2

ISSN 1373-6787

January 1999

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)
Kardinaal Mercierlaan 94
3001 Leuven-Heverlee, Belgium

email: ida.tassens@esat.kuleuven.ac.be
phone: + 32 16 32 17 09
fax: + 32 16 32 19 70

1 Editorial

Welcome to the second issue of the NICONET newsletter which informs you about the evolution of the SLICOT library and other NICONET activities related to CACSD software developments. NICONET started on January 1, 1998, with 17 partners of which six are industrial. We gather each six months (previous meetings were held in Leuven in January 1998, in Chemnitz in June 1998, and in Valencia in December 1998) to discuss the development of the SLICOT library and its integration in user-friendly environments such as Scilab, Matlab, and ANDECS. Many readers might not be familiar with Scilab, therefore, we introduce in Section 8 this freely available CACSD software package. To assess the performance, reliability and versatility, of the new routines, we are developing benchmarks and select industrial design examples, which should reveal the numerical robustness and computational efficiency of the SLICOT software. More detailed information on the present status of the SLICOT tools is given in Sections 3 to 7. In addition, much attention is paid to the selection and development of industrially oriented benchmarks. The newest developments in the SLICOT benchmark library are described in Section 2.

The importance of numerics in control software for industry was also the topic of our first NICONET workshop held in Valencia on December 4, 1998. Section 10 describes the highlights of this one-day workshop. To illustrate the industrial relevance of SLICOT, we add a new section to our newsletter, entitled "SLICOT: a useful tool in industry?". This section describes well-chosen industrial examples in which SLICOT is being used and shown to improve the performance. To start with, the use of SLICOT in the development of a GPS system for a farm tractor is described in Section 9.

As announced in our first NICONET newsletter, we also issue quarterly a NICONET E-letter[1], which informs you about the newest SLICOT updates, any feedback of SLICOT users (on performance, improvements, new suggestions). Please, subscribe[1] if you haven't done it so far and if you want to be informed on SLICOT and NICONET on a more regular basis.

Sabine Van Huffel
Chairperson of WGS and Coordinator of NICONET.

[1] To subscribe, send e-mail to majordomo@win.tue.nl with the message "subscribe niconet" in the body. Old E-letters are accessible through World Wide Web (URL http://www.win.tue.nl/wgs/) or via anonymous ftp from `wgs.esat.kuleuven.ac.be/pub/WGS/E-LETTER/E-issue-N.Z.` where N denotes the issue number.

2 New developments in the SLICOT benchmark library

Two new extensive benchmark collections will be added to the library:

1. *CTDSX-a collection of benchmark examples for state-space realizations of continuous-time dynamical systems* , see SLICOT Working Note 1998-9.[2]

2. *DTDSX-a collection of benchmark examples for state-space realizations of discrete-time dynamical systems*, see SLICOT Working Note 1998-10.[3]

Updated versions of the benchmark collections for continuous and discrete-time algebraic Riccati equations have been developed and will be added to the library.

Volker Mehrmann and Thilo Penzl

[2] Report available by anonymous ftp from `wgs.esat.kuleuven.ac.be/pub/WGS/REPORTS/SLWN1998-9.ps.Z`.

[3] Report available by anonymous ftp from `wgs.esat.kuleuven.ac.be/pub/WGS/REPORTS/SLWN1998-10.ps.Z`.

3 Basic Numerical SLICOT Tools for Control

Basic Software Tools for Standard and Generalized State-space Systems and Transfer Matrix Factorizations

Basic mathematical software tools and their availability in a well balanced software library is an effort that requires a lot of investment. But once made available, they are of invaluable help for future users, provided the library is well documented, standardized and tested on a full set of benchmark examples. This is why the effort in the first year went mainly to the further elaboration of basic mathematical routines (this holds for both Sections 3 and 4).

The current library consists of more than 100 fully tested and standardized routines (and many more lower level ones), a collection of benchmark problems (mainly for Riccati equations), and on-line documentation and easily accessible routines which can be down-loaded from an ftp site.

For a concise description of the problems being handled in this section and the appropriate references to the literature we refer to the NICONET Newsletter 1 (section 4). This section delivers basic building blocks for most other activities and is therefore very important. The selection, standardization and documentation of basic routines has essentially been completed as originally planned. Care has been put in selecting numerical methods that are state of the art : the numerical reliability, efficiency and flexibility of the selected routines have indeed been demonstrated in the literature.

In this first year it appeared that it would be useful to keep this activity alive beyond the two years that were originally planned. We e.g. decided to add new basic routines related to the periodic Schur form and improved Riccati solvers with condition estimators, which will have to be standardized and documented. We expect that during the elaboration of the other NICONET Tasks, more such adaptations will have to be made.

The following 21 user-callable routines have been standardized in the first 12 months :

Mathematical Routines

Name	Function
MB03RD	computes the block diagonal form of a square matrix (former MB03PD)
MB03QD	reorders the eigenvalues of a real Schur matrix according to several reordering criteria
MB03SD	computes the eigenvalues of a Hamiltonian matrix in square-reduced form
MB03UD	computes all, or part, of the singular value decomposition of an upper triangular matrix
MB04DD	applies a specified symplectic scaling to a Hamiltonian matrix
MB04ZD	transforms a Hamiltonian matrix to square-reduced form by a symplectic orthogonal similarity transformation

Transformation Routines

Name	Function
TB01ID	performs the scaling of a state-space model
TB01KD	computes the terms G_1 and G_2 of an additive spectral decomposition of a transfer-function matrix G with respect to a specified region of the complex plane
TB01LD	performs an orthogonal similarity transformation to reduce the system state matrix to an ordered real Schur form (former TB01SD)
TB01WD	performs an orthogonal similarity transformation to reduce the system state matrix to the real Schur form (former TB01RD)

Analysis Routines

Name	Function
AB13AD	computes the Hankel norm and the Hankel singular values of the stable projection of a transfer-function matrix
AB13BD	computes the L_2- or l_2-norm of a transfer-function matrix

Synthesis Routines

Name	Function
SB03OD	solves for $X = \mathrm{op}(U)'\mathrm{op}(U)$ either the stable non-negative definite continuous-time Lyapunov equation $\mathrm{op}(A)'X + X\mathrm{op}(A) = -\sigma^2\mathrm{op}(B)'\mathrm{op}(B)$ or the convergent non-negative definite discrete-time Lyapunov equation $\mathrm{op}(A)'X\mathrm{op}(A) - X = -\sigma^2\mathrm{op}(B)'\mathrm{op}(B)$, where $\mathrm{op}(K) = K$ or K'
SG03AD	solves GCLEs and GDLEs
SG03BD	solves non-negative GCLEs and GDLEs

Factorization Routines

Name	Function
SB08CD	computes the state-space representations of the factors of a LCFID of a TFM
SB08DD	computes the state-space representations of the factors of a RCFID of a TFM
SB08ED	computes the state-space representations of the factors of a LCF with prescribed stability degree
SB08FD	computes the state-space representations of the factors of a RCF with prescribed stability degree
SB08GD	computes the state-space representation of the TFM corresponding to a LCF
SB08HD	computes the state-space representation of the TFM corresponding to a RCF

In addition to these user-callable routines, a large number of auxiliary routines have been written, standardized and documented, such as specialized versions of Lyapunov and Sylvester solvers. Although these routines are not "user callable", they are very valuable and can still be called in their own right. For this reason, the same documentation standards were followed as for user-callable routines.

