LUND UNIVERSITY

Integrating Different Symbolic and Numeric Tools for Linear Algebra and Linear System Analysis

Holmberg, Ulf; Lilja, Mats; Mårtensson, Bengt

1986

Document Version:
Publisher's PDF, also known as Version of record

Link to publication

Citation for published version (APA):
Holmberg, U., Lilja, M., & Mårtensson, B. (1986). *Integrating Different Symbolic and Numeric Tools for Linear Algebra and Linear System Analysis*. (Technical Reports TFRT-7338). Department of Automatic Control, Lund Institute of Technology (LTH).

Total number of authors:
3

General rights
Unless other specific re-use rights are stated the following general rights apply:
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.
• Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
• You may not further distribute the material or use it for any profit-making activity or commercial gain
• You may freely distribute the URL identifying the publication in the public portal

Read more about Creative commons licenses: https://creativecommons.org/licenses/

Take down policy
If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

LUND UNIVERSITY

PO Box 117
221 00 Lund
+46 46-222 00 00

CODEN: LUTFD2/(TFRT-7338)/1–017/(1986)

Integrating Different Symbolic and Numeric Tools for Linear Algebra and Linear System Analysis

Ulf Holmberg
Mats Lilja
Bengt Mårtensson

Department of Automatic Control
Lund Institute of Technology
November 1986

Department of Automatic Control
Lund Institute of Technology
P.O. Box 118
S-221 00 Lund Sweden

Document name
Report

Date of issue
November 3, 1986

Document Number
CODEN: LUTFD2/(TFRT-7338)/1-017/(1986)

Author(s)
Ulf Holmberg, Mats Lilja, Bengt Mårtensson

Supervisor

Sponsoring organisation
The National Swedish Board of Technical Development (STU contract 85-4809)

Title and subtitle
Integrating Different Symbolic and Numeric Tools for Linear Algebra and Linear Systems Analysis

Abstract

There are presently several good programs and program packages for dealing with numeric and symbolic mathematics. This paper centers around integrating these different "worlds": the numerical linear algebra world, the (non-linear) differential equation world, and the world of symbolic manipulation etc. The idea is to allow interaction between several different, existing programs from different origins, in order to solve more composite problems. We also cover interface to computerized type-setting and generation of book-quality figures.

The emphasis in this work is on ideas, not on implementation. We demonstrate the ideas to the following set of programs: The differential equation program Simnon, the matrix-manipulation program CTRL-C, the symbolic manipulation program MACSYMA, the type-setting program TEX, and the page description language PostScript. Of course, the ideas apply to other similar combinations. Formats for communication have been defined, and several utilities written. We also demonstrate macros/functions/programs for solving some more composite problems arizing in linear algebra and linear system analysis, using symbolic and numeric computation. Interface to automatic documentation, TEX, and PostScript are presented. Examples are given.

Key words

Classification system and/or index terms (if any)

Supplementary bibliographical information

ISSN and key title

ISBN

Language
English

Number of pages
17

Recipient's notes

Security classification

The report may be ordered from the Department of Automatic Control or borrowed through the University Library 2, Box 1010, S-221 03 Lund, Sweden, Telex: 33248 lubbis lund.

Integrating Different Symbolic and Numeric Tools for Linear Algebra and Linear System Analysis

Ulf Holmberg

Mats Lilja

Bengt Mårtensson

Department of Automatic Control
Lund Institute of Technology
Box 118, S-221 00 Lund
Sweden

Abstract: There are presently several good programs for dealing with numerical and symbolic mathematics. This paper demonstrates some ideas on how to combine existing programs from different categories. As an illustration, these ideas are applied to combining the differential equation program Simnon, the matrix-manipulation program CTRL-C, the symbolic manipulation program MACSYMA, the type-setting program TEX, and the page description language PostScript. We demonstrate macros, functions and programs for solving more composite problems arising in linear system analysis, using symbolic and numerical computation. Interface to automatic documentation, TEX, and PostScript are presented.

1. Introduction

There are presently several good programs for dealing with numeric and symbolic mathematics and control theory. These live in one of several different "worlds": the numerical linear algebra world, the (non-linear) differential equation world, and the world of symbolic manipulation etc. This work centers around the "integration" of these.

The idea is to allow interaction between several different, existing programs from different origins, in order to solve more composite problems. We want to use existing, well tested programs rather that trying to beat everyone else on all fields simultaneously by writing the program *The Program.* This is also a very important point when discussing expert system interfaces to CAD-programs. In this context, it is natural to cover interface to automatic documentation, computerized type-setting and generation of book-quality figures.

The emphasis in this paper is on ideas, not on implementation. (Details are found in reports cited in the reference list.) The ideas are demonstrated on the following

set of programs: The differential equation program Simnon (see [Åström] or [Åström-Wittenmark]), the matrix-manipulation program CTRL-C, [CTRL-C], the symbolic manipulation program MACSYMA [Mathlab], the type-setting program TEX [Knuth], and the page description language PostScript [Adobe]. Of course, the ideas apply to other similar combinations. Formats for communication have been defined, and several utilities written. We also demonstrate macros, functions, and programs for solving some more composite problems arizing in linear algebra and linear system analysis, using symbolic and numeric computation. Interface to automatic documentation, TEX, and PostScript are presented.

This paper is organized as follows: The next section will describe the communication format used. This is a file for describing the A, B, C, and D-matrices of the standard linear system, here denoted by $\mathcal{S}(A,B,C,D)$:

$$\begin{aligned} \dot{x} &= Ax + Bu, \qquad x \in \mathbb{R}^n, \quad u \in \mathbb{R}^m \\ y &= Cx + Du, \qquad y \in \mathbb{R}^p \end{aligned} \tag{MIMO}$$

or the corresponding discrete-time system. Then "filters" are allowed to operate on these files in Unix-like fashion, creating another file for another program. Section 3 lists the available CTRL-C-functions, mostly for scalar transfer functions. MACSYMA functions for analysis and manipulation of multivariable linear systems, especially in the frequency domain, are presented in Section 4. The next section presents automatic generation of simulation code for multivariable linear systems. Section 6 describes a method for computing and plotting multivariable root loci using interaction between MACSYMA and Simnon.

