# Task I.A - Basic Software Tools for Standard and Generalized State-space Systems and Transfer Matrix Factorizations [1]

Andras Varga[2] and Paul Van Dooren[3]

December 1999

[1]This document presents research results of the European Community BRITE-EURAM III Thematic Networks Programme NICONET (contract number BRRT-CT97-5040) and is distributed by the Working Group on Software WGS. *WGS secretariat*: Mrs. Ida Tassens, ESAT - Katholieke Universiteit Leuven, K. Mercierlaan 94, 3001-Leuven-Heverlee, BELGIUM. This report is also available by anonymous ftp from `wgs.esat.kuleuven.ac.be/pub/WGS/REPORTS/SLWN1999-17.ps.Z`
[2]Deutsches Zentrum für Luft und Raumfahrt, Institut für Robotik und Mechatronik, DLR Oberpfaffenhofen, Postfach 1116, D-82230 Wessling, Germany
[3]Université Catholique de Louvain, CESAME, B-1348 Louvain-la-Neuve, Belgium

## Abstract

This report surveys the deliverables of Task I.A. We first give a brief description of the control problems that are solved by the basic numerical tools developed in this Task and we list the different routines of SLICOT that correspond to these control problems and that are available via ftp. We then describe the toolboxes that give interactive access via Matlab or Scilab to those routines and describe the benchmark problems for this Task. We finally give a few numerical examples exhibiting the accuracy and speed of the new tools and describe a demo for the routines of this Task.

## Contents

# 1 Introduction

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 a lot of effort went to the elaboration of basic mathematical routines. This activity is one that most other activities rely on since they provide them with basic building blocks. The selected routines mainly came from contributors of the WGS group, who now became NICONET partners. 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 the first two years it appeared that it would be useful to keep activity I.A alive beyond the two years that were originally planned, because of the relevance of this activity to all other tasks of our network. Already in the first year we decided to add new basic routines, based on recent developments that should be useful for the other tasks (periodic Schur form, improved Riccati solvers and condition estimators). The current status of the SLICOT library covers a large number of basic mathematical and system theoretic computations. To guarantee a proper distribution of the library, it has been made freely available via ftp.

In this final report, we first survey the basic control problems that are solved by the software delivered in this Task. We then list the different routines that correspond to the problems described earlier. The next section describes the toolboxes that give interactive access via Matlab or Scilab to those routines. In sections 5 and 6 we describe the benchmark problems for this Task and give numerical examples exhibiting the accuracy and speed of the new tools. The last section describes a demo for the routines of this Task.

# 2 Basic Control Problems Solved by Task I.A

## 2.1 Analysis and Synthesis in State-space

Most of the algorithms in this task deal with so-called linear state-space models

$$\begin{array}{rcl} \lambda x(t) & = & Ax(t) + Bu(t) \\ y(t) & = & Cx(t) + Du(t), \end{array} \tag{1}$$

where $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$, $C \in \mathbb{R}^{p\times n}$, $D \in \mathbb{R}^{p\times m}$, and where $\lambda$ is either the differential operator $d/dt$ for a continuous-time system or the advance operator $z$ for a discrete-time system. The system (1) will be alternatively referred to as the quadruple $G = (A, B, C, D)$ or as the triple $G = (A, B, C)$ if $D = 0$ or $D$ is not important for the context. The *transfer function matrix* (TFM) of system (1) is the $p \times m$ proper rational matrix

$$G(\lambda) = C(\lambda I - A)^{-1}B + D. \tag{2}$$

### 2.1.1 Transformations

For an invertible matrix $T$, two state-space systems $(A, B, C, D)$ and $(\widetilde{A}, \widetilde{B}, \widetilde{C}, \widetilde{D})$ related by

$$\widetilde{A} = T^{-1}AT, \quad \widetilde{B} = T^{-1}B, \quad \widetilde{C} = CT, \quad \widetilde{D} = D, \tag{3}$$

are called *similar* and the transformation (3) is called a *similarity transformation*. Similar state-space systems have the same TFM and similarity transformations are the basic preprocessing tools for most of analysis, model conversion and synthesis problems. We give the transformations handled by this task.

1) Given a system $G = (A, B, C)$, we can try to compute a diagonal transformation matrix $T$ in (3) to reduce the 1-norm of the transformed system matrix

$$\widetilde{\mathcal{S}} = \left[\begin{array}{c|c} \widetilde{A} & \widetilde{B} \\ \hline \widetilde{C} & 0 \end{array}\right].$$

Such a transformation is typically used to improve the accuracy of subsequent numerical computations involving the system matrices.