Integration of the Basic Software Tools in User-friendly Environments

The above-mentioned basic software tools are more "accessible" when implemented in a user-friendly environment, so that little technical background is required to use the tools to almost full functionality. We have started to integrate top level routines of SLICOT in Matlab via mex-files. Since such mex-files are rather big, we minimize their number by grouping routines which require similar basic routines into one mex-file with multiple functionality (several m-files will call these mex-files). The integration in SCILAB will be done similarly as for Matlab. These tasks should be completed by June 1999. Further tasks are the selection of benchmarks and industrial design problems.

Paul Van Dooren and Andras Varga

4 SLICOT tools for model reduction

Development of standard software for model reduction

The basic model reduction algorithms implemented and standardized in SLICOT in the first 12 months belong to the class of methods based on or related to balancing techniques and are intended to solve model reduction problems for stable or unstable, continuous- or discrete-time linear systems. For a concise description of the numerical algorithms developed in this section, we refer to the NICONET Newsletter 1 (section 5). All routines come from the RASP-MODRED library of the DLR[4] and the selected methods rely on guaranteed mathematical error bounds and have particular features which recommend them for use in specific applications. The underlying computational algorithms represent the latest developments of various procedures for solving computational problems appearing in the context of model reduction. It is worth mentioning that the new standardized model reduction routines of SLICOT are generally superior to those available in the model reduction tools of commercial packages.

The following 7 user-callable routines have been standardized in the first 12 months :

Name	Function
AB09AD	computes reduced order models of stable systems using the *balance* & *truncate* method with enhanced accuracy
AB09BD	computes reduced order models of stable systems using the *singular perturbation approximation* method with enhanced accuracy
AB09CD	computes reduced order models of stable systems using the optimal *Hankel-norm approximation* method with enhanced accuracy
AB09DD	computes a reduced order model by using the singular perturbation formulas
AB09ED	computes reduced order models for unstable systems using the optimal *Hankel-norm approximation* method in conjunction with additive spectral decomposition
AB09FD	computes reduced order models for unstable systems using the enhanced accuracy *balance* & *truncate* method in conjunction with left/right coprime factorization methods
AB09GD	computes reduced order models for unstable systems using the enhanced accuracy *singular perturbation approximation* method in conjunction with left/right coprime factorization methods

As for section 3, several auxiliary routines have also been written, standardized and documented. The standardization activity is finished. However, for functional completeness, two other user-callable routines will be further implemented for reduction of unstable systems. By mid 1999, all model reduction routines will also be accessible within the user-friendly environment MATLAB. In the next 6 months several benchmark problems for model reduction (including examples with poorly scaled systems) will be defined and included in the standard SLICOT benchmark collection for systems. Further testing of standardized model reduction routines and SLICOT – SCILAB – MATLAB comparisons will be performed.

Andras Varga and Paul Van Dooren

[4] German Aerospace Center (DLR), Oberpfaffenhofen

5 SLICOT tools for subspace identification

Development of standard software for linear, time-invariant state space model identification

As mentioned in NICONET newsletter 1 (section 6), an inventory has been made of existing subspace identification methods that can be applied to identify linear, time-invariant systems operating in open-loop. Three routines which were publicly accessible were collected and subsequently tested on 15 real-life data sets. These data sets correspond to important representative and industrially relevant problems from DAISY[5], a database for the identification of systems. The results were documented in a SLICOT working note 1998-6.[6] These three selected subspace identification routines were further evaluated. The aim of the evaluation study was to *split* the 3 routines into basic building blocks (parts) and subsequently to recombine a number of these blocks to obtain more attractive (from a numerical robustness and/or efficiency point of view) routines. To enable a systematic analysis of this evaluation use was made of the software package MWEB[7]. This software package is based on the WEB package of D.E. Knuth [8] to integrate latex (for documentation) into a programming language.

The result of the evaluation study was a new (re-) combination of basic building blocks of the 3 existing routines that yielded computationally more efficient results on the 15 benchmark data sets included in the DAISY database. This new combination will be the version selected for inclusion in SLICOT. In addition, a fast RQ factorization method, which exploits the structure of the data matrices involved, is being developed in order to speed up the time-consuming RQ factorization and will be included in SLICOT in the future. The results of the selection will be documented in a forthcoming SLICOT working note.

Michel Verhaegen

[5] DAISY is a database for the identification of systems which is publicly available on the website `http://www.esat.kuleuven.ac.be/sista/daisy/`

[6] Report available by anonymous ftp from `wgs.esat.kuleuven.ac.be/pub/WGS/REPORTS/SLWN1998-6.ps.Z`.

[7] M. Potse, "MWEB user manual." Department of Medical Physics, University of Amsterdam, 1997 (Internal report to be obtained from the author by sending e-mail to M.Potse@amc.uva.nl)

[8] D.E. Knuth, "Literate Programming." Comp. Journal 27:2(97-111), 1984.

6 SLICOT tools for robust control

The SLICOT robust control section has been substantially expanded in the last six months. The work was focusing on the implementation and standardization of subroutines for the H_2 and H_∞ controller synthesis in the continuous-time case. The following table summarizes the user-callable routines developed so far concerning the above tasks.

Name	**Function**
SB10HD	top-level routine for H_2 optimal controller
SB10UD	H_2 normalization transformation routine with rank condition checking
SB10VD	computation of H_2 optimal state feedback and output injection matrices
SB10WD	calculation of H_2 optimal controller state-space matrices
SB10FD	top-level routine for H_∞ sub-optimal controllers
SB10PD	H_∞ normalization transformation routine with rank condition checking
SB10QD	computation of H_∞ state feedback and output injection matrices
SB10RD	calculation of H_∞ controller state-space matrices
SB10LD	calculation of closed-loop system state-space matrices

Another subroutine *DHINFSYN*, which follows the direct formulae described in the NICONET report 1998-7[9], has been programmed but is not yet standardized, and therefore not included in SLICOT.