In this context it is also natural to cover automatic documentation, including generation of high-quality illustrations and typesetting code. Automatic documentation is less error-prone, saves (boring) human labor, and possibly will make more work documented. Some document generation tools are presented in Section 7. Finally, some conclusions are drawn in Section 8.

2. The Communication Format

All communication is done by using text-files. This allows a much greater flexibility than using binary files, since the text-files are human readable and allows modification by any text editor. The drawback of the larger file-sizes has not been considered as a problem, especially since these files are small anyhow.

The $\mathcal{S}(A,B,C,D)$ file is the general protocol for communicating between different programs. It was introduced in [Mårtensson 1984]. Assume that the matrices A, B, C, and D in (MIMO) are

$$A = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$$

$$B = \begin{pmatrix} 5 \\ 6 \end{pmatrix}$$

$$C = \begin{pmatrix} 7 & 8 \end{pmatrix}$$

$$D = \begin{pmatrix} 9 \end{pmatrix}$$

Summary of functions

Frequency response

FRNY	Computes the frequency response of a system with time delay
FRFU	Computes the frequency response for arbitrary transfer functions
NPL1	Plots one Nyquist curve in a new diagram
NPL2	Plots two Nyquist curves in a new diagram
NPL	Plots one Nyquist curve in an old diagram
APL1	Plots one amplitude curve in a new diagram
PPL1	Plots one phase curve in a new diagram
POP1	Plots one Popov curve in a new diagram

Pole placement

DIOP	Solves the DAB equation $AX + BY = C$
RSTC	Solves the pole placement problem
CRST	Simple version of RSTC (without canceling any zeros)
YUCL	Calculates the step response of the closed loop system. A load disturbance (a step) is introduced after half the simulation time.
YUPL	Plots one step response in a new diagram
YUP	Plots one step response in an old diagram

Root locus plots

RLOC	Calculates and plots root locus

Output

TOMIMO	Creates the $S(A, B, C, D)$-file

An Example

Consider the system with transfer function

$$G(s) = \frac{-s+2}{(s+1)(s^2+2s+10)}$$

The objective is to do a pole placement design for which the closed loop poles are placed in $-4, -2 \pm 2i$. The two observer poles are placed in -10. Since a pole-zero cancellation will give raise to an unstable closed loop system in this case, the simpler pole-placer CRST is chosen. We are thus to find polynomials $R(s)$, $S(s)$ and $T(s)$ such that with the control law $Ru = -Sy + Tr$ the closed loop system obtains the desired poles (see [Åström-Wittenmark]). This is done by entering the following commands

```
[>    b = [-1 2]; a = conv([1 1],[1 2 10]);
```

The $S(A,B,C,D)$ file, generated e.g. from CTRL-C by the procedure `TOMIMO` is a text file that looks as

```
NMP    =

     2.     1.     1.

A      =

     1.     2.
     3.     4.

B      =

     5.
     6.

C      =

     7.     8.

D      =

     9.
```

The format for reading the matrix description file is as follows: First three lines are skipped. Then n, m, and p follows. Then three more lines are skipped, and the A-matrix follows. Three more lines are skipped, and the B-matrix follow. Etc... The skipped lines are generally used for textual information and comments. See the preceding example. All numbers are read as real numbers, to be read using free format. [Mårtensson 1986b] and [Holmberg] contains CTRL-C and MACSYMA-function for generating the $S(A,B,C,D)$-file.

3. CTRL-C Functions

The matrix manipulation language CTRL-C contains some functions for handling polynomials. Polynomials are represented as row vectors and polynomial multiplication is implemented as convolution of vectors. There are however presently no control system tools that use a transfer function representation. The next subsection contains a summary of some CTRL-C-functions, which are useful for SISO systems given on transfer function form. This includes functions for calculation and plotting of frequency response, solving of DAB (Diophantine-Aryabhatta-Bezout) equations and plotting of root loci. The root locus function deviates from the tradition of plotting roots of $A(s) + kB(s) = 0$ for equi-distant values of the gain k. Instead the increment in k is calculated in such a way that the distance between two subsequent roots is limited.

For a full description of the presented functions, and some less brief examples, the reader is referred to [Lilja 1986a].

```
[>    i = sqrt(-1);
[>    am = real(poly([-3 -2+2*i -2-2*i]))';
[>    ao = poly(-5*ones(1,2))';
[>    [r,s,t] = crst(a,b,am,ao,1)
```

The step response for the closed loop system can now be plotted:

```
[>    cl1 = yucl(a,b,r,s,t,20);
[>    yupl(cl1)
```

