Automatic TeX Code Generation from Macsyma and CTRL-C

Mårtensson, Bengt

1986

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

CODEN: LUTFD2/(TFRT-7334)/1–011/(1986)

Automatic TEX Code Generation from Macsyma and CTRL-C

Bengt Mårtensson

Department of Automatic Control
Lund Institute of Technology
October 1986

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

Document name
Report

Date of issue
October 16, 1986

Document Number
CODEN: LUTFD2/(TFRT-7334)/1-011/(1986)

Author(s)
Bengt Mårtensson

Supervisor

Sponsoring organisation

Title and subtitle
Automatic TEX Code Generation from Macsyma and CTRL-C

Abstract

This paper deals with the automatic generation of typesetting code in TEX from the programs Macsyma and CTRL-C. It consists of two parts: The first part is the documentation of the program MacEQ2TEX, which generates TEX code from EQN/Troff output from Macsyma. The second part is the documentation of the program *S2TEX*, which generates TEX code from a system description file containing the matrices of a multivariable linear system. 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
11

Recipient's notes

Security classification

MacEQ2TEX

Macsyma log file to TEX filter

Bengt Mårtensson, October 15, 1986
Revised November 6, 1986

This paper documents the program MacEQ2TEX that translates a Macsyma log file, containing typeset commands for the Unix typesetting program Troff/EQN, into TEX code.

1. Introduction

Macsyma can generate typesetting code for the Unix typesetting program Troff/EQN by setting the global variable `typeset` to `true`. In this case, the *d*-lines will be output to the screen and to the log file as EQN-code. MacEQ2TEX is a program for translating a Macsyma log file, containing these typesetting commands, into TEXcode. It will turn `alpha` into α, `alfa` into *alfa* do correct square roots, fractions, and matrices, and recognize function such as "sin" to be typeset in roman. An example is given in Section 4. For the format chosen by the present macros, the reader is referred to this example.

It is believed that the program should be possible to use without any knowledge of TEX, except for how to call the program and print out the DVI-file. However, the reader is assumed to have an elementary knowledge of Macsyma and TEX.

Both the Troff/EQN code and the generated TEX code are to be considered as "*dumb*", i.e. might contain not to smart line-breaks, layout etc, etc. Therefore, there are two in principle completely different ways of using this program: It can be used to make your log file more beautiful and easy to read; and secondly, it can be used as a decent first iteration for high-quality typesetting. The second iteration you do yourself with your favorite text editor and TEX. To automatically generate "*smart*" type-setting code for mathematical formulas I consider a task for a smaller expert system. The purpose with this program is to generate "*dumb*" code (the `/index` and `/subscript` qualifiers are an exception from this principle, though). Therefore requests to incorporate new features will most likely be treated cold-hearted. Furthermore, "smartification" is probably better done in using super-editor, such as EMACS.

This paper is compatible with the version of MacEQ2TEX that is dated November 6, 1986. The program consists of approximately 1000 lines of Pascal, and runs under VAX/VMS version 4.x.

Convention. *The escape-character of TEX, "\" in Plain TEX, is represented as "!" in this paper. Furthermore, begin-group and end-group ("{" and "}" in PlainTEX) are*

represented as "<" and ">".

MacEQ2TEX generates TEX-code according to this convention. However, these characters are defined as `const`'s in the Pascal program, and therefore e.g. a PlainTEX version can be created by just changing three lines.

2. Function

Basic Operation

The program is run by the command `maceq2tex`[/⟨*options*⟩] *file_name* where *file_name* is the name of the Macsyma log file. Default file-type is `log`. The filename can also be omitted, in which case the default file name is `macsyma.log`. By default, MacEQ2TEX creates a TEXfile with the name *file_name*`.tex`. If the `tex`-option is selected, the command "`tex` *file_name*" is given after completion.

If the `include` or `tex` qualifier is given, MacEQ2TEX will insert the macro definition file `tex$inputs:macsymac.tex` in the output file. It contains macro definitions necessary for TEX to understand the commands generated by MacEQ2TEX.

Qualifiers

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

`/include` *(Default)* `/noinclude` `/tex`

The `include` qualifier will include the macro file `tex$inputs:macsymac.tex` into the TEXfile as described above. Also the TEX command "`!bye`" will we written at the end of the file. The `/noinclude` qualifier will inhibit this, which is more suitable for generating files for inclusion in documents. The `tex` qualifier will send the generated TEX-file to TEX after completion. The `/tex` qualifier will imply the `/include` qualifier.