Together with the subroutines *SB02MT* and *SB02ND*, which are for the linear-quadratic regulator state-feedback control and are available in SLICOT for both continuous and discrete time cases, the above subroutines form a basis for H_2 and H_∞ controller synthesis and design.

Gateway functions, HIFSYNG.F and H2SYNG.F, and mex-files for Sun SPARC 4 and PC have been developed and tested so that the user may use the above-mentioned routines within a MATLAB environment.

During development of the subroutines, the emphasis has been put on the numerical performance of the algorithms. For instance, condition and error estimators have been included into the algebraic Riccati equation solvers used in the computation in order to provide the user more complete and correct information. The subroutines implementing these estimators are summarized in the following table.

Name	**Function**
SB02QD.F	Estimation of the condition number of the continuous-time matrix Riccati equation and estimation of the forward error
SB03QD.F	Estimation of the condition number of the continuous-time matrix Lyapunov equation and estimation of the forward error
SB03SD.F	Estimation of the condition number of the discrete-time matrix Lyapunov equation and estimation of the forward error

[9] Report available by anonymous ftp from `wgs.esat.kuleuven.ac.be/pub/WGS/REPORTS/nic1998-7.ps.Z`.

They can easily be called from MATLAB via the corresponding gateway functions CNDRICG.F, CNDLYPG.F and CNLYPDG.F, developed and tested with MATLAB V.5.2.1.

In addition, two subroutines *SB01DD* and *MB02QD*, which perform the eigenstructure assignment, have been included into SLICOT and are integrated into MATLAB via the gateway function DEIGENG.F.

Da-Wei Gu

7 SLICOT tools for nonlinear systems in robotics

Since the publication of our previous newsletter, the extension of SLICOT to deal with the identification and simulation of nonlinear systems has been the main topic of work. The need of a user-friendly, simplified but powerful and widely-applicable interface has been the main conclusion.

In order to ease the work with standard packages for nonlinear simulation, a common interface to the most widely used packages for nonlinear systems simulation has been proposed and it is being designed. This interface will only depend on the problem formulation rather than on the special features of each package. This will enable the users to compare quickly the performance of the different packages with different kind of problems.

This interface will cover DASSL, ODEPACK, RADAU5, GELDA, DASPK and MEXX packages, thus covering DAE and ODE problem formulations, and will be available in forthcoming versions of the SLICOT library. The standards of this common interface, which is based on a reduced set of DSBLOCK standard, will provide an easy-to-use SLICOT-compatible interface to these nonlinear systems simulation packages (which are not conform the SLICOT standards).

Development of standard software for simulation and identification of nonlinear systems using robot manipulators as benchmark

The identification problem for a class of mechanical systems was defined as a parameter optimization problem. An inventory was made of existing software to solve this class of nonlinear optimization problems. This study highlighted that a number of optimization packages are available, but in most cases not freely accessible.[10]

Ignacio Blanquer-Espert, Vicente Hernandez, Michel Verhaegen

[10] J.J. Moré, S.J. Wright, “Optimisation Software Guide”, Siam, Philadelphia 1993.

8 An introduction to the CACSD software package Scilab

Scilab is a free (see section 8.9), powerful interactive programming environment very similar to MATLAB. It greatly facilitates the task of numerical computation and data analysis. Problems can be expressed just as they are written mathematically. Even if they result of complex computations, solutions are obtained and visualized with a minimal programming effort.

Scilab's main features are:

- a high-level interpreted, programming language,
- a data environment, including data abstraction,
- a large collection of built-in primitive functions
- a collection of Scilab coded toolboxes
- integrated graphics

8.1 The Programming Language

Scilab is fully endowed with a powerful programming language whose syntax is easy to use. When Scilab is executed, a user prompt within a graphical user interface appears. The prompt is Scilab's interpreter waiting for the user to type a command, which is then immediately evaluated. An interpreted programming language for data analysis is much more powerful than one that requires the construction of complete programs followed by compilation before the data begins to be manipulated. An interpreter in conjunction with a data environment provides immediate access to data, which is easier and less time-consuming than having to plan out a series of computations that lead to some final result. With an interpreter, the user can operate on data a command at a time, which leads to better understanding of a potential processing sequence and allows for rapid experimentation with parameters, as shown in the following example. Here, two step responses for a PID-controlled plant are calculated and plotted using transfer functions.

```
-->G=(s-1)/(s^2+5*s+1);    // the plant
-->Kp=-1.2;T0=1;T1=0.1;  K1=Kp*(1+T0/s+T1*s);// a PID regulator
-->t=(0:0.3:100)';      // time discretization
-->plot2d(t,csim('step',t,G/.K1)') //Closed loop step response
-->K2=Kp*(1+T0/s+T1*8*s);// another PID regulator
-->plot2d(t,csim('step',t,G/.K2)',5,'000')
```

8.2 A Rich Collection of Data Types

Scilab works with a rich collection of data types. Scilab is fundamentally matrix based. However, Scilab's matrices can contain many interesting data types. For example, matrices can contain real and complex numbers, variable-length strings, polynomials, rationals, and Boolean data types. Sparse matrices are also supported.

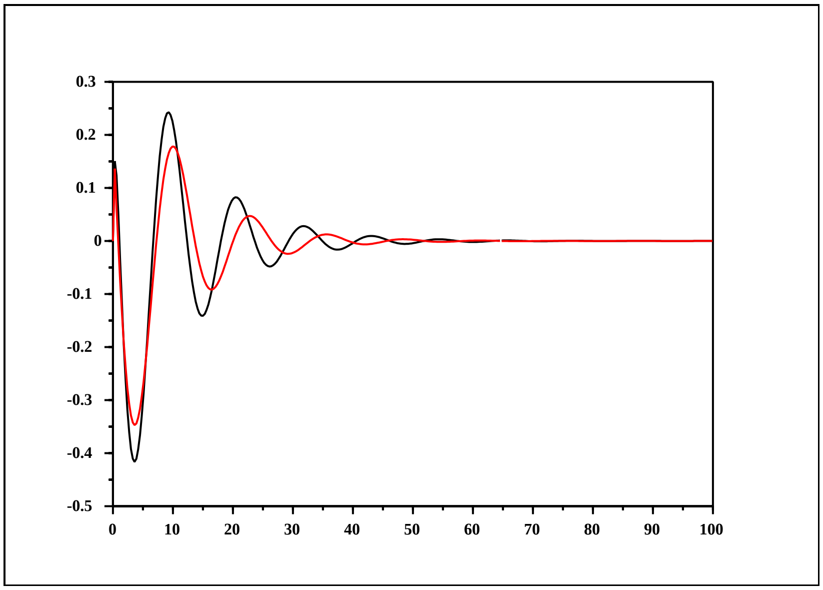

Figure 1: Step response display

```
-->p=poly([1,2],'w')
 p  =
 
                2
   2 - 3w + w
 
-->Q=[p,p^2]
 Q  =
 
!             2                  2    3    4 !
!   2 - 3w + w     4 - 12w + 13w - 6w + w   !
 
-->roots(Q(1))
 ans  =
!    .3333333 !
! - 1.        !
 
-->q=p^2;r=(p+q)/(p*q)
 r  =
                     2
        1 - w + 4w
   --------------------
                2    3    4
  - 2 + 7w - w - 7w + 3w
```