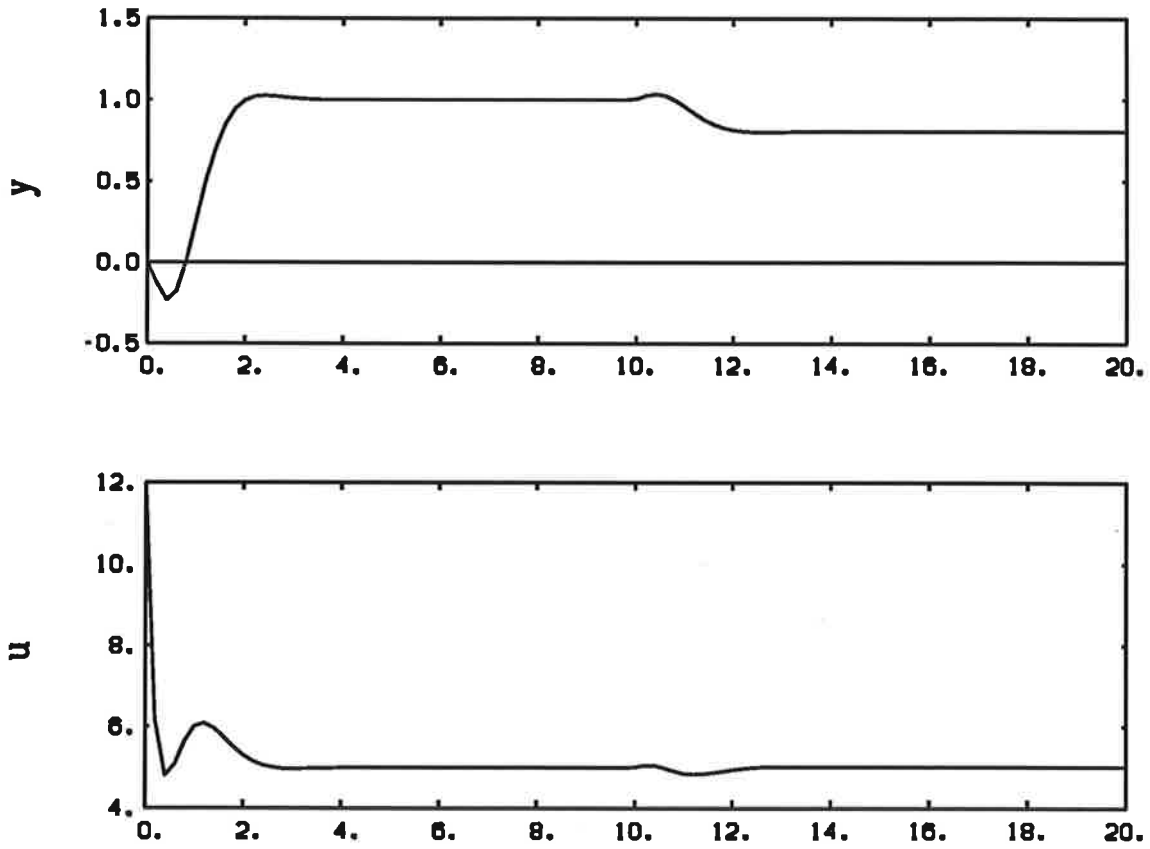


Figure 1. Simulation of closed loop system.

The step response for the first 20 seconds will now appear in the upper diagram and the lower diagram will show the output from the controller. Notice that a load disturbance is introduced at $t = 10$. A new design, for which the regulator dynamics contains an integrator, is easily obtained:

```
[>    ao = poly(-5*ones(1,3))';
[>    [r,s,t] = crst(a,b,am,ao,[1 0]);
```

where the degree of the observer polynomial had to be increased by one. All observer poles are still located in -10.

4. MACSYMA Functions

The framework of polynomial matrices is useful for analysis of multivariable linear systems, see [Kailath]. However, polynomial matrices are not easily manipulated by hand. It is thus very important that good analysis tools for polynomial matrices are available. In this section we try to fill the gap between theory and practice by illustrating the use of MACSYMA as a powerful tool for manipulating polynomial matrices.

A MACSYMA demo with matrix fraction decompositions, co-prime factorizations, multivariable realizations, etc., will illustrate the beauty of symbolic manipulations. The examples are taken from [Kailath]. For further examples and a more elaborate examination of the following MACSYMA functions we refer to [Holmberg].

Summary of MACSYMA Functions

The following functions are available:

Linearization

`LINEARIZE`	Linearizes the dynamical system $\dot{x} = f(x,u), \quad y = g(x,u)$

Stability analysis

`ROUTH`	Generates the stability conditions for a continuous time system
`JURY`	Gives the stability conditions of discrete time systems and the steady state output variance

Sampling

`SAMP`	Sampling from transfer function to pulse-transfer function
`SAMPSTATE`	Sampling from state space to state space

Geometry functions — state space

`HERMITE`	Gaussian elimination when applied to a constant matrix
`KER`	Computes the Kernel $\{X \mid AX = 0\}$
`INVERSE_IMAGE`	Calculates the inverse image $\{X \mid AX = B\}$ (A possibly singular)
`INTERSECTION`	Computes the intersection of two subspaces
`GRAM_SCHMIDT`	Calculates an orthogonal base for a subspace
`AINV`	Computes the maximal A-invariant subspace in a given subspace
`ABINV`	Computes the maximal (A,B)-invariant subspace in a given subspace

Factorization — Frequency domain

`SMITH`	Calculates the Smith form together with transformation matrices
`SMITH_McMILLAN`	Calculates the Smith-McMillan form with transformation
`HERMITE`	Calculates the Hermite form
`COLUMNREDUCE`	Makes a denominator polynomial matrix column reduced
`ROWREDUCE`	Makes a denominator polynomial matrix row reduced
`RMFD`	Right Matrix Fraction Decomposition (MFD) of a transfer matrix
`LMFD`	Left MFD of a transfer matrix
`RIGHTCOPRIME`	Gives a right coprime MFD from a noncoprime MFD
`LEFTCOPRIME`	Gives a left coprime MFD from a noncoprime MFD
`SS2TF`	State space to transfer function conversion
`MAKESYS`	Makes a list of the A, B, C, D matrices to represent a system

Multivariable Realizations

`CONTROLLER`	Calculates a controller form realization
`OBSERVER`	Calculates an observer form realization
`CONTROLLABILITY`	Calculates a controllability form realization
`OBSERVABILITY`	Calculates an observability form realization

Generation of the $S(A,B,C,D)$-file

`TOMIMO`	Generates the file `ABCD.MIM` from A, B, C, and D.