2) A similarity transformation (3) on the system $G = (A, B, C)$, can also be used to put the state transition matrix in a *block-diagonal form* (BDF)

$$\widetilde{A} = T^{-1}AT = \mathrm{diag}(A_1, \ldots, A_k) \tag{4}$$

with the matrices $\widetilde{B}$ and $\widetilde{C}$ partitioned accordingly

$$\widetilde{B} = T^{-1}B = [\,B_1^T, \ldots, B_k^T\,]^T, \qquad \widetilde{C} = CT = [\,C_1, \ldots, C_k\,]. \tag{5}$$

This partition of system matrices is equivalent with the additive decomposition $G = \sum_{i=1}^{k} G_i$, where $G_i(\lambda) = C_i(\lambda I - A_i)^{-1}B_i$, for $i = 1, \ldots, k$, and this is useful for many control computations.

For example, the BDF is useful to compute the exponential of a matrix $A$ using the simple formula

$$\exp(A) = T\exp(\widetilde{A})T^{-1} = T\mathrm{diag}(\exp(A_1), \ldots, \exp(A_k))T,$$

where the exponentials of diagonal blocks are evaluated by Padé approximation.

Another application of BDF is the cheap evaluation for large order systems of the frequency response $G(j\omega)$ or $G(e^{j\omega T})$ for many values of the frequency $\omega$ [20]. For a given frequency value $\omega$, $G(j\omega)$ can be computed as

$$G(j\omega) = \sum_{i=1}^{k} G_i(j\omega), \tag{6}$$

where

$$G_i(j\omega) = C_i(j\omega I - A_i)^{-1}B_i. \tag{7}$$

Recall that $A_i$ is already in a RSF and thus the evaluation of $G_i(j\omega)$ is computationally very cheap because of usually very low order of $A_i$.

3) In several applications, like model reduction of unstable systems or computation of the Hankel-norm of an unstable system it is necessary to use a stable/unstable additive spectral decomposition of a transfer matrix $G$ as $G = G_1 + G_2$, where $G_1$ and $G_2$ are determined such that $G_1$ has only poles in the stable region and $G_2$ has exclusively poles outside that region. For $G = (A, B, C)$ this involves the computation of a transformation matrix $T$ such that the transformed system has the form

$$\left[\begin{array}{c|c} T^{-1}AT & T^{-1}B \\ \hline CT & 0 \end{array}\right] := \left[\begin{array}{cc|c} A_1 & 0 & B_1 \\ 0 & A_2 & B_2 \\ \hline C_1 & C_2 & 0 \end{array}\right],$$

where $A_1$ and $A_2$ contain the systems poles lying in the stable and unstable regions, respectively. The additive terms are then defined by $G_1 := (A_1, B_1, C_1)$ and $G_2 := (A_2, B_2, C_2)$.

4) Given a system $G = (A, B, C)$, one often requires an orthogonal state-space coordinate transformation

$$\widetilde{A} = Q^T AQ, \quad \widetilde{B} = Q^T B, \quad \widetilde{C} = CQ, \tag{8}$$

where $Q$ is chosen to reduce $A$ to a real Schur form $\widetilde{A}$. Such a reduction is frequently necessary as a preprocessing step in the routines for balancing related model reduction.

5) Another useful orthogonal similarity transformation is that reducing the pair $(A, B)$ to its controllability staircase form

$$\widetilde{A} = Q^T AQ = \begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix}, \quad \widetilde{B} = Q^T B = \begin{bmatrix} B_1 \\ B_2 \end{bmatrix}, \quad \widetilde{C} = CQ,$$

where the pair $(A_{11}, B_1)$ is controllable and the matrix $[\, B_1 \;\, A_{11} \,]$ is in a staircase form. Such a transformation is useful for calculating minimal realizations. A dual observability form exists as well.

### 2.1.2 Synthesis

1) The *eigenvalue assignment problem* (EAP) consists of determining a feedback matrix $F \in \mathbb{R}^{m,n}$ such that the closed-loop state matrix $A + BF$ has all its eigenvalues at desired locations $\Gamma = \{\lambda_1, \ldots, \lambda_n\}$ in the complex plane. There exist several numerically stable, which can be used to solve the EAP [12, 13, 9]. All these methods are based on the orthogonal controllability staircase form of the pair $(A, B)$ [16]. An interesting extension of the method in [13] can address parametric pole assignment and is implemented in SLICOT.

An alternative to these methods is the so-called *Schur method* proposed by Varga [18] which uses the *real Schur form* (RSF) of the matrix to accomplish the eigenvalue assignment. Although computationally more involved than the previous ones, the Schur method has the attractive feature to allow a partial pole assignment, *i.e.* it is possible to alter only those eigenvalues of $A$ which are unsatisfactory for the closed-loop system dynamics and to keep unmodified the rest of eigenvalues.

2) Stabilization of a system via feedback or verifying its stability can be done via Lyapunov equations. These equations are linear matrix equations of the type

$$AX + XA^T + Q = 0, \quad A^T X + XA + Q = 0 \tag{9}$$

for continuous-time systems and

$$AXA^T - X + Q = 0 \quad A^T XA - X + Q = 0 \tag{10}$$

for discrete-time systems. We have in the library routines that solve for $X$ when $Q$ is given, or that solve for the Cholesky factor of $X$ when $A$ is stable and $Q$ is positive semi-definite (given in factored form). The method used is based on [6]. These routines are very useful for constructing balancing transformations as well.