Scilab's most interesting data types are lists and typed lists. These powerful data types are the basis of Scilab's ability to perform data abstraction and to manipulate complex objects. Data abstraction is implemented using typed lists, which allow for the aggregation of heterogeneous data types and the labeling of the new abstract data type. Data abstraction, in conjunction with operator overloading, allows the user to define new data types, which can be combined with each other or with other data types using any of Scilab's built-in operators as shown in the following example:

```
x=tlist(['lss','A','B','C','D','X0','dt'],A,B,C,D,X0,dt)
```

defines `x` as a `lss` (for linear state space) class structure with component names `A` , `B`, `C`, `D`, `X0`, and `dt` (time domain/sampling period).

Furthermore, functions can easily be constructed to handle abstract data types transparently. As an example, the function `%lss_f_lss` below defines the column concatenation `[a;b]` of two variables of type `lss`:

```
function s=%lss_f_lss(s1,s2)
[s1,s2]=sysconv(s1,s2)   //Convert to consistent lss systems
s=tlist(['lss','A','B','C','D','X0','dt'],..
   sysdiag(s1('A'),s2('A')),..
   [s1('B');s2('B')],..
   sysdiag(s1('C'),s2('C')),[s1('D');s2('D')],..
   [s1('X0');s2('X0')],s1('dt'))
```

Scilab is delivered with three built-in abstract data types: rationals, linear systems, and graphs. Furthermore, Scilab users can easily define new abstract data types, implement operator and function overloading for these new types, and develop functions that will perform other typical operations for them.

8.3 Primitive Functions

The most fundamental operations in Scilab, called primitive functions, are implemented in C and FORTRAN. The user can create new primitives by writing C or FORTRAN programs, which are then made accessible to the user in Scilab through a variety of interfacing methods.

8.4 Toolboxes

Since Scilab has its own programming language, it is useful to combine collections of Scilab commands into programs, called functions. These are referred to as Scilab coded functions to differentiate them from primitive functions. The creation of functions adds a great deal of functionality to Scilab. A large collection of useful function toolboxes has already been created, among which there are the following:

- Linear Algebra Toolbox
- ARMA Toolbox
- Classical Control Toolbox
- Polynomial Toolbox
- Hybrid System Simulation Toolbox (Scicos)

- Communication Toolbox
- Parallel Toolbox (using PVM)
- Graphs and Networks Toolbox (Metanet)
- Nonlinear Optimization and Simulation Toolbox
- Robust Control Toolbox
- Signal Processing Toolbox
- Hidden Markov Models Toolbox
- Fuzzy Logic Toolbox
- Fractal Analysis Toolbox

Toolboxes are just collections of functions that have some common application, and consequently, new toolboxes can be created by any Scilab user.

8.5 Scilab and SLICOT

Many Scilab primitives in the control toolbox are based on SLICE and SLICOT routines. In particular, Riccati, Lyapunov, and Sylvester equations solvers, pole placement algorithm, controllability forms, system zeros computation are issued from the SLICOT library. The Scilab group is now working on the integration of a large part of the new SLICOT library developed by the NICONET project. Of course, it is crucial both for reliability and speed to have many built-in primitives coded along the SLICOT standards.

8.6 Integrated Graphics

Finally, integrated graphics are one of Scilab's most useful tools, since they allow the user to examine data interactively. Data understanding is a very important aspect of any scientific or engineering activity, and the capability to rapidly examine large collections of data and present them graphically is very powerful. Just as in the case of Scilab primitives and coded functions, the plotting tools act directly on data in the environment.

Scilab can manage multiple plot windows simultaneously. Furthermore, Scilab plots in 2-D and 3-D, performs contouring and parametric plots, and has a variety of other useful plotting modes (such as Postscript and Xfig).

8.7 Scicos

Scicos (Scilab Connected Object Simulator) is a Scilab package for modeling and simulation of hybrid dynamical systems. Scicos is a simulation environment in which both continuous and discrete systems can co-exist. Unlike many other existing hybrid system simulation software, Scicos has not been constructed by extension of a continuous simulator or of a discrete simulator; Scicos has been developed based on a formalism that considers both aspects from the beginning. Scicos includes a graphical editor for building complex models by interconnecting blocks which represent either predefined basic functions defined in Scicos libraries (palettes), or user defined functions.

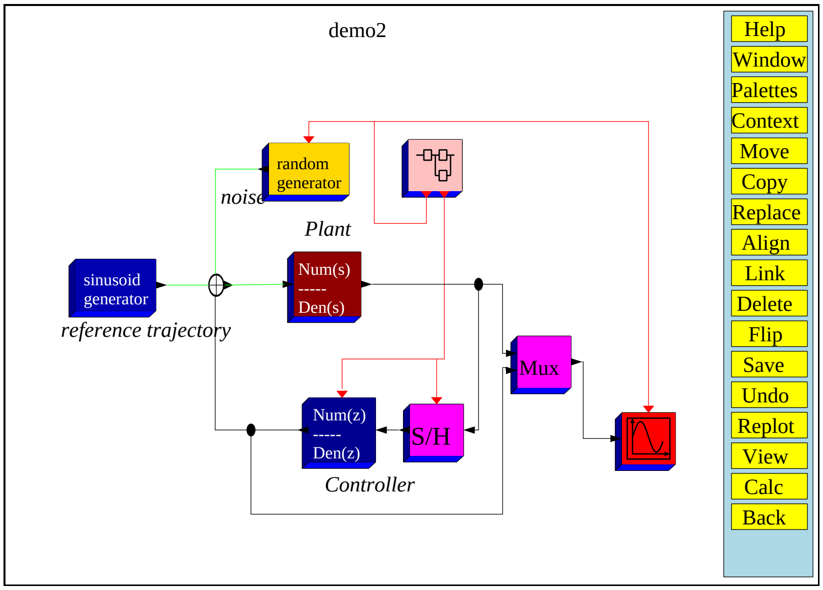

Figure 2: A typical Scicos diagram

8.8 MATLAB Compatibility

The MATLAB and Scilab philosophy and syntax are very similar, so it is rather simple to translate code from MATLAB to Scilab. Scilab distribution includes useful features to improve compatibility:

- a Mex library, which allows to use, without any change, MATLAB mex-files within Scilab,
- a MATLAB4 to Scilab m-file translator,
- function to save and load binary MATLAB4 files.