Example—Polynomial Matrix Manipulations

The following example is a MACSYMA *Demo* that illustrate the use of polynomial matrices for analysis of multivariable linear systems. The cumbersome manipulations are done by the above functions. The Demo starts with a transfer function matrix, describing a multivariable linear system. The description is transfered into a matrix fraction decomposition, MFD, i.e. a polynomial matrix description. Extraction of different polynomial matrix factors of the MFD are made. Also, different multivariable realizations are presented. For terminology and a background the reader is refered to [Kailath], especially Chapter 6.

The file shown is a MACSYMA log file, output with the `typeset` switch `true`. The resulting Troff/EQN typesetting code has automatically translated to TEX by the program MacEQ2TEX.

```
(c1) load("login.mac")$
(c2) demo("realizations.dem");
/* This demo describes a couple of examples in Kailath, chapter 6.
Example 6.2-1. Alternative MFDs for a Transfer Function. p. 368-9.
Example 6.4-1. Controller-Form Realization of a Right MFD. p. 407-8.
Example 6.4-2. Observer-Form Realization of a Left MFD. p. 416
Example 6.4-6. Constructing Canonical Controllability Forms. p. 433-4. */
(c3) g:matrix([s/((s+1)*(s+2))^2,s/(s+2)^2],[-s/(s+2)^2,-s/(s+2)^2]);
```

$$\text{(d3)}\qquad \begin{bmatrix} \frac{s}{(s+1)^2(s+2)^2} & \frac{s}{(s+2)^2} \\ -\frac{s}{(s+2)^2} & -\frac{s}{(s+2)^2} \end{bmatrix}$$

```
/* Example 6.2-1. Alternative MFDs for a Transfer Function. p. 368-9. */
(c4) rmfd(g);
```

$$\text{(d4)}\qquad \left[dr = \begin{bmatrix} (s+1)^2(s+2)^2 & 0 \\ 0 & (s+2)^2 \end{bmatrix}, nr = \begin{bmatrix} s & s \\ -s(s+1)^2 & -s \end{bmatrix}\right]$$

```
(c5) ev(rightcoprime(dr,nr),%);
```

$$\text{(d5)}$$

$$\left[dr = \begin{bmatrix} (s+1)^2(s+2)^2 & -(s+1)^2(s+2) \\ 0 & s+2 \end{bmatrix}, nr = \begin{bmatrix} s & 0 \\ -s(s+1)^2 & s^2 \end{bmatrix}, rr = \begin{bmatrix} 1 & 1 \\ 0 & s+2 \end{bmatrix}\right]$$

```
(c6) ev(columnreduce(dr,nr),%);
```

$$\text{(d6)}\quad \left[dr=\begin{bmatrix}0 & -(s+1)^2(s+2)\\ (s+2)^2 & s+2\end{bmatrix}, nr=\begin{bmatrix}s & 0\\ -s & s^2\end{bmatrix}, u=\begin{bmatrix}1 & 0\\ s+2 & 1\end{bmatrix}\right]$$

```
/* Example 6.4-1. Controller-Form Realization of a Right MFD. p. 407-8. */
(c7) ev(real:controller(dr,nr),%);
```

$$\text{(d7)}\quad \left[a=\begin{bmatrix}-4 & -4 & 0 & -1 & -2\\ 1 & 0 & 0 & 0 & 0\\ 0 & 0 & -4 & -5 & -2\\ 0 & 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 1 & 0\end{bmatrix}, b=\begin{bmatrix}0 & 1\\ 0 & 0\\ -1 & 0\\ 0 & 0\\ 0 & 0\end{bmatrix}, c=\begin{bmatrix}1 & 0 & 0 & 0 & 0\\ -1 & 0 & 1 & 0 & 0\end{bmatrix}\right]$$

```
/* Example 6.4-6. Constructing Canonical Controllability Forms. p. 433-4.
*/
/* Search by Crate 2 */
(c8) ev(controllability(a,b,c),%);
```

$$\text{(d8)}\quad \left[a=\begin{bmatrix}0 & 0 & -2 & 0 & 0\\ 1 & 0 & -5 & 0 & 0\\ 0 & 1 & -4 & 0 & 0\\ 0 & 0 & 2 & 0 & -4\\ 0 & 0 & 1 & 1 & -4\end{bmatrix}, b=\begin{bmatrix}1 & 0\\ 0 & 0\\ 0 & 0\\ 0 & 1\\ 0 & 0\end{bmatrix}, c=\begin{bmatrix}0 & 0 & 1 & 1 & -4\\ -1 & 4 & -12 & -1 & 4\end{bmatrix}\right]$$

```
/* Search by Crate 1 */
(c9) crate_nr:1;
```

$$\text{(d9)}\quad 1$$

```
(c10) ev(controllability(a,b,c),real);
```

$$\text{(d10)}\quad \left[a=\begin{bmatrix}0 & 0 & 0 & -4 & 2\\ 1 & 0 & 0 & -12 & 5\\ 0 & 1 & 0 & -13 & 4\\ 0 & 0 & 1 & -6 & 1\\ 0 & 0 & 0 & 0 & -2\end{bmatrix}, b=\begin{bmatrix}1 & 0\\ 0 & 0\\ 0 & 0\\ 0 & 0\\ 0 & 1\end{bmatrix}, c=\begin{bmatrix}0 & 0 & 1 & -6 & 1\\ -1 & 4 & -12 & 32 & -1\end{bmatrix}\right]$$

```
/* Example 6.4-2. Observer-Form Realization of a Left MFD. p. 416 */
(c11) lmfd(g);
```

(d11) $$\left[dl = \begin{bmatrix} (s+1)^2(s+2)^2 & 0 \\ 0 & (s+2)^2 \end{bmatrix}, nl = \begin{bmatrix} s & s(s+1)^2 \\ -s & -s \end{bmatrix}\right]$$

```
(c12) ev(leftcoprime(dl,nl),%);
```

