CODEGEN -- Automatic Simnon Code Generator for Multivariable Linear Systems

Mårtensson, Bengt

1986

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

Link to publication

Citation for published version (APA):
Mårtensson, B. (1986). *CODEGEN -- Automatic Simnon Code Generator for Multivariable Linear Systems*. (Technical Reports TFRT-7323). Department of Automatic Control, Lund Institute of Technology (LTH).

Total number of authors:
1

CODEN: LUTFD2/(TFRT-7323)/1-008/(1986)

CODEGEN— Automatic Simnon Code Generator for Multivariable Linear Systems

Bengt Mårtensson

Department of Automatic Control
Lund Institute of Technology
August 1986

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

Document name
Report

Date of issue
August 3, 1986

Document Number
CODEN: LUTFD2/(TFRT-7323)/1-008/(1986)

Author(s)
Bengt Mårtensson

Supervisor

Sponsoring organisation

Title and subtitle
CODEGEN—Automatic Simnon Code Generator For Multivariable Linear Systems

Abstract

This paper documents the program CODEGEN that generates Simnon code for multivariable linear systems from a textfile containing the A, B, C, and D matrices. 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
8

Security classification

Recipient's notes

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.

CODEGEN

Automatic Simnon Code Generator for Multivariable Linear Systems

Bengt Mårtensson, 860803

This paper documents the program CODEGEN that generates Simnon code for multivariable linear systems from a textfile containing the A, B, C, and D matrices. Examples are given.

1. Introduction

CODEGEN is a utility for the automatic generation of simulation code in the simulation language Simnon. From a text file describing the integers n, m, p, and the matrices A, B, C, and D in the standard linear system

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

or the corresponding discrete time system

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k); \qquad x \in \mathbb{R}^n; \qquad u \in \mathbb{R}^m \\ y(k) &= Cx(k) + Du(k); \qquad y \in \mathbb{R}^p \end{aligned} \tag{MIMOD}$$

Simnon-code describing the same system is generated. The reader might want to check the examples in Section 4 at this point. All coefficients in the matrices will be available as parameters in the Simnon sense. u will be declared as `input` in the Simnon system, unless a controller is included in the system. It will get the name `MIMO`, unless otherwise specified.

CODEGEN is the code-generating part of the MIMNON-package, described in the report *B. Mårtensson: Multivariable Linear Systems in Simnon*, Department of Automatic Control 1984, TFRT-7278, but isolated from that context, and possible to run as a separate program. Since the Simnon code is a human readable text file which allows additional editing for special modification, this is a very flexible system.

2. Options

There are several different option for using CODEGEN. To use the program on BODE make the definition `codegen == "$scr:[bengt.dirs.exe]codegen"` in your login file. The program is then run by the command `codegen` *argument*, where *argument* is the name of the matrix description file. The default file type is `.mim`. The file name may be omitted, in which case the file name `abcd.mim` is assumed. If the matrix description file is called *filename.filetype*, then the output file will be called *filename.t* if no filename for the output is specified.

Qualifiers

Next the different qualifiers will be described. They can be abbreviated as long as the abbreviations are unique.

`/continuous`
This qualifier states that Simnon code for the continuous time system (MIMOC) should be generated. This option is default.

`/discrete`
This qualifier states that Simnon code for the discrete time system (MIMOD) should be generated.

`/norms`
This will include the (truncated) L^2- (resp. ℓ^2-) norms of both x and y in the Simnon code. See the second example in Section 4.

`/regulator=`*l*
This qualifier signals that a linear controller of order $l \geq 0$ should be included in the Simnon code. This will be

$$\begin{aligned} \dot{z} &= Fz + Gy; \qquad z \in \mathbb{R}^l \\ u &= -Hz - Ky \end{aligned}$$

in the continuous time case, or the corresponding discrete time controller in the discrete time case. The coefficients in the matrices F, G, H, and K are available as parameters in Simnon sense. They are presently all initiated to 0. There will be no input to the Simnon system. *Warning:* If both D and K are present, an algebraic loop will be detected. This situation should be avoided. (Mathematically, of course only the weaker requirement $DK = 0$ is necessary, but Simnon is not smart enough to detect this.)

`/outfile=`*filename*
This directs the output to the file *filename*, instead of the default file, described above. The default file type is `.t`.

`/name=`*system_name*
This will give the Simnon system the name *system_name* instead of the default `MIMO`.

3. The Matrix Description File

A text file is chosen for communicating to the program, thereby allowing manual editing by standard text editors. The format of the matrix description file is as follows: First three lines are skipped. Then n, m, and p follows. Since the program reads them as real numbers, decimal points etc. are OK. 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 example in Section 4. The Appendix contains a CTRL-C macro `tomimo` that will generate this file from CTRL-C, provided that the appropriate matrices are defined, and of compatible dimensions.

