A Quick and Dirty Guide to *EMaxima*

Jay Belanger

1 Introduction to EMaxima

EMaxima is a major mode for Emacs that allows the user to write documents while interacting with *Maxima*. It is based on Dan Dill's TEX/*Mathematica* package[1], and uses a modified version of William Schelter's `maxima.el`. While the `maxima-mode` provided by `maxima.el` is designed to help write *Maxima* programs, *EMaxima* is designed to help write documents that include *Maxima* code. *EMaxima* is an extension of the LATEX mode provided by AUCTEX[2], and so has the LATEX mode commands available. The resulting document can be processed by LATEX; this requires putting

```
\usepackage{emaxima}
```

in the preamble.

2 Cells

The basic unit of *Maxima* code in *EMaxima* is a **cell**. A cell consists of text between the delimiters

```
\beginmaxima
```

and

```
\endmaxima
```

A cell can be created by typing `C-c C-o`. (The `C-o` in this case stands for **o**pening a cell.) The delimiters will then be placed in the buffer, and the point will be placed between them.

When working with several cells, you can jump between them by using `C-c +` to go to the next cell and `C-c -` to go to the previous cell.

[1] TEX/*Mathematica* is available from `ftp://chem.bu.edu/pub/tex-mathematica-2.0`.

[2] This can be configured so that *EMaxima* extends the standard TEX mode provided by Emacs, or just text mode.

3 Evaluating cells

To evaluate the contents of a cell, the command `C-cC-uc` (`emaxima-update-cell`)[3] will send the contents of the cell to a *Maxima* process (if there is no *Maxima* process running, one will be started) and return the results to the cell, separated from the input by the marker

```
\maximaoutput
```

To differentiate $sin(x^2)$, for example, type `diff(sin(x^2),x);` in a cell:

```
\beginmaxima
diff(sin(x^2),x);
\endmaxima
```

After typing `C-c C-u c`, it will look like

```
\beginmaxima
diff(sin(x^2),x);
\maximaoutput
                                         2
                                2 x COS(x )
\endmaxima
```

To delete the output and return the cell to its original form, you can use the command `C-c C-d`. If the document is to be TEXed, the above cell will look like:

```
Maxima
diff(sin(x^2),x);
```

and the cell with output will look like:

```
Maxima
diff(sin(x^2),x);
Output
                                         2
                                2 x COS(x )
```

EMaxima mode can take advantage of the fact that *Maxima* can give its output in LATEX form. The command `C-c C-u C` works the same as `C-c C-u c`, except now the output is in LATEX form, ready to be formatted by LATEX. In general, if `C-c C-u` *letter* returns *Maxima* output, then `C-c C-u` *capital letter* will return the output in TEX form. The above cell would become

[3]Sending the cells contents to a *Maxima* process and returning the results is called **updating** the cell, the prefix `C-c C-u` will be used to update cells in different ways.

```
\beginmaxima
diff(sin(x^2),x);
\maximatexoutput
$$   2\*x\*\cos x^{2} $$
\endmaxima
```

which, when LaTeXed, would become

Maxima

```
diff(sin(x^2),x);
```

TeX Output

$$2x\cos x^2$$

(Note that whenever a cell is updated, any old output is discarded and replaced with new output.) The command `C-c C-u a` will update all of the cells in your document, stopping at each one to ask if you indeed want it updated. Given an argument, `C-u C-c C-u a`, it will update all of the cells in your document without asking. The command `C-c C-u A` behaves similarly, except now all the output is returned in LaTeX form.

4 Initialization Cells

It is possible that you want certain cells evaluated separate from the others; perhaps, for example, you want certain cells evaluated whenever you open the document. This can be done using initialization cells. An initialization cell is delimited by

```
\beginmaxima[* Initialization Cell *]
```

and

```
\endmaxima
```

The command `C-c C-t` will turn a cell into an initialization cell, applying `C-c C-t` again will turn it back into a regular cell. When LaTeXed, an initialization cell will look like

Maxima Initial

```
diff(sin(x^2),x);
```

Initialization cells behave like regular cells, except that they can be treated as a group. To evaluate all initialization cells (without displaying the output in the document buffer), the command `C-c C-u t` will go to each of the initialization cells and evaluate them. If you want the output of the initialization cells to be brought back to the document buffer, stopping at each one to see it you indeed want it updated, then use the command `C-c C-u i`. With an argument, `C-u C-c C-u i`, the initialization cells will be updated without asking. The command `C-c C-u I` behaves just like `C-c C-u i`, except that the output is returned in TEX form.