(d12) $$\left[dl = \begin{bmatrix} (s+1)^2(s+2)^2 & 0 \\ (s+1)^2(s+2) & s+2 \end{bmatrix}, nl = \begin{bmatrix} s & s(s+1)^2 \\ 0 & s^2 \end{bmatrix}, rl = \begin{bmatrix} 1 & 0 \\ -1 & s+2 \end{bmatrix}\right]$$

```
(c13) ev(rowreduce(dl,nl),%);
```

(d13) $$\left[dl = \begin{bmatrix} 0 & -(s+2)^2 \\ (s+1)^2(s+2) & s+2 \end{bmatrix}, nl = \begin{bmatrix} s & s \\ 0 & s^2 \end{bmatrix}, u = \begin{bmatrix} 1 & -(s+2) \\ 0 & 1 \end{bmatrix}\right]$$

```
(c14) ev(observer(dl,nl),%);
```

(d14) $$\left[a = \begin{bmatrix} -4 & 1 & 0 & 0 & 0 \\ -4 & 0 & 0 & 0 & 0 \\ 0 & 0 & -4 & 1 & 0 \\ 1 & 0 & -5 & 0 & 1 \\ 2 & 0 & -2 & 0 & 0 \end{bmatrix}, b = \begin{bmatrix} 1 & 1 \\ 0 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, c = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 \end{bmatrix}\right]$$

5. Generation of Simulation Code

This section describes the facility for generation of simulation code in the "differential equation language" Simnon. This is done by the Pascal program Codegen. Since this is extensively documented in [Mårtensson 1986b] this section will be very briefly written.

The program Codegen operates on the $S(A,B,C,D)$-file. For the $S(A,B,C,D)$-file presented in Section 2 it generates the Simnon system:

```
continuous system MIMO

"  This file generated by CODEGEN  at 19:57:48.99    3-AUG-1986

state x1 x2
der dx1 dx2
output y
input u
time t
zero = 0
```

```
dx1=a11*x1+a12*x2+b1*u
dx2=a21*x1+a22*x2+b2*u

y=c1*x1+c2*x2+d*u

a11 :  1.00000E+00
a12 :  2.00000E+00
a21 :  3.00000E+00
a22 :  4.00000E+00

b1 :  5.00000E+00
b2 :  6.00000E+00

c1 :  7.00000E+00
c2 :  8.00000E+00

d :  9.00000E+00

x1 : 0
x2 : 0

end
```

This can be modifyed with any text editor, or used together with other (non-linear) systems in Simnon.

6. Root Locus

In this section we will demonstrate a method for computing root loci by combining symbolic and numerical computations. Only a very brief description will be given. A fuller description is given in [Lilja 1986b].

The naive way for plotting root loci of the type

$$A(s) + kB(s) = 0$$

where A and B are polynomials and k real, is to solve the equation for a number of equi-distant k-values and then mark the roots by an '$\times$' for each k. This method has severe disadvantages: Firstly it is rather time consuming and secondly it gives a very bad resolution near multiple roots. To be presentable, the plots also need heavy manual paste-up. In the following subsection, a method based on the implicit function theorem is suggested. A non-linear differential equation, that describes the root locus locally, is obtained by some manipulations of the transfer function (done in MACSYMA for example) and a package for solving the differential equation (e.g. Simnon) can then be utilized to compute and plot the root locus. This method is both faster and gives a better performance near multiple roots than the method mentioned above.

Some theory

This subsection proposes a method for plotting the locus of points s in the complex plane satisfying the equation

$$f(s,k) = 0, \qquad s,k \in \mathbb{C} \tag{♭}$$

where f is analytic in s and k and where k is restricted to the real axis. Several common control theory problems are covered in this formulation: Ordinary root loci, LQG root loci (k = control weighting), zeros of sampled systems (k = the sampling interval), etc.

The method is based on the implicit function theorem applied to (♭). It states that for all s_0 and k_0 such that

$$f(s_0,k_0) = 0$$
$$\frac{\partial f}{\partial s}(s_0,k_0) \neq 0$$

it holds that in a neighborhood of $(s,k) = (s_0,k_0)$, s can locally be written as a function of k, $s = g(k)$, for some analytic function g with derivative

$$\frac{dg}{dk} = -\frac{\frac{\partial f}{\partial k}}{\frac{\partial f}{\partial s}}$$

This non-linear differential equation can be integrated numerically. The solutions to $f(s,k_0) = 0$ gives the initial values.

The idea is the following: The problem is to compute $\{s|f(s,k) = 0, \quad k \in [a,b] \subset \mathbb{R}\}$. For this, compute the k's such that (♭) has multiple roots in s. Away from these, the branches $s_i(k)$ satisfies

$$\frac{d}{dk}s_i(k) = \frac{\frac{\partial f}{\partial k}}{\frac{\partial f}{\partial s}} \tag{♯}$$

Integrating (♯) numerically, the integration routine will crash when $s_i(k)$ approaches a multiple pole, since the right hand side of (♯) will be infinite there. The algorithm will therefore be, loosely speaking:

1. Find the k_i's giving multiple roots in s of (♭).
2. For all k_i, find k_i^+ and k_i^-, $k_i^- < k_i < k_i^+$, which are suitable points close to k_i, to start and interrupt the integration of (♯).
3. Solve $f(s,k_i^+) = 0$. Denote the solutions with s_i^j.
4. Integrate the differential equation

$$\frac{d}{dk}s(k) = \frac{\frac{\partial f}{\partial k}}{\frac{\partial f}{\partial s}}$$
$$s(k_i^+) = s_i^j$$

for $k \in [k_i^+, k_{i+1}^-]$.

The rest of this section will be devoted to the special case of f being a polynomial, which is the case e.g. in the root locus problem. g will then be a branch of an algebraic function (see e.g. [Postlethwaite-MacFarlane]).