`/index`

This qualifier will convert $a10$ to a_{10} etcetera. The precise rule is as follows: If a variable consists of letters, followed by a digit and possibly some extra characters, the conversion will take place. Also parameters with names such as "$\%r1$" will be converted.

`/subscript`

This qualifier will convert ki to k_i etcetera. If a variable consists of exactly two letters, the conversion will take place. This might be desirable in some situations.

`/outfile=`*file_name*

This directs the output to the file *file_name*, instead of the default file, described above. The default file type is `tex`.

Conflicting options are allowed, in which case the rightmost of the conflicting qualifier takes effect. E.g. `/noinclude/include` is equivalent to `/include`. This makes it possible for you to change defaults by defining e.g. `mactex == "''maceq2tex/noinclude'"`.

Qualifiers Not to be Used by Normal Users

There are also some qualifiers that are not to be used by the normal user. They exist for debugging purposes or historical reasons, and might disappear in coming versions.

`/debug`

The `debug` qualifiers will open a log file with the name `debug.log` and will write in it, first the complete conversion list it knows of, then the outcome of every call to `ReadToken` and `TransformToken`. (This behavior might change in the future.)

3. Hints, Discussion, Bugs, Problems, and Possible Improvements

As described in the introduction, there are in principle two different uses of this program. Essentially only the latter one, namely to produce high-quality type-set formulas will be discussed here.

The EQN code looks faily weird on the screen when you are running Macsyma interactively. Therefore, it might be a good idea to first run Macsyma the usual way, then to open the log file, turn on the typesetting and then "playback".

The general idea is to in the generated TEX-file write some general macro call, which the user can (re-) define according to his or her needs or tastes. "Standard" macros are given in Section 5, which will serve as a guide for writing new.

Fairly often, Macsyma's EQN-code generation fails to break an expression (see the example in the next section), and asks you to try to break it yourself with a text editor. Most often, this is simpler to do *before* MacEQ2TEX, in the EQN-code, than in the TEX-code. The reason for this is that MacEQ2TEX will balance occurrences of all left- and right parentheses, braces etc. by inserting the corresponding "`!left.`" and "`!right.`". To introduce a breakpoint, simply write

```
.EN
.EQ
```

on two separate lines of the log file at the place of the desired break.

A particularly "dumb" feature of the generated code is that it uses "`$$`" between all displayed lines in the *d*-lines. This will make them to widely spaced apart if there is more than on line of display in a *d*-line. High-quality type-setting code should instead use the TEX-command `!displaylines` (or equivalent).

MacEQ2TEX typesets names longer than one letter in italics, not math italics. This is done with the TEX macro call `!name`, which is included in the macro file. It also puts a thinspace on both sides of the name. This macro can be redefined according to personal taste.

4. An Example

The following example was run with the `/index` and `/subscript` qualifiers. Note in particular the failure of breaking the long list on line (d7), the overfull `hbox`'es, and the

awfully bad breaks in line (d9). Cf. the comments made above.

```
(c3) a+aa+aaa;
```

(d3) $$aaa + a_a + a$$

```
(c4) x^2+a1*x+a2;
```

(d4) $$x^2 + a_1 x + a_2$$

```
(c5) solve(%,x);
```

(d5) $$\left[x = -\frac{\sqrt{(a_1^2 - 4a_2)} + a_1}{2}, x = \frac{\sqrt{(a_1^2 - 4a_2)} - a_1}{2}\right]$$

```
(c6) x^3+a*x^2+b*x+c;
```

(d6) $$x^3 + ax^2 + bx + c$$

```
(c7) solve(%,x);
 Breakup of expression failed.
 You may try to break it yourself
```

(d7)

$$\left[x = \left(-\frac{\sqrt{3}i}{2} - \frac{1}{2}\right)\left(\frac{\sqrt{(27c^2 + (4a^3 - 18ab)\,c + 4b^3 - a^2b^2)}}{6\sqrt{3}} - \frac{27c - 9ab + 2a^3}{54}\right)^{\frac{1}{3}} + \frac{}{9\left(\frac{\sqrt{(27c^2}}{}\right.}\right.$$

```
(c8) part(1,%);
(c9) part(d7,1);
```

(d9) $$x = \left(-\frac{\sqrt{3}i}{2} - \frac{1}{2}\right)\exp\left(\frac{\sqrt{(27c^2 + (4a^3 - 18ab)\,c + 4b^3 - a^2b^2)}}{6\sqrt{3}}\right.$$

$$-\frac{27c-9ab+2a^3}{54},$$

$$\frac{1}{3}\Bigg)$$

$$+\frac{\left(\frac{\sqrt{3}i}{2}-\frac{1}{2}\right)\left(a^2-3b\right)}{9\left(\frac{\sqrt{\left(27c^2+\left(4a^3-18ab\right)c+4b^3-a^2b^2\right)}}{6\sqrt{3}}-\frac{27c-9ab+2a^3}{54}\right)^{\frac{1}{3}}}$$

$$-\frac{a}{3}$$

```
(c10) a:matrix([cos(phi),-sin(phi),0],[sin(phi),cos(phi),0],[0,0,1]);
```