8.9 Distribution

Scilab is free software, it is distributed with all its source code, and many thousands of copies have already been distributed via the Internet throughout the world. Relevant updates, news, and contributed packages can be obtained from Scilab's home page on the web:

http://www-rocq.inria.fr/scilab/

ftp:ftp.inria.fr:/INRIA/Scilab

Also, there is a newsgroup for discussions on the usage and developments in Scilab. The newsgroup can be found on

comp.soft-sys.math.scilab

Serge Steer and François Delebecque

9 SLICOT: a useful tool in industry?

As mentioned in the editorial, this section is new and is added to our newsletter, from now on. It describes well-chosen industrial examples in which SLICOT is being used and shown to improve the performance. To start with, the use of SLICOT in the development of a GPS system for a farm tractor is presented here:

SLICOT drives tractors!

Motivation

In control engineering applications, the numerical solution of one of the fundamental matrix equations of linear control theory often causes a bottleneck for its successful realization. Here we will describe one such application, the automatic steering of a farm tractor, in which this situation occurs. By using a SLICOT subroutine, the problem is resolved, i.e., the used subroutine yields a numerical solution with sufficient accuracy and in due time.

Through the development of ever more cheap and reliable GPS (Global Positioning System) receivers, automatic steering of ground vehicles along prescribed trajectories has become an attractive research topic for several reasons: hazardous operations can be performed without risking human life, cars may be automatically piloted to the desired destination ("smart highways" – an ongoing research project for several decades), maneuvering at high precision, etc. One of the first fields of application are vehicles used for farming on the large fields of Northern America. As farm vehicles usually move at moderate speed and agricultural fields usually are ideal in order to operate GPS properly as nothing blocks the GPS signals, they are among the first ones for which such a technology becomes feasible for industrial application.

We will describe a project realized by the GPS Lab at the Department of Aeronautics and Astronautics of Stanford University (USA). The project has been supported by Deere and Company and it is hoped that John Deere tractors equipped with the developed technology will soon go into production at Deere and Company.

The research team at Stanford has realized a control mechanism for automatic steering of a farm tractor. Though on first glimpse, this may not seem to be of major interest, there are several reasons why such a system may be desirable. On the large fields in the Midwest of the USA or Canada, tractor driving gets extremely boring. It is reported that drivers even fall asleep, causing significant damage by getting off track and thereby destroying many rows of crops or hoses for watering the fields. Even drivers awake may not be able to drive at necessary precision level to avoid crossing the watering hoses. As other trajectories than "back-and-forth, using U-turns" are difficult for manual drivers, automatic steering would make it possible to steer optimal patterns (e.g., spiral patterns). GPS–based controllers might enable a single driver to operate a convoy of several tractors at the same time or it could allow farmers to operate during the night, through heavy dust or fog. While a tractor driving automatically could be controlled from an office, there are also other reasons to free the farmer from driving. On large fields usually the soil conditions and other factors vary. It is therefore desirable to adapt bedding, seeding, fertilizing, cultivating, and irrigation to these variations and thereby minimizing the use of herbicides and pesticides while maximizing the crop. The manual adaption of these tasks is facilitated if the farm vehicle is steered automatically. Eventually, these adaptions may even be executed automatically as well.

Ground position determination

The approach taken in this project is to first measure the field and record every obstacles to be avoided like, e.g., watering devices, and then to program the tractor to follow a prescribed path. As deviations from the path are unavoidable, a regulator using reference tracking needs to be implemented in order to keep the tractor "on track". Therefore the knowledge of the exact location of the driving tractor is required to regulate the driven trajectory. This has become possible by using GPS.

The ***G**lobal **P**ositioning **S**ystem*[11] is a constellation of 24 satellites circling the Earth every 12 hours. Using standard geodetic methods it is possible to locate your position on earth by receiving the signals (position and time) of at least three (in theory — in practise at least four are necessary) satellites [15]. Since this system has been made available for civilian use it has found many applications from aviation to wilderness expeditions. Hand-held GPS receivers are now even available for outdoor camping. A precision of about 100 yards as attained by normal civilian GPS receivers is of course not sufficient for the application described here. Therefore, a high precision GPS-based system, called *Carrier Phase Differential GPS* (CDGPS), has been used here [7, 8, 10, 14]. CDGPS requires ground based local transmitters, called *pseudolites.* With this technology, it is possible to obtain position and attitude at centimeter-level and 0.1° accuracy. The technology has been made feasible for use in farming by reducing the number of necessary pseudolites for initializing the CDGPS to only one [13].

The research group at Stanford University has equipped the John Deere 7800 farm tractor shown in Figure 3[12] with such a CDGPS receiver. This has then been used for system identification and automatic control of the tractor [3, 7, 8, 11, 12, 14]. Here we will focus on the automatic control aspect. We will describe the model used to design the controller for the tractor in the next section.

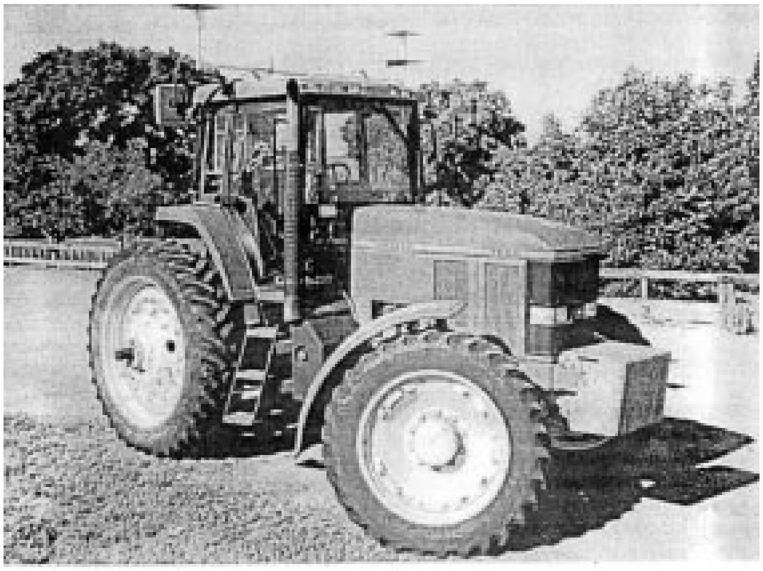

Figure 3: GPS–equipped tractor.

[11] The system is operated by the U.S. Department of Defense.