The multiple roots can be found by solving the following system of equations with respect to s and k:

$$\begin{cases} f(s,k) = 0 \\ \dfrac{\partial f}{\partial s} = 0 \end{cases}$$

The k-values corresponding to multiple roots can be found by calculating the discriminant of f with respect to s and solve for the roots in k. The discriminant of a polynomial p (with respect to s) is defined as the resultant of p and dp/ds. The resultant of two polynomials $p(s) = s^n + p_1 s^{n-1} + \cdots + p_n$ and $q(s) = q_1 s^{n-1} + q_2 s^{n-2} \cdots + q_n$ is in turn defined as the determinant of the Sylvester matrix of p and q, $\mathrm{Syl}(p,q)$:

$$\mathrm{Syl}(p,q) := \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 & 0 & \cdots \\ p_1 & 1 & 0 & \cdots & q_1 & 0 & \cdots \\ p_2 & p_1 & 1 & \cdots & q_2 & q_1 & \cdots \\ \vdots & \vdots & \vdots & & \vdots & \vdots & \\ p_n & p_{n-1} & p_{n-2} & \cdots & q_n & q_{n-1} & \cdots \end{pmatrix}$$

Near multiple roots some condition has to be imposed e.g.

$$\left| \frac{\dfrac{\partial f}{\partial k}}{\dfrac{\partial f}{\partial s}} \right| \leq d_{max}$$

for some constant d_{max}. The value of d_{max} is chosen with respect to the integration algorithm being used. The integration is carried out as described above. To find the k-values, k_i^+ and k_i^-, for which equality is attained above, one has to solve a non-linear equation in three real variables. A less tedious way to find the "near-multiple-root" k-values is instead to find an expansion of s in k near the multiple-root values. A systematic method for finding such expansions is the use of so-called Newton diagrams (see [Postlethwaite-MacFarlane]). Given a bivariate polynomial (in our case in s and k), the degree of each term with respect to the two variables is plotted in a two-dimensional integer lattice. The convex hull of the point is then formed. This is a polygon and the different slopes of the lower left faces of this polygon determines the possible behaviors of s for small k. In the general case s and k are replaced by $s + s_0$ and $k + k_0$, where s_0 and k_0 corresponds to a multiple root. The Newton diagram method and its implementation in MACSYMA is discussed in the next section.

A second type of problem appears at the "branching points", i.e. where some of the branches switches to another "sheet" of the Riemann surface. This happens when f has a multiple root with respect to k, which only can happen for multivariable systems. These can be found by solving the equation:

$$\begin{cases} f(s,k) = 0 \\ \dfrac{\partial f}{\partial k} = 0 \end{cases}$$

At these points the derivative dg/dk is zero, i.e. the points are stationary points for the differential equation. Sometimes a branching point coincide with a multiple root is s. In this case the right member of the differential equation can achieve non-zero (even infinite) values at the branching point.

Implementation

The transfer function $G(s)$ is specified in MACSYMA and the closed loop characteristic polynomial $p(s,k)$ is calculated. The real and imaginary part of the right hand side of (♯) are then computed and written to a file using the function `print_ode`. To avoid divide overflow in SIMNON when a multiple root is encountered one has to stop the integration before the multiple root, as mentioned in the previous section. For each multiple root, one therefore has to find the local behavior of s with respect to k. A graphical method to do this is to plot a Newton diagram as mentioned before. This is equivalent to making the substitution $s := bk^a$ in the characteristic polynomial and then finding pairs of dominating terms. The function `newton` implements this and returns two lists. The first list contains the possible k^a-alternatives and the second list gives the corresponding coefficients (expressed as a polynomial in b). The function `near_multiple_roots` uses `newton` for calculating the values of k for which $|g(s,k)| = d_{max}$. The function `print_kxy` uses `near_multiple_roots` to print out these k values and the corresponding solutions in s.

The differential equation for the real and imaginary parts of the root locus is written into the file `ode.rl`. The k-values specifying the intervals for which the root locus is to be plotted for are written (together with the corresponding initial values for the branches of the root locus) into the file `rootloc.rl`. These two files are then processed by a procedure written in the "editor language" TPU (Text Processing Utility) in VMS generating one Simnon system description file `ode.t` (the "dynamics" file) and one file `rootloc.t` containing the commands for setting initial values and integrating.

In the present implementation, the distinction between different sheets of the Riemann surface has not been made in the plot.

An Example

The following MACSYMA dialogue shows an example where the functions `print_ode` and `print_kxy` are used. In the example the interval for the gain k is chosen to $-2 \leq k \leq 2$ and the maximum derivative to $d_{max} = 100$.

```
(c1) load("rootloc.mac")$
(c2) g:matrix([1/s^2,1/s],[-1/s^2,0]);
```

(d2)
$$\begin{bmatrix} \frac{1}{s^2} & \frac{1}{s} \\ -\frac{1}{s^2} & 0 \end{bmatrix}$$

```
(c3) print_ode(g);
```

(d3) *ode.rl*

```
(c4) print_kxy(g,-2,2,100);
```

(d4) *rootloc.rl*

The resulting files `ode.rl` and `rootloc.rl` are then processed by the TPU file `rootloc.tpu` to get the Simnon system description file `ode.t` and the Simnon command file ("macro" file) `rootloc.t`. The Simnon commands required to plot the root locus are:

```
> syst ode
> axes h -2 2 v -2 2
> rootloc
```

The result is shown in Figure 2.