3) Another important linear equation in control systems design is the Sylvester equation

$$AX + XB + C = 0 \tag{11}$$

which plays a role in decoupling problems and in observer design. In SLICOT we use the Hessenberg Schur approach to solve this equation because of its good properties of speed and accuracy.

4) For the control of the linear system (1) an optimal state-feedback control law

$$u(t) = Fx(t) \tag{12}$$

can be computed which minimizes the quadratic performance index

$$J = \int_0^\infty [\, x(t)^T Qx(t) + u(t)^T Ru(t)\,] dt \tag{13}$$

in the continuous-time case, or

$$J = \sum_{k=0}^{\infty} [\, x(t)^T Qx(t) + u(t)^T Ru(t)\,] \tag{14}$$

in the discrete-time case, where $Q$ and $R$ are symmetric matrices with $Q \geq 0$ and $R > 0$. In order to solve this problem one requires the solution of the *continuous-time algebraic Riccati equation*

$$Q + A^T X + XA - XBR^{-1}B^T X = 0 \tag{15}$$

and the *discrete-time algebraic Riccati equation*

$$X = A^T XA - A^T XB(R + B^T XB)^{-1} B^T XA + Q. \tag{16}$$

These nonlinear matrix equations can be shown to be equivalent to particular eigenvalue problems with Hamiltonian and symplectic structure. Structure preserving algorithms to solve these eigenvalue problems are available in SLICOT. These are also useful for computing infinity norms of transfer matrices or in methods to compute inner-outer factorizations.

### 2.1.3 Periodic Systems

Periodic systems are linear discrete-time systems of the form

$$\begin{aligned} x_{k+1} &= A_k x_k + B_k u_k \\ y_k &= C_k x_k + D_k u_k \end{aligned} \tag{17}$$

where the matrices $A_k \in \mathbb{R}^{n\times n}$, $B_k \in \mathbb{R}^{n\times m}$, $C_k \in \mathbb{R}^{p\times n}$ and $D_k \in \mathbb{R}^{p\times m}$ are periodic with period $K \geq 1$. Such models arise usually by the discretization of linear continuous-time periodic models which are the primary mathematical descriptions encountered in some practical applications.

Using the notation $\mathcal{M} \doteq \text{diag}\{M_1, M_2, \ldots M_K\}$ and $\sigma\mathcal{M} \doteq \text{diag}\{M_2, \ldots M_K, M_1\}$ one defines periodic similarity transformations as follows. Two periodic systems $(\mathcal{A}, \mathcal{B}, \mathcal{C}, \mathcal{D})$ and $(\widetilde{\mathcal{A}}, \widetilde{\mathcal{B}}, \widetilde{\mathcal{C}}, \mathcal{D})$ related by

$$\widetilde{\mathcal{A}} = (\sigma\mathcal{T})^{-1}\mathcal{A}\mathcal{T}, \quad \widetilde{\mathcal{B}} = (\sigma\mathcal{T})^{-1}\mathcal{B}, \quad \widetilde{\mathcal{C}} = \mathcal{C}\mathcal{T}, \tag{18}$$

are called *similar* and the transformation (18) is called a *periodic similarity transformation*. There exist orthogonal similarity transformations that put a given system in its periodic Schur form which is at the basis of many analysis and design techniques for periodic systems [15]. In this form all matrices $\widetilde{A}_k$ are upper triangular. This transformation is available in SLICOT and is also useful for the computation of structure preserving eigenvalue solvers of Hamiltonian or symplectic structure.

These transformations can be used to solve the discrete periodic Lyapunov equations or construct optimal periodic state-feedback control law $u_k^* = F_k x_k$ can be computed which minimizes the performance index

$$J = \sum_{k=0}^{\infty} [\, x_k^T Q_k x_k + u_k^T R_k u_k \,]. \tag{19}$$

via the solution of a discrete-time periodic Riccati equation.

## 2.2 Factorization of Transfer Function Matrices

For a system $G = (A, B, C, D)$ with the TFM

$$G(\lambda) = C(\lambda I - A)^{-1}B + D$$

a *left coprime factorization* (LCF) is defined as the fractional representation $G = M^{-1}N$, where $N$ and $M$ are stable and proper rational matrices and where there exist stable and proper rational $U$ and $V$ such that $NU + MV = I$. Similarly, a *right coprime factorization* (RCF) is defined as the fractional representation $G = NM^{-1}$, where $N$ and $M$ are stable and proper rational matrices, and where there exist stable and proper rational $U$ and $V$ such that $UN + VM = I$.