(d10)
$$\begin{bmatrix}\cos(\phi) & -\sin(\phi) & 0\\ \sin(\phi) & \cos(\phi) & 0\\ 0 & 0 & 1\end{bmatrix}$$

```
(c11) c:subst(psi,phi,%);
```

(d11)
$$\begin{bmatrix}\cos(\psi) & -\sin(\psi) & 0\\ \sin(\psi) & \cos(\psi) & 0\\ 0 & 0 & 1\end{bmatrix}$$

```
(c12) b:matrix([1,0,0],[0,cos(theta),-sin(theta)],[0,sin(theta),cos(theta)]);
```

(d12)
$$\begin{bmatrix}1 & 0 & 0\\ 0 & \cos(\theta) & -\sin(\theta)\\ 0 & \sin(\theta) & \cos(\theta)\end{bmatrix}$$

```
(c13) a . b . c;
```

(d13)
$$\begin{bmatrix}\cos(\phi)\cos(\psi)-\sin(\phi)\sin(\psi)\cos(\theta) & -\sin(\phi)\cos(\psi)\cos(\theta)-\cos(\phi)\sin(\psi) & \sin(\phi)\sin(\theta)\\ \cos(\phi)\sin(\psi)\cos(\theta)+\sin(\phi)\cos(\psi) & \cos(\phi)\cos(\psi)\cos(\theta)-\sin(\phi)\sin(\psi) & -\cos(\phi)\sin(\theta)\\ \sin(\psi)\sin(\theta) & \cos(\psi)\sin(\theta) & \cos(\theta)\end{bmatrix}$$

5. MACSYMAC.TEX

This is the file `macsymac.tex`:

```
% MACSYMAC --- macros for MacEq2TeX
% Bengt Martensson 86-10-06
% LastEditDate: "Thu Nov  6 17:21:32 1986"
<!obeyspaces!gdef <! >>
!def!beginMACSYMAlog <!begingroup
                      !def!!<<!char"21>>%
                      !def!^<<!char"5E>>%
                      !def!~<<!char"7E>>%
       !def!abs<!mathop<!rm abs>!nolimits>
       !def!csc<!mathop<!rm csc>!nolimits>
                      !def!laplace<<!cal L>>
       !def!bold ##1 <!hbox< <!bf ##1> >>
       !let!oldsqrt=!sqrt
       !def!sqrt ##1 <!oldsqrt<##1>>
       !def!name ##1<!,!hbox<<!it ##1>>!,>
                      !begingroup
                      !parskip=0pt!parindent=0pt
                      !obeylines!obeyspaces%
                      !tt>
!def!endMACSYMAlog <!endgroup!endgroup>
!def!beginMacsymaEQ #1<!endgroup
                       !bgroup
                       !def!dotheequationnumber<!leqno<!rm #1>>
                       $$>
!def!endMacsymaEQ<!dotheequationnumber
                  $$
                  !egroup
                  !begingroup
                  !parskip=0pt!parindent=0pt
                  !obeylines!obeyspaces%
                  !tt>
```

6. Revision History

October 20, BM

Vocabulary increased by `cdot` and `abs`. Fixed bug occuring when a double quote (") is not preceded by a space.

November 6, BM

Fixed minor bugs. Increased maximum token length to 35. Increased vocabulary with standard mathematical functions. Fixed bug when "}" is not preceded by a space. Proper handling of `bold`. Changed the handling of names longer than one letter to be typeset in italics, not math italics. Stops gracefully on defect MACSYMA files. `!sqrt` changed.

Acknowledgement

The command decoding is stolen from Leif Andersson. I am very thankful for being able to use this.

S2TEX

Automatic TEX Code Generator for Multivariable Linear Systems

Bengt Mårtensson, October 16, 1986

The conference paper [Holmberg-Lilja-Mårtensson] defines a simple communication protocol for describing multivariable linear systems. This is a text-file containing the A, B, C, and D-matrices. This paper documents the program $\mathcal{S}$2TEX that generates TEX code for the A, B, C, and D matrices.