5 Referencing Other Cells

Instead of *Maxima* code, a cell can contain a reference to another cell, and when the original cell is sent to *Maxima*, the reference is replaced by the referenced cell's contents (but only in the *Maxima* process buffer, the cell's content in the document's buffer is not changed). In order to do this, the original cell must be marked by having a label of the form <*filename*:*cell label*>. (The reason for the *filename* will become apparent later, and *cell label* is optional for the referencing cell.) The referenced cell must also be labeled, with the same *filename* but a unique *cell label*. To reference the other cell, the original cell need only contain the marker for the referenced cell. For example, given cell 1:

```
\beginmaxima<filename:optional>
<filename:definef>
diff(f(x),x);
\endmaxima
```

and cell 2:

```
\beginmaxima<filename:definef>
f(x):=sin(x^2);
\endmaxima
```

then the result of updating cell 1 (`C-c C-u c`) will be:

```
\beginmaxima<filename:optional>
<filename:definef>
diff(f(x),x);
\maximaoutput
                                                 2
                                   f(x) := SIN(x )
                                               2
                                     2 x COS(x )
\endmaxima
```

When LATEXed, the top line will contain a copy of the marker.

```
Maxima ──────────── <filename:optional> ────────────
<filename:definef>
diff(f(x),x);
──── Output ────
                                         2
                          f(x) := SIN(x )
                                       2
                            2 x COS(x )
```

A cell can contain more than one reference, and referenced cells can themselves contain references.

To aid in labelling the cells, the command `C-c C-x` will prompt for a label name and label the cell. To aid in calling references, the command `C-c C-TAB` can be used for completing the the *filename* and *cell label* parts of a reference, based on the current labels. Another option is to set the Emacs variable `emaxima-abbreviations-allowed` to `t`, say, by putting the line

```
(setq emaxima-abbreviations-allowed t)
```

in your `.emacs` file. This will allow the *filename* and *cell label* parts of a reference to be abbreviated by enough of a prefix to uniquely identify it, followed by ellipses `...` For example, if there are cells labelled

```
<filename:long description>
<filename:lengthy description>
```

Then the reference

```
<...:le...>
```

will suffice to refer to the second label above.

If you want the references in a cell to be replaced by the actual code, the command `C-c @` will expand all the references and put the code into a separate buffer (so it will not affect the original document).

6 WEB

The reason for the ability to reference other cells is so that you can write what Donald Knuth calls literate programs. The idea is that the program is written in a form natural to the author rather than natural to the computer. (Another aspect of Knuth's system is that the code is carefully documented, hence the name "literate programming", but that is done naturally in *EMaxima*.) Knuth called his original literate programming tool `WEB`, since, as he puts it, "the structure of a software program may be thought of as a web that is made up of many interconnected pieces." *EMaxima*'s ability in this respect is taken directly from TEX/*Mathematica*, and is ultimately based on `WEB`. To create a program, the “base cell” or “package cell” should contain a label of the form <*filename*:> (no cell label), and can contain references of the form <*filename*:*part*> (same file name as the base cell).

As a simple (and rather silly) example, suppose we want to create a program to sum the first n squares. We could start:

```
\beginmaxima<squaresum.max:>
squaresum(n) := (
  <squaresum.max:makelist>
  <squaresum.max:squarelist>
  <squaresum.max:addlist>
  );
\endmaxima
```

We would then need cells

```
\beginmaxima<squaresum.max:makelist>,
L:makelist(k,k,1,n),
\endmaxima

\beginmaxima<squaresum.max:squarelist>
<squaresum.max:definesquare>
L:map(square,L),
\endmaxima

\beginmaxima<squaresum.max:addlist>
lsum(k,k,L)
\endmaxima
```

and then we would also need:

```
\beginmaxima<squaresum.max:definesquare>
square(k) := k^2,
\endmaxima
```

When TeXed, the header of the cell will say that it determines the file `squaresum.mu`.

Maxima — Definition of package squaresum.max

```
squaresum(n) := (
  <squaresum.max:makelist>
  <squaresum.max:squarelist>
  <squaresum.max:addlist>
  );
```

The command `C-u C-c @` will put all the pieces together in the file it determines. The resulting file, in this case, will be `squaresum.max` and will look like:

```
squaresum(n) := (
  L:makelist(k,k,1,n),
  square(k) := k^2,
  L:map(square,L),
  lsum(k,k,L)
  );
```

(Although the idea is that only the computer need look at this file.)

7 Other types of cells

When a cell is TEXed, the input and output are kept separate. To have the results look like a *Maxima* session, there are, in addition to the standard cells, special cells called *session cells*. A session cell is delimited by

```
\beginmaximasession
```

and

```
\endmaximasession
```

The command `C-c C-p` will create a session cell. When a session cell is updated, the output will be marked off with `\maximasession`, and will contain both the input and the output, with the *Maxima* prompts. For example, if the session cell

```
\beginmaximasession
diff(sin(x),x);
int(cos(x),x);
\endmaximasession
```

were updated, the result would look like

```
\beginmaximasession
diff(sin(x),x);
integrate(cos(x),x);
\maximasession
(C1)diff(sin(x),x);

(D1)                                COS(x)
(C2)integrate(cos(x),x);

(D2)                                SIN(x)
\endmaximasession
```

which, when TEXed, would look like

```
(C1)diff(sin(x),x);
(D1)                                COS(x)
```

```
(C2)integrate(cos(x),x);
(D2)                                SIN(x)
```

If it is updated in TEX form, it will look like

```
\beginmaximasession
diff(sin(x),x);
integrate(cos(x),x);
\maximatexsession
\C1.  diff(sin(x),x); \\
\D1.   \cos x \\
\C2.  integrate(cos(x),x); \\
\D2.   \sin x \\
\endmaximasession
```

which, when TEXed, will look like

(C1) `diff(sin(x),x);`
(D1)

$$\cos x$$

(C2) `integrate(cos(x),x);`
(D2)

$$\sin x$$

For particularly long output lines inside the `\maximatexsession` part of a session cell, the command `\DD` will typeset anything between the command and `\\`. Unfortunately, to take advantage of this, the output has to be broken up by hand. If a session cell has not been updated, or has no output for some other reason, it will not appear when the document is TEXed.