A LCF $G = M^{-1}N$ or a RCF $G = NM^{-1}$ with the additional restriction that the denominator factor $M$ is *inner* (that is, $M^{\sim}M = I$, where $M^{\sim}(s) = M^T(-s)$ for a continuous-time system and $M^{\sim}(z) = M^T(1/z)$ for a discrete-time system) are called *left coprime factorization with inner denominator* (LCFID) and *right coprime factorization with inner denominator* (RCFID), respectively. This factorizations are useful in computing $L_2$ and $l_2$ norms of unstable systems.

Several routines have been implemented in SLICOT to perform coprime factorizations of TFMs and to compute the state-space representation corresponding to a RCF or LCF. These routines are useful to perform model reduction of unstable systems using coprime factorization techniques.

### 2.3 Analysis and Synthesis of Descriptor Systems

Linear descriptor systems are described by the model

$$\begin{array}{rcl} E\lambda x(t) & = & Ax(t) + Bu(t) \\ y(t) & = & Cx(t) + Du(t) \end{array}, \tag{20}$$

where $E, A \in \mathbb{R}^{\ell \times n}$, $B \in \mathbb{R}^{\ell \times m}$, $C \in \mathbb{R}^{p \times n}$, $D \in \mathbb{R}^{p \times m}$, and where $\lambda$ is either the differential operator $d/dt$ or the advance operator $z$, depending on the type of the system. In most applications $A$ and $E$ are square matrices (i.e., $\ell = n$), however, even in this case, the matrix $E$ can be generally singular. Descriptor systems can be used to manipulate numerically rational or polynomial matrices. In such cases, we shall assume additionally that the pencil $\lambda E - A$ is regular, that is $\det(\lambda E - A) \not\equiv 0$ and its TFM is the $p \times m$ rational matrix

$$G(\lambda) = C(\lambda E - A)^{-1}B + D. \tag{21}$$

The system (20) is referred to as the quadruple $G = (A - \lambda E, B, C, D)$ or as the triple $G = (A - \lambda E, B, C)$ if $D = 0$.

Two descriptor representations $(A - \lambda E, B, C, D)$ and $(\widetilde{A} - \lambda \widetilde{E}, \widetilde{B}, \widetilde{C}, \widetilde{D})$ related by

$$\widetilde{A} - \lambda \widetilde{E} = Q(A - \lambda E)Z, \quad \widetilde{B} = QB, \quad \widetilde{C} = CZ, \tag{22}$$

where $Q$ and $Z$ are square invertible matrices, are called *similar* and the transformation (22) is called a *similarity transformation*. Note that similar descriptor systems with regular $A - \lambda E$ have the same TFM.

1) Given a descriptor system $G = (A - \lambda E, B, C)$, we can compute diagonal transformation matrices $Q$ and $Z$ to make the rows and columns of the matrices of the transformed system pencil

$$\widetilde{\mathcal{S}}(\lambda) = \left[ \begin{array}{c|c} \widetilde{A} - \lambda \widetilde{E} & \widetilde{B} \\ \hline \widetilde{C} & 0 \end{array} \right]$$

as close in norm to 1 as possible. Such a transformation is frequently necessary to improve the accuracy of numerical computations involving the system matrices. The implemented software for this computation accepts a descriptor description with non-square $A$ and $E$ matrices. Thus, the scaling approach can be used to balance an arbitrary pencil $A - \lambda E$, corresponding formally to a descriptor system with no inputs and no outputs.

2) Orthogonal similarity transformations are useful as preprocessing steps for further reductions of the system matrices, as for example, reducing only $E$ to so-called RQ, QR, QR-like, SVD or SVD-like coordinate forms. The reduction of $E$ to a SVD-like (or complete orthogonal decomposition) form is the first step in the recently developed algorithm to compute the system

zeros [10]. Further reduction of $A$ can be also useful in some computations. For example, the QR-like and SVD, SVD-like reductions of $E$ jointly with the complementary part of $A$ are useful to convert descriptor representations to standard state-space representations by eliminating the non-dynamical part of the system. Several such transformations are implemented as user callable routines in SLICOT.

3) Another useful orthogonal similarity transformation is to reduce the pair $(A-\lambda E, B)$ with regular $A-\lambda E$ to the descriptor controllability staircase form

$$\widetilde{A}-\lambda\widetilde{E} = Q(A-\lambda E)Z = \left[\begin{array}{cc} A_{11}-\lambda E_{11} & A_{12}-\lambda E_{12} \\ 0 & A_{22}-\lambda E_{22} \end{array}\right], \quad \widetilde{B} = Q^T B = \left[\begin{array}{c} B_1 \\ B_2 \end{array}\right], \qquad (23)$$

where the pair $(A_{11}-\lambda E_{11}, B_1)$ is controllable and the pencil $[\, B_1 \; A_{11}-\lambda E_{11} \,]$ is in a staircase form. The pair $(A_{22}, E_{22})$ contains the finite uncontrollable generalized eigenvalues of the pair $(A, E)$. Given a descriptor system $G = (A-\lambda E, B, C)$ an *irreducible* realization of least order $G = (\widehat{A}-\lambda\widehat{E}, \widehat{B}, \widehat{C})$ having the same TFM can be determined by eliminating successively the uncontrollable and unobservable finite and infinite poles using the controllability staircase algorithm of [19].