1. Introduction

The conference paper [Holmberg-Lilja-Mårtensson] defines a simple communication protocol for describing multivariable linear systems. This is a text-file containing the A, B, C, and D-matrices. It makes it possible for the programs Simnon, CTRL-C, and Macsyma to communicate, and to solve more composite problems by using all these three programs. It is also a general format on which other programs can operate. This is further developed in [Holmberg-Lilja-Mårtensson], [Holmberg], [Lilja], and [Mårtensson]. $\mathcal{S}$2TEX is a utility for the automatic generation of typesetting code in the typesetting language TEX.

2. The Matrix Description File

Consider the standard linear system

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

or the corresponding discrete time system

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

The main idea of [Holmberg-Lilja-Mårtensson] is to have a common data-structure, a "$\mathcal{S}(A,B,C,D)$" file, for letting different programs operate on this. 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. [Holmberg] contains Macsyma functions also generating this file. [Mårtensson] describes a program for generating simulation code in the simulation language Simnon from this file.

Since the $S(A,B,C,D)$-file 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 S2TEX. To use the program on BODE make the definition `s2tex == "$scr:[bengt.dirs.exe]s2tex"` in your login file. The program is then run by the command `s2tex [/`⟨*options*⟩`]` *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*.`tex` if no filename for the output is specified.

If the `tex`-option is selected, the command "`tex` *file_name*" is given after completion. If the `/include` or `/tex` qualifier is given, MacEQ2TEX will insert the macro definition file `tex$inputs:sabcdmac.tex` in the output file. It contains macro definitions necessary for TEX to understand the commands generated by S2TEX.

Qualifiers

Next the different qualifiers will be described. They can be abbreviated as long as the abbreviations are unique. The case of the letters is not significant. However, below the qualifieres are written with upper and lower letters for clarity.

`/include` *(Default)* `/noinclude` `/tex`
The `include` qualifier will include the macro file `tex$inputs:sabcdmac.tex` into the TEXfile as described above. Also the TEX command "`!bye`" will we written at the end of the file. The `/noinclude` qualifier will inhibit this, which is more suitable for generating files for inclusion in documents. The `/tex` qualifier will send the generated TEX-file to TEX after completion. The `/tex` qualifier will imply the `/include` qualifier.

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

`/noDmatrix` *(Default)* `/Dmatrix`
If the `/noDmatrix` qualifier is in effect and $D = 0$, no TEX code for D will be generated.

/width=*width*
This qualifiers govern how many positions in the output the entries of the matrices will occupy. Normally, this is entirely uncrucial, and should just be "large enough". Default is 10.

/decimals=*decimals*
This qualifiers allows the user to specify the number of decimals in the output. Default is 0.

4. Examples

The file `abcd.mim` has been generated from CTRL-C by the macro `tomimo` 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 `s2tex` created the file `abcd.tex`, which when run through TEX looks as follows:

$$A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

$$B = \begin{pmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{pmatrix}$$

$$C = \begin{pmatrix} 1 & 23 & 5 \\ 5 & 3 & 8 \\ 5 & 3 & 0 \end{pmatrix}$$

$$D = \begin{pmatrix} 0 & 9 \\ 8 & 5 \\ 8 & 0 \end{pmatrix}$$

5. Discussion

By no means, I consider the above presented example as the most beautiful, or definitive way of presenting four matrices. The generated code should be considered as "dumb", free to the user to manually modify with his or her favorite text editor. E.g. the provided macros are only one possibility.

Probably it would have been a much better idea to write this program in Gnu-Emacs directly, or in Lisp.

Acknowledgement.

The command decoding has been stolen from Leif Andersson. It is a pleasure to thank him for this excellent code.

References

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

HOLMBERG, U., M. LILJA, and B. MÅRTENSSON (1986): "Integrating Different Symbolic And Numeric Tools for Linear Algebra and Linear Systems Analysis," *Proceedings of the Siam Conference on Linear Algebra in Signals, Systems and Control, Boston, August 1986*, To appear.

LILJA, M. (1986): "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.

MÅRTENSSON, B. (1986): "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.

Appendix 1: The TEX macro sabcdmac

```
% Macro for S(A,B,C,D) - file
%
!def!beginSABCD #1 !endSABCD<!begingroup
      !def!Matrix ##1<!left!lgroup!matrix<##1>!right!rgroup>
      !def!Amatrix ##1<A &= !Matrix<##1>!cr>
      !def!Bmatrix ##1<B &= !Matrix<##1>!cr>
      !def!Cmatrix ##1<C &= !Matrix<##1>!cr>
      !def!Dmatrix ##1<D &= !Matrix<##1>!cr>
      $$
      !eqalign<#1>
      $$
      !endgroup
    >
!def!endSABCD<!relax>
```

Appendix 2: 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;
```