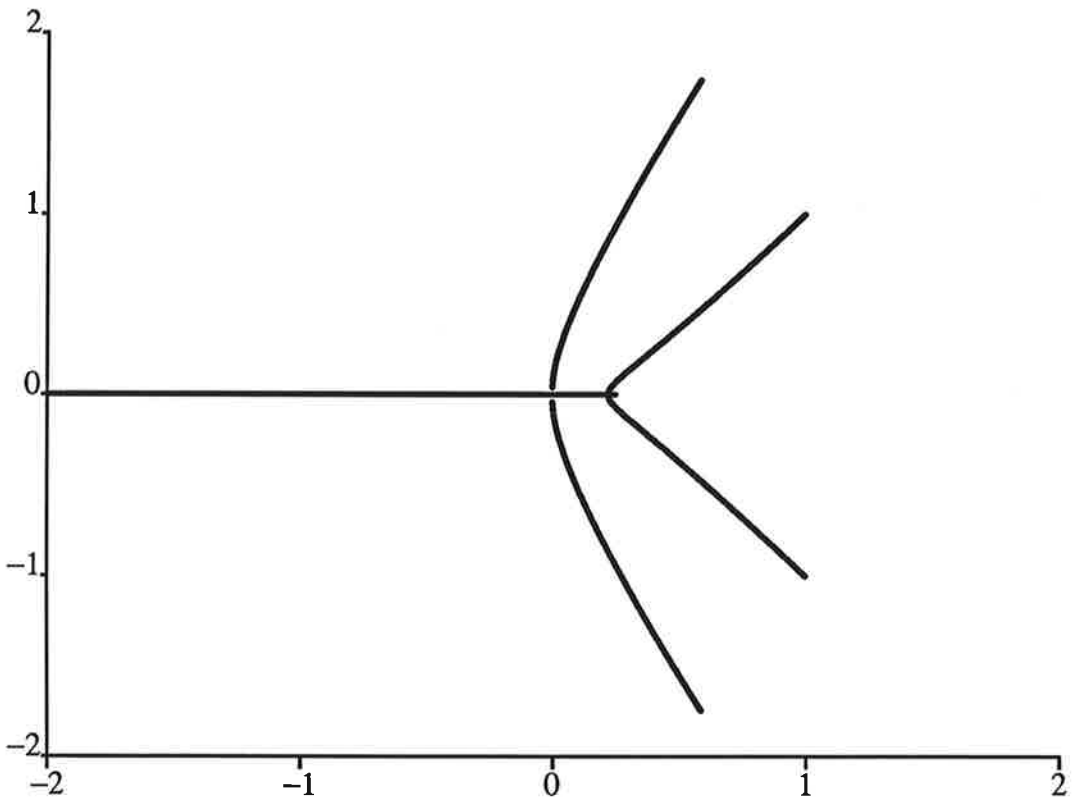


Figure 2. The Root Locus Plot in the Text.

7. Document Generation

Almost all packages of the sort discussed in this paper contains a log-file feature. Also, programs with graphic possibilities generate hard-copy outputs, for pen-plotters or laser printers. Up until now, these have held a quality no book publisher will accept, and therefore illustrations have either been entirely hand made, or machine generated illustrations needing extensive manual paste-up.

Instead, we would like to generate device-independent, book-quality documentation, both typesetting and plots. This is far less error-prone and saves a boring human labor. In this work, we are using the typesetting program TEX [Knuth], and the page description language PostScript, [Adobe].

The following utilities, all written in VAX/VMS Pascal, are presently available:

S2TEX	Generates a TEX-file of the $S(A,B,C,D)$-file.
Hcopy2PS	Generates a PostScript figure of a Simnon plot.
CC2PS	Generates a PostScript figure of a CTRL-C plot.
MacEQ2TEX	Converts the EQN/Troff code generated by MACSYMA to TEX.

The TEX code for the A, B, C, and D matrices in Section 2 of this paper has been generated by S2TEX. Figure 2 has been processed by Hcopy2PS. Sample output from Hcopy2PS is also found in [Mårtensson 1986a,f]. CC2PS has produced Figure 1. The MACSYMA output in this paper has been run through MacEQ2TEX. These programs are documented in [Mårtensson 1986c,d]. See also [Mårtensson 1986e].

A great advantage of these utilities is that they generate human readable text-files (in TEX and PostScript respectively), thus making it possible for manual inspection and modification with a standard text editor.

There are two, conceptually entirely different, uses of automatically generated "log-files": First, it can simply be a logging of your terminal session, which you want to use as documentation of your work for your own records. Secondly, for your books, papers, etc., you want to produce book-quality typesetting and figures. For the first use, quality requirements are not important per se, but rather the ease with which the printed file can be digested. It is our experience that e.g. typeset MACSYMA output is much more easily "digested", than the equivalent line printer output. Therefore, it is believed that this program can be useful not only for producing papers, but also for creative research using MACSYMA for discovering new knowledge. Also, MACSYMA can be used as a convenient TEX code generator for mathematical formulas, available also to the TEX novice.

8. Conclusions

Streamlining the interaction between different programs for symbolic and numeric computation offers great new possibilities. Today, nobody can claim that they can offer you "everything you need" in the form of CAD-packages, simply because of the great effort spent at so many different places, and in so many diverse areas. Therefore, we believe that this principles will show to be of high importance. Automatic document generation also falls naturally in this category.

A large number of functions in CTRL-C and MACSYMA, together with "filters" for communicating between CTRL-C, MACSYMA, Simnon, TEX, and PostScript was presented very briefly. Details are found in [Holmberg], [Lilja 1986ab], and [Mårtensson 1986b–e].

Acknowledgement

The work of U. Holmberg has been partially supported by the National Swedish Board of Technical Development (STU contract 85-4809).

References

ADOBE SYSTEMS INCORPORATED (1985): *PostScript Language Reference Manual*, Addison-Wesley, Reading, Massachusetts.

ÅSTRÖM, K. J. (1985): "A Simnon Tutorial," Report CODEN: LUTFD2/(TFRT-3176)/1–87/(1985), Dept. of Automatic Control, Lund Institute of Technology, Lund, Sweden.

ÅSTRÖM, K. J. and B. WITTENMARK (1984): *Computer Controlled Systems—Theory and Design*, Prentice-Hall Inc., Englewood Cliffs, New Jersey.