4) The computation of zeros of a descriptor system can be done by using the structure preserving reduction of the system pencil

$$\mathcal{S}(\lambda) = \left[\begin{array}{c|c} A-\lambda E & B \\ \hline C & D \end{array}\right].$$

to a reduced square pencil

$$\mathcal{S}_r(\lambda) = \left[\begin{array}{c|c} A_r-\lambda E_r & B_r \\ \hline C_r & D_r \end{array}\right],$$

with $D_r$ square and non-singular, having the same finite zeros as the original pencil. The method proposed in [10] to compute zeros uses exclusively orthogonal transformations and is numerically stable. The reduction method allows to additionally determine the complete Kronecker structure of the system pencil (e.g., multiplicities of infinite zeros, left/right Kronecker indices). Software implementing this method is available in SLICOT. Since this software has been developed for more general descriptor representations with $A$ and $E$ possibly non-square, the underlying routine can be also used to determine the Kronecker structure of an arbitrary pencil $A-\lambda E$ (i.e., corresponding formally to a system with no inputs no outputs). Moreover, input-decoupling and output-decoupling zeros can be computed by considering systems with no outputs or no inputs, respectively.

5) Generalized Lyapunov equation for descriptor systems have the forms

$$AXE^T + EXA^T + Q = 0, \quad A^T XE + E^T XA + Q = 0$$

for continuous-time systems and

$$AXA^T - EXE^T + Q = 0, \quad A^T XA - E^T XE + Q = 0$$

for discrete-time systems. Routines for the solution of these equations are available in SLICOT. For the case of non-negative solution, when the pair $(A, E)$ has stable generalized eigenvalues, and when $Q$ is positive semi-definite and given in a factored form, we use an extension of Hammarling's method [6].

## 3 Task I.A.1 : List of Routines

Task I.A.1 consisted of the selection and standardization of basic numerical routines for systems and control. There are now 45 user-callable routines ready that have been standardized in the first 2 years and can be grouped in the following chapters :

- **Mathematical Routines**: Routines for Hamiltonian, symplectic and various other eigenvalue and singular value problems
- **Transformation Routines** : Routines for various state space transformations
- **Analysis Routines** : Routines for transfer function norm calculations
- **Synthesis Routines** : Routines for Lyapunov and Riccati equations
- **Factorization Routines** : Routines for coprime factorizations and state space representations.

In addition to these user-callable routines, a large number of auxiliary routines have been written, standardized and documented, such as Lyapunov, Sylvester and Riccati 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. Together with the user-callable routines the SLICOT library contains more that 100 standardized and documented routines.

The user-callable routines are now listed by chapter.

**Mathematical Routines**

| Name | Function |
|---|---|
| MB03RD | computes the bloc 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 |
| MB03VD | computes the periodic Hessenberg decomposition of a matrix product |
| MB03VY | accumulation of periodic orthogonal transformation to compute the periodic Hessenberg decomposition of a matrix product |
| MB03WD | computes the periodic Schur decomposition of a matrix product in a periodic Hessenberg form |
| MB03WX | computes the eigenvalues of a matrix product in a periodic Schur form |

**Analysis routines**

| Name | Function |
|---|---|
| AB13AD | computes the Hankel norm and the Hankel singular values of the stable projection of a transfer-function matrix |
| AB13AX | computes the Hankel norm of a stable system with the state matrix in real Schur form |
| AB13BD | computes the $H_2$- or $L_2$-norm of a transfer-function matrix |
| AB13CD | computes the $H_\infty$- or $L_\infty$-norm of a system |
| AG08BD | computes the zeros and the Kronecker structure of a descriptor system pencil |

**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) |

**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 |

**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 generalized continuous/discrete Lyapunov equations |
| SG03BD | solves non-negative generalized continuous/discrete Lyapunov equations |
| SB01BD | performs multi-input pole assignment using the Schur method |
| SB01DD | performs parametric multi-input pole assignment using the Hessenberg method |
| SB02QD | estimates the conditioning and a forward error bound for the solution of a continuous-time algebraic Riccati equation |
| SB02RD | solves a continuous-time or discrete-time algebraic Riccati equation and estimate the conditioning and a forward error bound for the solution using the Schur vector method |
| SB02SD | estimates the conditioning and a forward error bound for the solution of a discrete-time algebraic Riccati equation |
| SB03QD | estimates the conditioning and a forward error bound for the solution of a continuous-time Lyapunov equation |
| SB03SD | estimates the conditioning and a forward error bound for the solution of a discrete-time Lyapunov equation |
| SB03TD | solves the real continuous-time Lyapunov matrix equation estimate the conditioning, and compute an error bound on the solution |
| SB03UD | solves the real discrete-time Lyapunov matrix equation estimate the conditioning, and compute an error bound on the solution |