[12] Taken from [12].

Automatic control of a farm tractor

The state-space model used for simulating the tractor movement is derived from the nonlinear equations of motion; see [3, 10]. The relevant variables are shown in Figure 4[13]. The desired trajectory is part of the prescribed path to be followed by the tractor; d is the distance from the center of the rear axle of the tractor (which is the reference point) to the closest point on the reference trajectory, ψ is the yaw angle of the tractor relative to the desired trajectory, δ is the front wheel angle relative to the centerline of the tractor, and l is the distance from the front wheels to the reference point. Linearization of the equations of motion and removing redundancies yields a continuous, fifth-order linear time-invariant (LTI) system in state-space form,

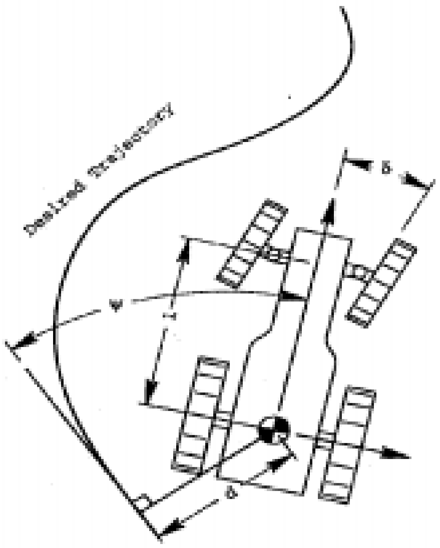

Figure 4: State space variables.

$$
\begin{aligned}
\dot{x} &= Ax + Bu, && (1)\\
y &= Cx, && (2)
\end{aligned}
$$

where the states are $x = [\psi,\ \dot{\psi},\ \delta,\ \dot{\delta},\ d]^T$ and

$$
A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & -\frac{1}{\tau_\psi} & \frac{V}{l\tau_\psi} & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -\frac{1}{\tau_u} & 0 \\ V & 0 & 0 & 0 & 0 \end{bmatrix}, \qquad B = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \frac{1}{\tau_u} \\ 0 \end{bmatrix}, \qquad C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}.
$$

[13] Taken from [3].

Here, V is the forward velocity of the tractor while τ_ψ and τ_u are parameters which were identified using experimental data. The specification of trajectories and generation of reference states is described in detail in [3].

For the closed-loop control of the tractor, a discrete-time linear-quadratic Gaussian (LQG) regulator is used consisting of a linear-quadratic optimal control with reference state tracking [1] and a Kalman filter for estimating the states [4]. The disturbances (sensor noise/process noise) were modeled based on experimental data. The conversion between the continuous plant dynamics and GPS on the one side and the discrete LQG regulator on the other side was performed assuming zero-order hold on the inputs. The block diagram of the closed-loop system is shown in Figure 5[14].

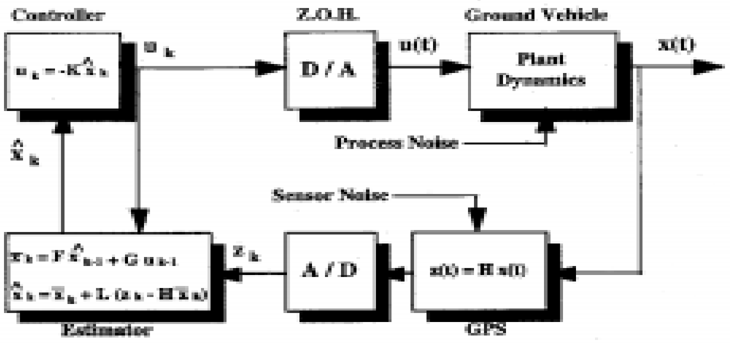

Figure 5: Block diagram of LQG regulator.

The control gains are obtained using the standard Riccati approach, i.e., via the stabilizing solution of the discrete-time algebraic Riccati equation (DARE)

$$X = C^T Q C + A^T X A - A^T X B (R + B^T X B)^{-1} B^T X A, \tag{3}$$

where A, B, C are as in (1)–(2) while Q and R are the weighting matrices for state deviations from nominal and control inputs, respectively, in the quadratic cost functional to be minimized (see, e.g., [1, 4]). For satisfactory regulation results, a DARE of the form (3) has to be solved to high accuracy five times per second. This turns out to be the bottleneck for the realization of an automatic controlled farm tractor as all other necessary computations (matrix adds and multiplies, solution of linear systems, etc.) can be performed sufficiently fast on the available hardware. (The tractor is equipped with a 100MHz Pentium–PC.)

In a first approach the control gains were calculated with MATLAB[15] using gain-scheduling on forward velocity. The computed gains were uploaded to the computer on board of the tractor. The controller then switched between various gains based on the velocity V. Employing

[14] Taken from [14].

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

the SLICOT subroutine **SB02MD**[16] which provides an implementation of the Schur vector method for solving (3) (see [9]), it is now possible to compute the control gains on-line. The DARE is solved in roughly 5% of the 200msec sampling time which gives plenty of time for doing the other necessary calculations. Having the ability to solve the DARE in real time now means that one can use information from an on-line adaptive identification algorithm to actually improve the control going along. Before **SB02MD** was used, a fixed model had to be assumed and changing conditions could not be incorporated into the control law [2].

Concluding remarks

The GPS-based control strategy described in this article can be used for other land vehicles, and, using a slightly more complex CDGPS even for autolanding of planes; see, e.g., [5, 6].

The application described in this article demonstrates how the use of robust numerical software provided in SLICOT can enable control engineers to realize innovative ideas and to make them available for production processes.

References

[1] B.D.O. Anderson and J.B. Moore. *Optimal Control – Linear Quadratic Methods.* Prentice-Hall, Englewood Cliffs, NJ, 1990.

[2] T. Bell. Personal communication, January 1999.

[3] T. Bell, M. O'Conner, V.K. Jones, A. Rekow, G. Elkaim, and B. Parkinson. Realisitic autofarming: Closed-loop tractor control over irregular paths using kinematic GPS. Department of Aeronautics and Astronautics, Stanford University, Stanford, USA, November 1997.

[4] A.E. Bryson and Y.C. Ho. *Applied Optimal Control.* Hemisphere Publ. Co., Washington, 1975.

[5] C.E. Cohen, M. Pervan, D. Lawrence, S. Cobb, D. Powell, and B. Parkinson. Real-time flight testing using integrity beacons for GPS category III precision landing. *Navigation*, 41(2):145–157, 1994.

[6] C.E. Cohen, M. Pervan, D. Lawrence, S. Cobb, D. Powell, and B. Parkinson. Autolanding a 737 using GPS integrity beacons. *Navigation*, 42(3):467–486, 1995.

