction of stable large-scale systems, Peter Benner, Enrique S. Quitana-Orti, Gregorio Quintana-Orti

State-feedback H_∞-type control of stochastic linear systems, Tobias Damm

Using Slicot in the Scilab environment, Francois Delebecque, Serge Steer

A Web computing environment for the SLICOT Library, Erik Elmroth, Pedher Johansson, Bo Kågström and Daniel Kressner

Parallel solution of the standard Lyapunov equation by Hammarling's method, David Guerrero, Vicente Hernandez and José Roman

LAPACK and ScaLAPACK : the underlying numerical linear algebra for SLICOT, Sven Hammarling

Some algorithms used in the polynomial toolbox for MATLAB, Didier Henrion, Martin Hromcik, Michel Sebek

Linear parametric design, Ivan Markovsky, Johan Suykens, Sabine Van Huffel

Robust control of triple inverted pendulum using μ-synthesis, P. Petkov, D.W. Gu, M.M. Konstantinov

LMI Optimization in connection with quadratic and nonlinear functions, Diana Sima

Comparison of subspace-based system identification software tools, Vasile Sima

Computational aspects of low order controller design, Andras Varga

Paul Van Dooren

9 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`

9.1 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/niconet/`

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.

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.

9.2 SLICOT Library updates in the period July 2000—December 2000

There has been one major SLICOT Library update during the period July 2000— December 2000: on September 21. Another, intermediate update took place on October 2000, when some updated and new files have been posted on the ftp site. The next major update will take place very soon, and it will incorporate all the routines developed during 2000, as well as all updated 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.

Two new computational routines have been then made available on the ftp site on September update, performing the following tasks:

- computation of the upper triangular factor in the QR factorization of the concatenated block-Hankel matrices, using a fast QR algorithm.

- construction of a hyperbolic plane rotation.

Also, a new directory called `contrib` (under the subdirectory `MatlabTools`) has been added. It contains externally contributed MATLAB files and mexfiles based on or related to the SLICOT library. Included are a Fortran program, two routines, and other associated files for calling SLICOT `TB03AD` routine from a MATLAB environment, developed by Dr. P.-O. Malaterre and used to compare SLICOT results with the MATLAB Polynomial Toolbox results. The codes are described in the NICONET Newsletter, 5, pp. 13-20, July 2000.

Known bugs in the m-files have been also corrected and several new m-files have been added. Moreover, the mexfile `linmeq` can now also solve discrete-time Sylvester equations, as well as Sylvester equations with `op(A) = 'N'`, `op(B) = 'T'`, or `op(A) = 'T'`, `op(B) = 'N'`. Another new option enables to select between the Bartels-Stewart and Hessenberg-Schur methods.

Several other changes have been performed during October 2000. A bug has been removed from the routine `AB13CD`. Also, the `LYAPACK` collections, developed by Dr. Thilo Penzl, for both Unix and Windows platforms, have been added in the subdirectory `MatlabTools/contrib`. `LYAPACK` is an efficient MATLAB toolbox (based on MATLAB m-functions) for the solution of certain large scale problems in control theory, which are closely related to Lyapunov equations. It can solve Lyapunov and Riccati equations, and do model reduction. `LYAPACK` uses iterative algorithms and is intended for large, sparse problems.

Over 25 new user-callable and computational routines for basic control problems, for model and controller reduction, and for $\mathcal{H}_\infty$ loop shaping have been implemented. They include *Analysis Routines*, *Mathematical Routines*, and *Synthesis Routines*, performing the following main computational tasks:

- computing a reduced order model for an original state-space representation, using the stochastic balancing approach in conjunction with the square-root or the balancing-free square-root Balance & Truncate (B&T) or Singular Perturbation Approximation (SPA) model reduction methods for the α-stable part of the system.

- computing a reduced order model for an original state-space representation, using the stochastic balancing approach in conjunction with the square-root or the balancing-free square-root B&T or SPA model reduction methods: state dynamics matrix in real Schur canonical form.

- computing the Cholesky factors of the controllability and observability Grammians.

- computing a reduced order model for an original state-space representation, using the frequency weighted square-root or balancing-free square-root B&T or SPA model reduction methods, trying to minimize the norm of the frequency-weighted error. For unstable systems, only the α-stable part is reduced. A specialized routine is available for stable systems with the state dynamics matrix in real Schur canonical form.

- computing the Cholesky factors of the controllability and observability Grammians for systems with input or output weighting transfer-function matrices.

- computing a reduced order model for an original state-space representation, using the frequency weighted optimal Hankel-norm approximation method in conjunction with square-root balancing for the α-stable part of the system.

- constructing a state-space representation of the stable projection of a stable, weighted transfer-function matrix, with the state dynamics matrix in real Schur canonical form.

- computing an upper bound on the structured singular value for a given square complex matrix and a given block structure of the uncertainty.