**Descriptor system transformation routines**

| Name | Function |
|---|---|
| TG01AD | performs the scaling of a descriptor system model |
| TG01CD | reduces a descriptor system to the QR-coordinate form with $\widetilde{E} = QE$ upper triangular and $Z = I$ |
| TG01DD | reduces a descriptor system to the RQ-coordinate form with $\widetilde{E} = EZ$ upper triangular and $Q = I$ |
| TG01ED | reduces a descriptor system to the singular value coordinate form with $\widetilde{E} = QEZ$ diagonal, having its singular values as diagonal elements |
| TG01FD | reduces a descriptor system to a singular value like coordinate form with $\widetilde{E} = QEZ$ in a complete orthogonal decomposition form |
| TG01HD | computes the controllability staircase form of a descriptor system |
| TG01ID | computes the observability staircase form of a descriptor system |
| TG01JD | computes an irreducible descriptor representation from a non-minimal one |

# 4 Task I.A.2 : Interfacing Toolboxes

Task I.A.2 makes the above-mentioned basic software tools more "accessible" by implementing them in a user-friendly environment, so that little technical background is required to use the tools to almost full functionality. We have integrated top level routines of SLICOT in MATLAB via mex-files. Since such mex-files are rather big, we minimized 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 was done similarly as for MATLAB.

The routines have been grouped in the following groups :

- **Linear Matrix Equations**: Routines for Lyapunov and Stein equations
- **Generalized Linear Matrix Equations** : Routines for generalized Lyapunov and Stein equations
- **Riccati Equations** : Routines for discrete and continuous time algebraic Riccati equations
- **Realizations** : Routines for controllability, observability and minimal realizations
- **Transformation Routines** : Routines for balancing, Schur form and block diagonal forms.
- **Coprime Factorization Routines** : Routines for constructing state space representations of left and right coprime factorizations.

Most of these routines are described in the Working Note SLWN1999-11 and the files are available via ftp. We list here all the mex files and their corresponding m-files.

| **Linear Matrix Equation (LME) Solver** | | |
|---|---|---|
| `linmeq` | Collection of LME routines in SLICOT | MEX-file. |
| `sldisy` | Solves discrete-time Sylvester equations. | m-files |
| `sllyap` | Solves Lyapunov equations. | |
| `slstei` | Solves Stein equations. | |
| `slstly` | Solves stable Lyapunov equations with factorized right hand side. | |
| `slstst` | Solves stable Stein equations with factorized right hand side. | |
| `slsylv` | Solves Sylvester equations. | |

| **Generalized Linear Matrix Equation (GLME) Solver** | | |
|---|---|---|
| `genleq` | Collection of GLME routines in SLICOT | MEX-file. |
| `slgely` | Solves generalized Lyapunov equations. | m-files |
| `slgesg` | Solves generalized pairs of matrix equations. | |
| `slgest` | Solves generalized Stein equations. | |
| `slgsly` | Solves generalized stable Lyapunov equations. | |
| `slgstst` | Solves generalized stable Stein equations. | |
| `slgesy` | Solves generalized Sylvester equations. | |

| Algebraic Riccati Equation (ARE) Solver | | |
|---|---|---|
| `aresol` | Collection of ARE routines in SLICOT | MEX-file. |
| `slcaregs` | Solves CARE with generalized Schur method. | m-files |
| `slcares` | Solves CARE with Schur method. | |
| `sldaregs` | Solves DARE with generalized Schur method. | |
| `sldares` | Solves DARE with Schur method. | |
| `sldaregsv` | Solves DARE with double sized generalized Schur method. | |

| System Controllability, Observability, Minimal Realization (COM) Forms | | |
|---|---|---|
| `syscom` | Collection of system COM computational routines in SLICOT | MEX-file. |
| `slconf` | Computes controllability staircase form. | m-files |
| `slminr` | Computes minimal realization sub-system. | |
| `slobsf` | Computes observability staircase form. | |

| Transformation Routines | | |
|---|---|---|
| `systra` | Collection of system transformation routines in SLICOT | MEX-file. |
| `slsbal` | Balances the system. | m-files |
| `slsdec` | Transforms state matrix to block diagonal form with ordered eigenvalues. | |
| `slsorsf` | Transforms state matrix to Schur form with ordered eigenvalues. | |
| `slsrsf` | Transforms state matrix to Schur form. | |

| Coprime Factorization Routines | | |
|---|---|---|
| `syscf` | Collection of factorization routines in SLICOT | MEX-file. |
| `lcf` | State-space representation of left coprime factorization. | m-files |
| `rcf` | State-space representation of right coprime factorization. | |
| `lcfid` | State-space representation of left inner coprime factorization. | |
| `rcfid` | State-space representation of right inner coprime factorization. | |

## 5 Task I.A.3 : Benchmarks

Tasks I.A.3 is the selection of benchmarks for task I.A. Six collections of benchmarks have been put together for this task and guidelines for such benchmark collections have been issued :