4. Examples

Example 1

The file `abcd.mim` has been generated by `tomimo` and looks as follows:

```
NMP    =

    1.     1.     1.

A      =

    1.

B      =

    2.

C      =

    3.

D      =

    0.
```

The command codegen created the following file with the name `abcd.t`

```
continuous system MIMO

"  This file generated by CODEGEN  at 14:32:26.12    5-JUN-1986

state x
der dx
output y
input u
time t
zero = 0

dx=a*x+b*u
```

```
y=c*x

a :  1.00000E+00

b :  2.00000E+00

c :  3.00000E+00

x : 0

end
```

Example 2

The file `system.qwe` has been generated by `tomimo`, renamed, and looks as follows:

```
NMP    =

    3.     2.     3.

A      =

    1.     2.     3.
    4.     5.     6.
    7.     8.     9.

B      =

    1.     2.
    3.     4.
    5.     6.

C      =

    1.    23.     5.
    5.     3.     8.
    5.     3.     0.

D      =

    0.     9.
    8.     5.
    8.     0.
```

The command codegen/disc/norms/reg=2/out=plant system.qwe created the following file with the name plant.t

```
discrete system MIMO

"  This file generated by CODEGEN  at 14:31:17.26    5-JUN-1986

state x1 x2 x3 z1 z2 l2nrm2x l2nrm2y
new nx1 nx2 nx3 nz1 nz2 nl2nrm2x nl2nrm2y
output norm2x norm2y
time t
tsamp ts
zero = 0

nx1=a11*x1+a12*x2+a13*x3+b11*u1+b12*u2
nx2=a21*x1+a22*x2+a23*x3+b21*u1+b22*u2
nx3=a31*x1+a32*x2+a33*x3+b31*u1+b32*u2

y1=c11*x1+c12*x2+c13*x3+d11*u1+d12*u2
y2=c21*x1+c22*x2+c23*x3+d21*u1+d22*u2
y3=c31*x1+c32*x2+c33*x3+d31*u1+d32*u2

norm2x=x1*x1+x2*x2+x3*x3
norm2y=y1*y1+y2*y2+y3*y3

nl2nrm2x = l2nrm2x + norm2x
nl2nrm2y = l2nrm2y + norm2y

nz1=f11*z1+f12*z2+g11*y1+g12*y2+g13*y3
nz2=f21*z1+f22*z2+g21*y1+g22*y2+g23*y3

u1=-h11*z1-h12*z2-k11*y1-k12*y2-k13*y3
u2=-h21*z1-h22*z2-k21*y1-k22*y2-k23*y3

ts = t + deltat


a11  :  1.00000E+00
a12  :  2.00000E+00
a13  :  3.00000E+00
a21  :  4.00000E+00
a22  :  5.00000E+00
a23  :  6.00000E+00
a31  :  7.00000E+00
a32  :  8.00000E+00
a33  :  9.00000E+00
```

```
b11 :  1.00000E+00
b12 :  2.00000E+00
b21 :  3.00000E+00
b22 :  4.00000E+00
b31 :  5.00000E+00
b32 :  6.00000E+00

c11 :  1.00000E+00
c12 :  2.30000E+01
c13 :  5.00000E+00
c21 :  5.00000E+00
c22 :  3.00000E+00
c23 :  8.00000E+00
c31 :  5.00000E+00
c32 :  3.00000E+00
c33 :  0.00000E+00

d11 :  0.00000E+00
d12 :  9.00000E+00
d21 :  8.00000E+00
d22 :  5.00000E+00
d31 :  8.00000E+00
d32 :  0.00000E+00

f11 :  0.00000E+00
f12 :  0.00000E+00
f21 :  0.00000E+00
f22 :  0.00000E+00

g11 :  0.00000E+00
g12 :  0.00000E+00
g13 :  0.00000E+00
g21 :  0.00000E+00
g22 :  0.00000E+00
g23 :  0.00000E+00

h11 :  0.00000E+00
h12 :  0.00000E+00
h21 :  0.00000E+00
h22 :  0.00000E+00

k11 :  0.00000E+00
k12 :  0.00000E+00
k13 :  0.00000E+00
k21 :  0.00000E+00
```

```
k22 :  0.00000E+00
k23 :  0.00000E+00

x1 : 0
x2 : 0
x3 : 0

z1 : 0
z2 : 0

deltat : 1

end
```

5. Discussion

The main flaw with the program as is presently stands is that, even if it as such allows arbitrary large systems (provided that a compilation constant is properly adjusted), Simnon will not allow arbitrarily long lines, or too complex expressions. It has happened that Simnon has given up on lines shorter than 40 characters. However, Mats Lilja at the Department of Automatic Control has written a postprocessor, solving exactly this problem by introducing intermediate variables. It will be documented in a future internal report at the Department.

Ulf Holmberg, also at the Department, has written code for generating the matrix description file from MACSYMA. This will also be documented in a future internal report.

Appendix: The CTRL-C macro TOMIMO

This is the CTRL-C macro tomimo

```
//  tomimo
//  This ctrlc-procedure dumps the matrices A, B, C, D
//  and their dimension to the file abcd.mim.
//  A check is made so that the dimensions are compatible.
//
error = 0;
[n1A,n2A] = size(A);
[nB,mB] = size(B);
[pC,nC] = size(C);
[pD,mD] = size(D);
if n1A <> n2A, error = 1;
if n1A <> nB, error = 1;
if n1A <> nC, error = 1;
if mB <> mD, error = 1;
```

```
if pC <> pD, error = 1;
if error = 1, tx = 'Incompatible dimensions'; display(tx);

if error = 0, nmp = [nB mB pC]; print nmp A B C D >abcd.mim -132;
```