- computing the matrix formula
  $$\bar{R} = \alpha\big(\,\mathrm{op}(A)^T\,\mathrm{op}(T)^T\,\mathrm{op}(T) + \,\mathrm{op}(T)^T\,\mathrm{op}(T)\,\mathrm{op}(A)\big) + \beta R,$$
  or
  $$\bar{R} = \alpha\big(\,\mathrm{op}(A)^T\,\mathrm{op}(T)^T\,\mathrm{op}(T)\,\mathrm{op}(A) - \,\mathrm{op}(T)^T\,\mathrm{op}(T)\big) + \beta R,$$
  where α and β are scalars, R, and $\bar{R}$ are symmetric matrices, T is a triangular matrix, A is a general or Hessenberg matrix, and $\mathrm{op}(M) = M$ or $\mathrm{op}(M) = M^T$.

- computing the matrix product U^TU or LL^T, where U and L are upper and lower triangular matrices, respectively, stored in the corresponding upper or lower triangular part of an array (blocked and unblocked algorithms).

- performing the symmetric rank k operations
  $$C \leftarrow \alpha\,\mathrm{op}(A)\,\mathrm{op}(A)^T + \beta C,$$
  where α and β are scalars, C is an $n \times n$ symmetric matrix, $\mathrm{op}(A)$ is an $n \times k$ matrix, and the matrix A has l nonzero codiagonals, either upper or lower.

- computing the matrix product $H \leftarrow \alpha \, \mathrm{op}(T) H$ or $H \leftarrow \alpha H \, \mathrm{op}(T)$, where α is a scalar, H is an $m \times n$ upper or lower Hessenberg-like matrix (with l nonzero subdiagonals or superdiagonals, respectively), and T is a unit, or non-unit, upper or lower triangular matrix.

- computing the Cholesky factor and the generator and/or the Cholesky factor of the inverse of a symmetric positive definite block Toeplitz matrix, defined by either its first block row, or its first block column.

- bringing the first blocks of a generator in proper form.

- applying the transformations created by the SLICOT Library routine `MB02CX` on other columns or rows of the generator.

- updating the Cholesky factor, the generator, and/or the Cholesky factor of the inverse of a symmetric positive definite block Toeplitz matrix, given the information from a previous factorization and additional blocks of its first block row, or its first block column.

- solving a system of linear equations, $TX = B$ or $XT = B$, with a symmetric positive definite block Toeplitz matrix T, defined either by its first block row or its first block column.

- computing the matrices of the positive feedback controller for the continuous-time shaped plant.

- converting a descriptor state-space system into regular state-space form.

- computing the matrices of the positive feedback controller for the discrete-time shaped plant.

- computing a reduced order controller for an original state-space controller representation, using the frequency weighted square-root or balancing-free square-root B&T or SPA model reduction methods.

- computing the Cholesky factors of the frequency-weighted controllability and observability Grammians corresponding to a frequency-weighted model reduction problem, with the controller state matrix in a block-diagonal real Schur form.

- computing, for a given open-loop model (A, B, C, D), and for a given state feedback gain F and full observer gain G, such that $A + BF$ and $A + GC$ are stable, a reduced order controller model (A_c, B_c, C_c, D_c) using a coprime factorization-based controller reduction approach.

- computing, for a given open-loop model (A, B, C, D), and for a given state feedback gain F and full observer gain G, such that $A + BF$ and $A + GC$ are stable, a reduced order controller model (A_c, B_c, C_c, D_c) using a coprime factorization-based controller reduction approach. For reduction of the coprime factors, a stability enforcing frequency-weighted model reduction is performed using either the square-root or the balancing-free square-root versions of the B&T model reduction method.

- computing, for a given open-loop model $(A, B, C, 0)$, and for a given state feedback gain F and full observer gain G, such that $A + BF$ and $A + GC$ are stable, the Cholesky factors S_u and R_u of a controllability Grammian, $P = S_u S_u^T$, and of an observability Grammian, $Q = R_u^T R_u$, corresponding to a frequency-weighted model reduction of the left or right coprime factors of the state-feedback controller.

In addition, several new mexfiles and Matlab m files have been added in the subdirectories `./SLmex` and `./SLTools`, respectively. Also, two demonstration packages, for the structured matrix decompositions toolbox, and for the linear systems identification toolbox, have been developed.

9.3 New NICONET Reports

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

`http://www.win.tue.nl/niconet/NIC2/reports.html`

or from the WGS ftp site,

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

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

- Ad van den Boom, Ton Backx and Yucai Zhu. *Benchmarks for Identification* (file `Nic1999-19.ps.Z`).

  This report describes the preliminary steps for setting up a benchmark collection for identification. The identification protocol is described, where aspects such as experiment set-up, signal pre-processing, modelling, parametrization, estimation methods and model validation are reviewed briefly. The relation of identification and control is stipulated. An analysis is given of requirements for good benchmarks for identification, and some relevant organisational issues are addressed.

- Vicente Hernandez, Ignacio Blanquer, Enrique Arias and Pedro Ruiz. *Definition and Implementation of a SLICOT Standard Interface and the associated* Matlab *Gateway for the Solution of Nonlinear Control Systems by using ODE and DAE Packages* (file `SLWN2000-3.ps.Z`).