- Benchmark collections in SLICOT (see NICONET Working Note 1998-5)
- CTDSX, a collection of benchmarks for state-space realizations of continuous-time dynamical systems(see NICONET Working Note 1998-9)
- DTDSX, a collection of benchmarks for state-space realizations of discrete-time dynamical systems(see NICONET Working Note 1998-10)

- CTLEX, a collection of benchmarks examples for continuous-time Lyapunov equations (see NICONET Working Note 1999-6)
- DTLEX, a collection of benchmarks examples for discrete-time Lyapunov equations (see NICONET Working Note 1999-7)
- CAREX, a collection of benchmarks examples for continuous-time algebraic Riccati equations (see NICONET Working Note 1999-14)
- DAREX, a collection of benchmarks examples for discrete-time algebraic Riccati equations (see NICONET Working Note 1999-15)

These collections contain as well examples from real systems as artificial examples that test numerical reliability of the subroutines of our library. The details of the different benchmark collections are described in the respective notes. These collections will also be of valuable help for the other tasks of the project.

## 6 Task I.A.4 : Examples

This task is the selection of industrial design problems. The routines of Task I.A are basic numerical routines that are not directly called in industrial applications but that are needed indirectly through the more advanced Tasks of this Network. For this reason we consider two sets of examples.

The first set are test examples from the benchmark collections. They test the reliability of the numerical methods implemented in the software library. These examples indeed contain a set of examples of which the sensitivity of the computed quantities varies, and they can therefore check if our routines react appropriately to such "difficult" cases. The second set of examples are borrowed from Task II.A since this task uses basic routines from Task I.A to build reduced order models. We refer to NICONET Report 1999-9 *Model Reduction Routines for SLICOT*, for more details about these examples.

Some used industrial benchmark examples for model reduction are:

- PS: Continuous-time power system model ($n = 7$)
- PSD: Discrete-time power system model ($n = 7$)
- TGEN : Nuclear plant Turbo-generator model ($n = 10$)
- ACT: Actuator model ($n = 5$)
- ATTAS: Linearized aircraft model ($n = 55$)
- CDP: CD-player finite element model ($n = 120$)
- GAS: Gasifier models linearized at 0%, 50%, 100% loads ($n = 25$)

Extensive numerical tests are reported in Working Notes 1999-9 and 1999-11. Here we report only a few selected examples which show the improved speed of the SLICOT routines with respect to the corresponding MATLAB routines form the Control Toolbox. These results are based on test examples from the CTDSX Benchmark collection. They compare the SLICOT-based mex-files for controllability, observability and minimality of a state space system (slconf, slobsf, slminr), with the corresponding m-files of the MATLAB Control Toolbox (ctrbf, obsvf, minreal). The table shows the comparison of execution times.

| n | m | p | Time | | Time | | Time | |
|---|---|---|---|---|---|---|---|---|
| | | | slconf | ctrbf | slobsf | obsvf | slminr | minreal |
| 39 | 20 | 19 | <0.01 | 0.05 | 0.05 | 0.05 | 0.05 | 0.11 |
| 100 | 1 | 100 | <0.01 | 2.91 | 0.06 | 0.11 | 0.05 | 6.75 |
| 421 | 211 | 211 | 5.66 | 22.13 | 6.05 | 17.354 | 15.16 | 6694.39 |

For these examples the numerical accuracy of the compared routines does not differ substantially (one digit of accuracy unless the routines yield full accuracy). But the the speed of the routines is clearly in favor of the SLICOT routines. For the last minimum realization problem with dimensions $n = 421$, $m = p = 211$, SLICOT needed 15.16 sec, while Matlab took 6694.39 sec, just for saying that the system is already minimal !

The comparisons made in the Working Notes 1999-9 and 1999-11 show that SLICOT is in general faster, more accurate and more reliable than the comparable MATLAB routines. As a consequence, Mathworks showed interest in including SLICOT software in an improved Control Toolbox (negotiations are still in progress).

# 7 Task I.A.5 : Toolbox

The deliverable for this task is a Basic Software Toolbox, containing all implemented new routines, the accompanying documentation, mex-files developed in this task and a demonstration script. This demo file uses the benchmark examples described in the Working Notes of this Task (1998-9 and 10, 1999-6,7,14 and 15) and the interfacing mex-files described in the Working Note 1999-11. This Demo was used at the European Control Conference of August 1999 in Karlsruhe, where many attendees learned to appreciate the reliability and speed of the numerical tools of NICONET. The Demo is also available in compressed form via ftp at `wgs.esat.kuleuven.ac.be/pub/WGS/SLICOT/Mexfiles`.

# References


[1] W. F. Arnold III and A. J. Laub. Generalized eigenproblem algorithms and software for algebraic Riccati equations. *Proc. of IEEE*, 72:1746–1754, 1984.