[7] G. Elkaim, M. O'Conner, T. Bell, and B. Parkinson. System identification of a farm vehicle using carrier-phase differential GPS. In *Proc. ION GPS-96*, pages 485–494, Kansas City, MO, September 1996.

[8] G. Elkaim, M. O'Conner, T. Bell, and B. Parkinson. System identification of a farm vehicle using carrier-phase differential GPS. In *Proc. ION GPS-97*, Kansas City, MO, September 1997.

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

[16] See `ftp://wgs.esat.kuleuven.ac.be/pub/WGS/SLICOT/doc/SB02MD.html` for more details.

[10] M. O'Conner. *Carrir-Phase Differential GPS for Automatic Control of Land Vehicles.* PhD thesis, Stanford University, 1997.

[11] M. O'Conner, T. Bell, G. Elkaim, and B. Parkinson. Kinematic GPS for closed-loop control of farm and construction vehicles. In *Proc. ION GPS-95*, pages 1261–1268, Palm Springs, CA, September 1995.

[12] M. O'Conner, T. Bell, G. Elkaim, and B. Parkinson. Automatic steering of farm vehicles using GPS. In *3rd Intl. Conf. Precision Agriculture*, Minneapolis, MN, June 1996.

[13] M. O'Conner, T. Bell, G. Elkaim, and B. Parkinson. Real-time CDGPS initialization for land vehicles using a single pseudolite. In *Proc. ION National Technical Meeting*, pages 717–724. Institute of Navigation, January 1997.

[14] M. O'Conner, G. Elkaim, and B. Parkinson. Carrier-phase DGPS for closed-loop control of farm and construction vehicles. *NAVIGATION, Journal of the Instititute of Navigation*, 43(2):167–178, 1996.

[15] G. Strang and K. Borre. *Linear Algebra, Geodesy, and GPS.* Wellesley–Cambridge Press, Wellesley, MA, 1997.

Peter Benner and Heike Faßbender

10 Highlights of the first NICONET workshop in Valencia, Spain

The objective of the first NICONET workshop was 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.

The final workshop programme was the following:

```
WORKSHOP PROGRAMME

 9:15 Registration
 9:45 Welcome  and  Introduction
10:00 Plenary Session: Prof. Pedro Albertos
      (Discrete Time Nonlinear Systems Models for Control)
11:00 Coffee Break
11:30 Plenary Session: Prof. Ton Backx
      (Numerical Problems in Industrial Control Systems)
12:30 Plenary Session: Prof. Ad van den Boom / Dr. Vasile Sima
      (Performance and Applicability of the SLICOT Library)

13:30 Lunch

15:30 Afternoon Schedule
15:35 First Poster Session
16:30 Coffee Break
17:00 Second Poster Session
18:00 Demo Session: Dr. Vasile Sima
      (Demonstration on the Use of the SLICOT Library)
18:30 Demo Session: Prof. Petko Petkov / Prof. Mihail Constantinov
      (Demonstration on the Codes for Hinf and H2 Design)
19:00 Panel discussion: moderated by Prof. V. Hernandez and Prof. S. Van Huffel
      (The Importance and Needs of Numerics in Control
       Software in Industry)
19:30 End of the Meeting
```

51 persons attended the workshop. The workshop included 3 plenary sessions on advanced topics in control, a discussion on the needs and problems of the control industry, and an overview of the features of SLICOT, which were also covered in two demonstrations. A poster session was organized to foster the discussion and the exchange of ideas among the attendees, covering several topics such as model reduction, identification, robust control, numerics, parallel processing, robotics and others. The attendance to the meeting was totally free. Attendees received an abstract book with summaries of the plenary sessions and the 27 posters, as well as the last version of the SLICOT software library, documentation and a free subscription to this newsletter.

The workshop ended with a panel discussion in which the industrial partners of NICONET exposed their opinions on the available solutions for control systems. The need for very user-friendly tools, that could be used without having a deep knowledge of the process was the main concern. Interfaces such as MATLAB and Scilab were considered as the minimum level of usability that industry requires.

The next NICONET workshop will be organized in the neighbourhood of Paris, France, on December 2-3, 1999.

Ignacio Blanquer-Espert, Vicente Hernandez, Sabine Van Huffel

11 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. 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`

11.1 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 test program, test data, execution results, the associated `.html` file, as well as the source code for the called SLICOT routines. This involves duplicating some routines, but it can be convenient for a user needing only a single function. There is also a file, called `slicot.tar.gz`, in the directory `/pub/WGS/SLICOT/`, which contains the entire library. The tree structure created after applying

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

and

```
tar xvf slicot.tar
```

is:

```
./slicot/       - for routine source files;
./slicot/doc/   - for html files;
./slicot/tests/ - for test programs/data/results files.
```

The user can then browse through the documentation on his local machine, starting from the index file `libindex.html` from `./slicot` subdirectory.

The library documentation has a new option: browsing through the on-line html documentation for most of the auxiliary routines, through a link from the Library index to the SLICOT Supporting Routines index. Details can be found in the file Release.Notes, in the ftp site, at the address

`wgs.esat.kulueven.ac.be, directory pub/WGS/SLICOT`.

11.2 Latest SLICOT Library update

The latest SLICOT Library update took place in December 1998. The library includes now 239 files for routine codes, 212 html files, and 381 files for example programs, data and results.

Several new user-callable routines for basic control problems have been made available on the ftp site. They include *Analysis Routines* and *Synthesis Routines*, performing the following tasks:

- model reduction for unstable systems using the optimal Hankel-norm approximation method in conjunction with square-root balancing for the α-stable part of the system;
- model reduction for unstable systems using either the square-root or the balancing-free square-root Balance & Truncate (B & T) methods in conjunction with left/right stable coprime factorization techniques;
- model reduction for unstable systems using either the square-root or the balancing-free square-root Singular Perturbation Approximation (SPA) model reduction method in conjunction with stable left/right coprime factorization techniques;
- compute the Hankel-norm of the α-stable projection of a transfer-function matrix for a state-space system;
- compute the H_2 or L_2 norm of the transfer-function matrix of a system without poles on the imaginary axis, for a continuous-time system, or on the unit circle, for a discrete-time system; if the H_2-norm is computed, the system must be stable;
- apply a specified symplectic scaling to a Hamiltonian matrix;
- transform a Hamiltonian matrix to square-reduced form by a symplectic orthogonal similarity transformation;
- compute the eigenvalues of a Hamiltonian matrix in square-reduced form;
- compute the state-space representations of the factors of a left coprime factorization with prescribed stability degree;
- compute the state-space representations of the factors of a right coprime factorization with prescribed stability degree;
- compute the state-space representations of the factors of a left coprime factorization with co-inner denominator of a transfer-function matrix;
- compute the state-space representations of the factors of a right coprime factorization with inner denominator of a transfer-function matrix;
- compute the state-space representation of the transfer-function matrix corresponding to a left coprime factorization;
- compute the state-space representation of the transfer-function matrix corresponding to a right coprime factorization;

- solve for X either the generalized continuous-time Lyapunov equation

  $$\mathrm{op}(A)^T X\mathrm{op}(E) + \mathrm{op}(E)^T X\mathrm{op}(A) = \sigma Y,$$

  or the generalized discrete-time Lyapunov equation

  $$\mathrm{op}(A)^T X\mathrm{op}(A) - \mathrm{op}(E)^T X\mathrm{op}(E) = \sigma Y,$$

  where $\mathrm{op}(M)$ is either M or M^T, A and E are square matrices, the right hand side Y is symmetric, and σ is a scale factor set to avoid overflow in X.