CTRL-C (1983): *CTRL-C User's Guide*, Systems Control Technology, Inc. Palo Alto, California, USA.

HOLMBERG U. (1986): "Some MACSYMA Functions for the Analysis of Multivariable Linear Systems," Report CODEN: LUTFD2/(TFRT-7333)/1–40/(1986), Dept. of Automatic Control, Lund Institute of Technology, Lund, Sweden.

KNUTH, D. E. (1984): *The TEXbook*, Addison-Wesley Publishing Company, Reading, Massachusetts.

LILJA. M. (1986a): "Some SISO Transfer Function Facilities in CTRL-C," Report CODEN: LUTFD2/(TFRT-7325)/1–20/(1986), Dept. of Automatic Control, Lund Institute of Technology, Lund, Sweden.

——— (1986b): "Symbolic and Numerical Computation of Implicit Functions, Especially Root Loci," Dept. of Automatic Control, Lund Institute of Technology, Lund, Sweden.

MÅRTENSSON, B. (1984): "Multivariable Linear Systems in Simnon," Report CODEN: LUTFD2/(TFRT-7281)/1–22/(1984), Dept. of Automatic Control, Lund Institute of Technology, Lund, Sweden.

——— (1986a): "Adaptive Stabilization," Report CODEN: LUTFD2/(TFRT-1028)/1–122/(1986), Dept. of Automatic Control, Lund Institute of Technology, Lund, Sweden.

——— (1986b): "CODEGEN—Automatic Simnon Code Generator for Multivariable Linear Systems," Report CODEN: LUTFD2/(TFRT-7323)/1–8/(1986), Dept. of Automatic Control, Lund Institute of Technology, Lund, Sweden.

——— (1986c): "MacEQ2TEX and $\mathcal{S}$2TEX—Automatic TEX-code generation from MACSYMA and CTRL-C," Report CODEN: LUTFD2/(TFRT-7334)/1–11/(1986), Dept. of Automatic Control, Lund Institute of Technology, Lund, Sweden.

——— (1986d): "Hcopy2PS—Hcopy to PostScript Filter," Report CODEN: LUTFD2/(TFRT-7335)/1–7/(1986d), Dept. of Automatic Control, Lund Institute of Technology, Lund, Sweden.

——— (1986e): "Experiences with Computerized Document Preparation Tools," Report CODEN: LUTFD2/(TFRT-7336)/1–XX/(1986), Dept. of Automatic Control, Lund Institute of Technology, Lund, Sweden.

——— (1986f): "Dynamic High-Gain Stabilization of Multivariable Linear Systems, with Application to Adaptive Control," *Proceedings of the SIAM Conference on Linear Algebra in Signals, Systems and Control*, Boston, to appear. Also available as Report CODEN: LUFTD2/(TFRT-7337), Dept. of Automatic Control, Lund Institute of Technology, Lund, Sweden.

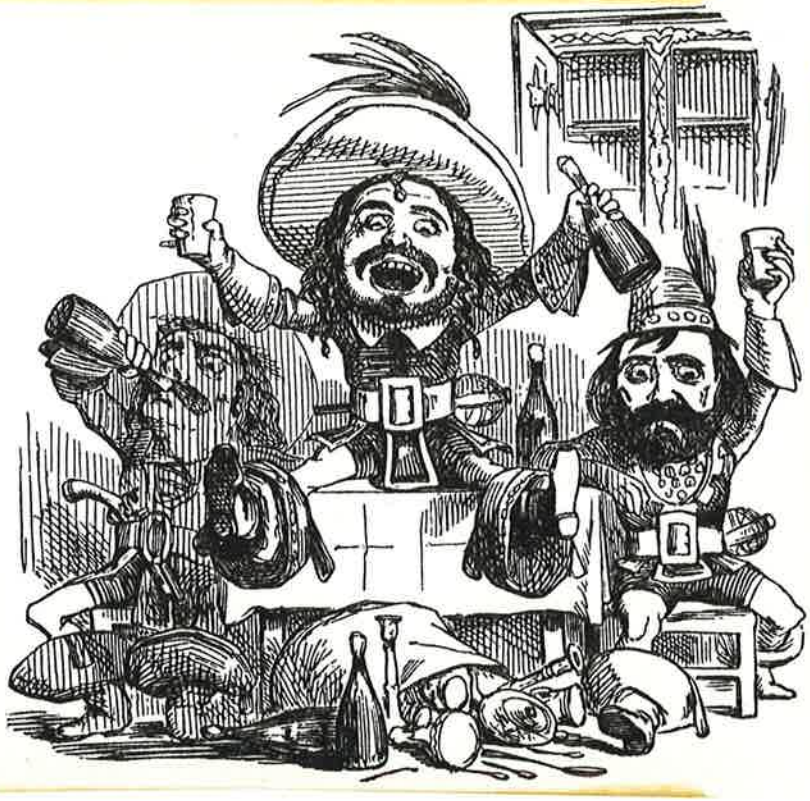
THE MATHLAB GROUP LABORATORY FOR COMPUTER SCIENCE (1983): *MACSYMA Reference Manual*, M.I.T., Cambridge, MA.

POSTLETHWAITE, I. and A. G. J. MACFARLANE (1979): *A Complex Variable Approach to the Analysis of Linear Multivariable Feedback Systems, Lecture Notes in Control and Information Sciences*, 12.

The authors integrating different symbolic and numeric tools for linear algebra and linear system analysis

THE MATHLAB GROUP LABORATORY FOR COMPUTER SCIENCE (1983): *MACSYMA Reference Manual*, M.I.T., Cambridge, MA.

POSTLETHWAITE, I. and A. G. J. MACFARLANE (1979): *A Complex Variable Approach to the Analysis of Linear Multivariable Feedback Systems, Lecture Notes in Control and Information Sciences*, 12.



The authors integrating different symbolic and numeric tools for linear algebra and linear system analysis