[2] P. Benner. *Contributions to the Numerical Solution of Algebraic Riccati Equations and Related Eigenvalue Problems*. Logos–Verlag, Berlin, Germany, 1997. *Also:* Dissertation, Fakultät für Mathematik, TU Chemnitz–Zwickau, 1997.

[3] P. Benner, V. Mehrmann, and H. Xu. A numerically stable, structure preserving method for computing the eigenvalues of real Hamiltonian or symplectic pencils. *Numer. Math.*, 78(3):329–358, 1998.

[4] A. W. Bojanczyk, G. Golub, and P. Van Dooren. The periodic Schur decomposition. Algorithms and applications. In F. T. Luk, editor, *Proceedings SPIE Conference*, volume 1770, pages 31–42, July 1992.

[5] P. M. M. Bongers and P. S. C. Heuberger. Discrete normalized coprime factorization. In A. Bensoussan and J. L. Lions, editors, *Proc. 9th INRIA Conf. Analysis and Optimization of Systems*, volume 144 of *Lect. Notes Control and Inf. Scie.*, pages 307–313. Springer-Verlag, Berlin, 1990.

[6] S. J. Hammarling. Numerical solution of the stable, non-negative definite Lyapunov equation. *IMA J. Numer. Anal.*, 2:303–323, 1982.

[7] B. Kågström and P. Van Dooren. Additive decomposition of a transfer function with respect to a specified region. In *Proc. MTNS Symp., Brussels*, 1989.

[8] V. L. Mehrmann. *The Autonomous Linear Quadratic Control Problem*, volume 163 of *Lect. Notes Contr. Inf. Scie.* Springer Verlag, Berlin, 1991.

[9] G. S. Miminis and C. C. Paige. A direct algorithm for pole assignment of time-invariant multi-input linear systems using state feedback. *Automatica*, 24:343–356, 1988.

[10] P. Misra, P. Van Dooren, and A. Varga. Computation of structural invariants of generalized state-space systems. *Automatica*, 30:1921–1936, 1994.

[11] C. Oara and A. Varga. Inner-outer factorization of rational matrices: the general case. In *Proceedings of the MTNS'98, Padova*, 1998.

[12] R. V. Patel and P. Misra. Numerical algorithm for eigenvalue assignment by state feedback. *Proc. IEEE*, 72:1755–1764, 1984.

[13] P. Hr. Petkov, N. D. Christov, and M. M. Konstantinov. A computational algorithm for pole assignment of linear multiinput systems. *IEEE Trans. Autom. Control*, AC-31:1755–1764, 1986.

[14] V. Sima. *Algorithms for Linear-Quadratic Optimization*, volume 200 of *Pure and Applied Mathematics*. Marcel Dekker, Inc., New York, NY, 1996.

[15] J. Sreedhar, P. Van Dooren, A Schur approach for solving some periodic matrix equations. In *Systems and Networks : Mathematical Theory and Applications*, pages 339–362, Eds. U. Helmke, R. Mennicken, J. Saurer, Mathematical Research, Vol 77, Akademie Verlag, Berlin 1994.

[16] P. Van Dooren. The generalized eigenstructure problem in linear systems theory. *IEEE Trans. Autom. Control*, 26:111–129, 1981.

[17] A. Varga. On stabilization algorithms for linear time-invariant systems. *Rev. Roum. Scie. Techn. – Electrotech. et Energ.*, 26:115–124, 1981.

[18] A. Varga. A Schur method for pole assignment. *IEEE Trans. Autom. Control*, AC-26:517–519, 1981.

[19] A. Varga. Computation of irreducible generalized state-space realizations. *Kybernetika*, 26:89–106, 1989.

[20] A. Varga. Computational techniques based on the block-diagonal form for solving large systems modeling problems. In *Proc. 1993 IEEE Conference on Aerospace Control Systems, Westlake Village, CA*, pages 693–697, 1993.

[21] A. Varga. A Schur method for computing coprime factorizations with inner denominators and applications in model reduction. In *Proc. 1993 American Control Conference, San Francisco, CA*, pages 2130–2131, 1993.

[22] A. Varga. Computation of Kronecker-like forms of a system pencil: Applications, algorithms and software. In *Proc. CACSD'96 Symposium, Dearborn, MI*, pages 77–82, 1996.

[23] A. Varga. Periodic Lyapunov equations: some applications and new algorithms. *Int. J. Control*, 67:69–87, 1997.

[24] A. Varga. Computation of normalized coprime factorizations of rational matrices. *Systems & Control Lett.*, 1998. (to appear).

[25] A. Varga and V. Sima. A numerically stable algorithm for transfer-function matrix evaluation. *Int. J. Control*, 33:1123–1133, 1981.

[26] M. Vidyasagar. *Control System Synthesis: A Factorization Approach.* The MIT Press, Cambridge, MA, 1985.

[27] M. Vidyasagar. Normalized coprime factorization for nonstrictly proper systems. *IEEE Trans. Autom. Control*, 33:300–301, 1988.