- compute the Cholesky factor U of the matrix X, $X = \mathrm{op}(U)^T\mathrm{op}(U)$, which is the solution of either the generalized c-stable continuous-time Lyapunov equation

  $$\mathrm{op}(A)^T X\mathrm{op}(E) + \mathrm{op}(E)^T X\mathrm{op}(A) = -\sigma^2\mathrm{op}(B)^T\mathrm{op}(B),$$

  or the generalized d-stable discrete-time Lyapunov equation

  $$\mathrm{op}(A)^T X\mathrm{op}(A) - \mathrm{op}(E)^T X\mathrm{op}(E) = -\sigma^2\mathrm{op}(B)^T\mathrm{op}(B),$$

  without first finding X and without the need to form the matrix $\mathrm{op}(B)^T\mathrm{op}(B)$, where $\mathrm{op}(M)$ is either M or M^T, A and E are square matrices, $\mathrm{op}(B)$ is a rectangular matrix, U is an upper triangular matrix with non-negative entries on its main diagonal, and σ is a scale factor set to avoid overflow in U.

Future changes in the library contents or routine updates—till the next SLICOT Release—are announced in the file `Release.Notes`, located in directory `pub/WGS/SLICOT/` on the WGS ftp site. Previous updates are described, in reverse chronological order, in the file Release.History, at the same address.

11.3 New NICONET Reports

Recent NICONET reports (available after August 1998), 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:

- Wouter Favoreel, Vasile Sima, Sabine Van Huffel, Michel Verhaegen and Bart De Moor. *Subspace model identification of linear systems in SLICOT* (file SLWN1998-6.ps.Z).

  This paper compares 3 commonly used subspace identification algorithms N4SID, MOESP and CVA, using their Matlab implementation, in terms of prediction accuracy, simulation accuracy and computational efficiency. The comparison is made on the basis of 15 publicly available practical datasets to which the codes are applied.

- Da-Wei Gu, Petko Hr. Petkov and Mihail M. Konstantinov. *Direct Formulae for the H_∞ Sub-Optimal Central Controller* (file nic1998-7.ps.Z).

  This paper presents alternative formulae (directly based on the original data) of an interconnected system, for the H_∞ suboptimal central controller.

- Petko Hr. Petkov, Da-Wei Gu, and Mihail M. Konstantinov. *Fortran 77 routines for H_∞ and H_2 design of continuous-time linear control systems* (file nic1998-8.ps.Z).

  This paper presents Fortran 77 routines for state-space design of H_∞ (sub)optimal controllers and H_2 optimal controllers for linear continuous-time control systems. The subroutines make use of LAPACK and BLAS libraries and produce estimates of the conditioning of the corresponding matrix algebraic Riccati equations. Modifed formulae are implemented in the case of H_∞ design, which allow to reduce the order of the inverted matrices. The subroutines will be included in the SLICOT library.

- I. Blanquer, D. Guerrero, V. Hernandez, E. Quintana-Orti, and P. Ruiz. *Parallel-SLICOT Implementation and documentation standards* (file SLWN1998-1.ps.Z).

  This paper presents the P-SLICOT (Parallel Subroutine Library in Control and Systems Theory) Implementation and Documentation Standards. Some useful guidelines for the contributors to the parallel version of SLICOT are proposed. The main goal of these rules is to facilitate the work for obtaining a portable, reliable, and easily maintainable code.

- Daniel Kressner, Volker Mehrmann and Thilo Penzl. *CTDSX—A collection of benchmark examples for state-space realizations of continuous-time dynamical systems* (file SLWN1998-9.ps.Z).

  This paper describes a benchmark collection for state-space realizations of time-invariant continuous-time dynamical systems. The collection is intended to provide a means for testing the correctness, accuracy, and speed of numerical methods for several problems arising in control theory. It has been implemented in FORTRAN and MATLAB.

- Daniel Kressner, Volker Mehrmann and Thilo Penzl. *DTDSX—A collection of benchmark examples for state-space realizations of discrete-time dynamical systems* (file SLWN1998-10.ps.Z).

  This paper describes a benchmark collection for state-space realizations of time-invariant discrete-time dynamical systems. The collection is intended to provide a means for testing the correctness, accuracy, and speed of numerical methods for several problems arising in control theory. It has been implemented in FORTRAN and MATLAB.

- Peter Benner, Enrique S. Quintana-Orti, Gregorio Quintana-Orti. *A portable subroutine library for solving linear control problems on distributed memory computers* (file nic1999-1.ps.Z)

  This paper describes the design of a software library for solving the basic computational problems that arise in analysis and synthesis of linear control systems. The library is intended for use in high performance computing environments based on parallel distributed memory architectures. The portability of the library is ensured by using the BLACS, PBLAS, and ScaLAPACK as the basic layer of communication and computational routines. Preliminary numerical results demonstrate the performance of the developed codes on parallel computers. The suggested library can serve as a basic layer for PSLICOT, a parallel extension of the Subroutine Library in Control Theory (SLICOT).

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

11.4 Forthcoming Conferences

Our second two-days NICONET workshop will be held in the direct neighbourhood of Paris, France, on December 2-3, 1999. This workshop is open for everybody. More information follows in our next newsletter.

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/or will disseminate information and promote SLICOT, are the following:

- Computer-Aided Control System Design, CACSD '99, Hawaii, August 22–26, 1999.
- The Fourth International Congress on Industrial and Applied Mathematics, Edinburgh, Scotland, ICIAM99, July 5–9, 1999.
- European Control Conference, Karlsruhe, Germany, 31 August–3 September, 1999.
- Fourteenth Householder Symposium on Numerical Linear Algebra, Chateau Whistler, Whistler B.C., Canada, June 14–18, 1999.

Vasile Sima