  In this report, an interface system for the execution of several widely-used integrator packages for the solution of Ordinary Differential Equations and Diffferential Algebraic Equations is presented. This package offers a SLICOT-compliant unique interface to the packages ODEPACK (LSODE, LSODA, LSODES, LSODI, LSOIBT), DASSL, RADAU5, DASPK and GELDA. All the parameters have been standardised to allow a quick change from one package to another and to take profit of the different capabilities of the different packages. The interface has also been migrated to Matlab, offering the possibility of defining the system functions as Matlab m-files, using the FORTRAN compiled solver packages instead of the Matlab functions. The source code of the system can be downloaded from the SLICOT repository.

- Vasile Sima. *SLICOT Linear Systems Identification Toolbox* (file `SLWN2000-4.ps.Z`).

  This report summarizes the achievements and deliverables of the Task III.A of the NICONET Project. After a short description of the linear system identification problem and of the available subspace-based techniques to solve it, the numerical algorithms implemented in SLICOT Linear Systems Identification Toolbox—SLIDENT—are surveyed. The associated Fortran routines are then listed, and their functional abilities are outlined. The developed interfaces to MATLAB or Scilab, as well as examples of use are presented. Comparisons with the available MATLAB codes are included, illustrating the efficiency and accuracy of the SLIDENT components.

- Vicente Hernandez, Ignacio Blanquer, Enrique Arias, Victor Garcia, Lourdes Penalver and Pedro Ruiz. *Nonlinear Control Systems Simulation Toolbox in SLICOT* (file `SLWN2000-5.ps.Z`).

  This report presents the SLICOT implementation of the nonlinear control systems toolbox. A common interface to several ODE and DAE libraries is prepared. This interface is the entry point to the SLICOT solvers and enables users to test the advantages of different approaches. In addition, an implementation of a MATLAB gateway to the nonlinear control systems simulation interface is developed, which enables the user to define the problems using MATLAB code, including the definition of the system functions and Jacobians. Also, the performance of the toolbox using benchmark problems, as well as industrial test cases is described.

- D.-W. Gu, P.Hr. Petkov and M.M. Konstantinov. *On Discrete $\mathcal{H}_\infty$ Loop Shaping Design Procedure Routines* (file `SLWN2000-6.ps.Z`).

  This report briefly introduces the $\mathcal{H}_\infty$ loop shaping design procedure (LSDP) in the discrete-time case as well as its implementation in the software package SLICOT. Solution formulae are presented with the exposure of a relationship between the solutions to the three discrete-time, algebraic Riccati equations (DARE) required in the construction of an LSDP controller. These SLICOT routines also produce estimates of the condition numbers of the DARE solutions, which reveal the accuracy and reliability of the computational results. The developed routines are tested in a design example, and are included as appendices.

- P.Hr. Petkov, D.-W. Gu and M.M. Konstantinov. *Robust Control of a Triple Inverted Pendulum using mu-Synthesis* (file `SLWN2001-1.ps.Z`)[7].

  In this paper we apply some of the SLICOT routines in the mu-synthesis of a robust control system for a triple inverted pendulum. We consider the case of a mixed type uncertainty consisting of two complex uncertainties in the actuators, three real uncertainties in the moments of inertia and three real uncertainties in the viscous friction coefficients. Using the D-K iteration, a further fictitious complex uncertainty block is included and a mu-controller is constructed for which the closed-loop control system achieves robust stability and robust performance as requested. The influence of the individual uncertainty on the robust stability is investigated using mu-analysis. In addition, a reduced order controller is found such that the robust stability and robust

[7] This report has also been written in 2000, but made available in 2001.

performance of the closed-loop system are preserved with the much lower order controller. In the design, the structured singular value mu is calculated with the SLICOT routine AB13MD and the model reduction toolbox in SLICOT is used in the model reduction of the mu controller. The computation experience shows that the SLICOT routines perform better than the counterpart routines in Matlab in terms of speed and accuracy.

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

9.4 Forthcoming Conferences

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

- Tenth SIAM Conference on Parallel Processing, Portsmouth, USA, March 12–14, 2001.
- Fifth SIAM Conference on Control and its Applications, San Diego, USA, July 11–14, 2001.
- SIAM Conference on Linear Algebra in Signals, Systems and Control, Boston, USA, August 13–16, 2001.
- IFAC Workshop on Periodic Control Systems, Cernobbio-Como, Italy, August 27–28, 2001.
- European Control Conference (ECC 2001), Seminário de Vilar, Porto, Portugal, September 4–7, 2001.

Vasile Sima

10 Announcement of upcoming SLICOT training course in Bremen, Germany

A first SLICOT tutorial will be held at the University of Bremen, September 27–29, 2001. The title of the training course will be *Advanced Computational Tools for Computer-Aided Control Systems Design*. It will consist of lectures introducing SLICOT-based software to be used either within MATLAB or Scilab. During the training part of the course, the attendees will use this software to solve some practical problems in control systems design. Major topics of the course are basic control software, system identification, model reduction, and robust control design using $\mathcal{H}_\infty$ techniques.

Detailed information, program, and registration forms will be mailed to interested persons and will be available in the next issue of the NICONET Newsletter.

For further information please contact Peter Benner, `benner@math.uni-bremen.de`.

Peter Benner