There is one other type of cell, a *noshow cell*, which can be used to send *Maxima* a command, but won't appear in the TEXed output. A noshow cell can be created with `C-c C-n`, and will be delimited by

```
\beginmaximanoshow
```

and

```
\endmaximanoshow
```

Session cells and noshow cells cannot be initialization cells or part of packages.[4]

If the command to create one type of cell is called while inside another type of cell, the type of cell will be changed. So, for example, the command `C-c C-p` from inside the cell

[4]That could be changed, but I don't know why it'd be useful.

```
\beginmaxima
diff(x*sin(x),x);
\endmaxima
```

will result in

```
\beginmaximasession
diff(x*sin(x),x);
\endmaximasession
```

If a standard cell is an initialization cell or a package part, its type cannot be changed.

8 Miscellaneous

Some *Maxima* commands can be used even outside of cells. The command `C-c C-u l` send the current line to a *Maxima* process, comment out the current line, and insert the *Maxima* output in the current buffer. The command `C-c C-u L` will do the same, but return the result in LaTeX form.

The command `C-c C-h` will provide information on a prompted for function (like *Maxima*'s `describe`), and `C-c C-i` will give the *Maxima* info manual.

Finally, the *Maxima* process can be killed with `C-c C-k`.

A Installation

The *EMaxima* package consists of the files `maxima.el`, `emaxima.el`, `maxima-font-lock.el`, `emaxima.sty` and `emaxima.lisp`. To install, place the `.el` files, as well as `emaxima.lisp`[5] somewhere in the load path for Emacs. Finally, if you want to run LaTeX on the resulting document, put `emaxima.sty` somewhere in the TeX inputs path. If you use pdflatex, you'll also need `pdfcolmk.sty`.

To make sure that `emaxima.el` is loaded when necessary, the line

```
(autoload 'emaxima-mode "emaxima" "EMaxima" t)
```

can be inserted into your `.emacs` file. Then typing `M-x emaxima-mode` will start *EMaxima* mode. The command `M-x emaxima-mark-file-as-emaxima` will put the line

```
%-*-EMaxima-*-
```

at the beginning of the file, if it isn't there already, and will ensure that the next time the file is opened, it will be in `emaxima-mode`. This can be done automatically everytime a file is put in `emaxima-mode` by putting the line

```
(add-hook 'emaxima-mode-hook 'emaxima-mark-file-as-emaxima)
```

somewhere in your `.emacs` file.

B Customizing EMaxima

There are a few (very few) things that you can do to customize *EMaxima*.

By default, *EMaxima* is an extension of AUCTeX mode. This can be changed by changing the variable `emaxima-use-tex`. The possible values are `'auctex`, `'tex` and `nil`. Setting `emaxima-use-tex` (the default) to `'auctex` will make *EMaxima* an extension of AUCTeX, setting it to `'tex` will make *EMaxima* an extension of Emacs's default TeX mode, and setting `emaxima-use-tex` to `nil` will make *EMaxima* an extension of text-mode. So, for example, putting

```
(setq emaxima-use-tex nil)
```

in your `.emacs` file will make *EMaxima* default to an extension of text mode.

Whether or not the dots (...) abbreviation is allowed in cell references is controlled by the elisp variable `emaxima-abbreviations-allowed`, which is set to `t` by default. Setting this to `nil` will disallow the abbreviations, but will speed up package assembly.

The LaTeXed output can also be configured in a couple of ways. The lines that appear around cells when the document is TeXed can be turned off with the command (in the LaTeX document)

```
\maximalinesfalse
```

[5] If Emacs cannot find `emaxima.lisp`, then the TeX output functions will not work, any attempts to get TeX output will only result in standard output.

They can be turned back on with the command

```
\maximalinestrue
```

The fonts used to display the *Maxima* input and output in a cell are by default `cmtt10`. They can be changed, seperately, by changing the TEX values of `\maximainputfont` and `\maximaoutputfont`. So, for example, to use `cmtt12` as the input font, use the command

```
\font\maximainputfont = cmtt12
```

The spacing in the cells can be controlled by changing the TEX variables `\maximainputbaselineskip` and `\maximaoutputbaselineskip`, and so to increase the space between the lines of the output, the command

```
\maximaoutputbaselineskip = 14pt
```

could be used. The amount of space that appears before a cell can be changed by changing the value of `\premaximaspace` (by default, 0pt), and that after a cell can be changed by changing the value of `\postmaximaspace` (by default, 1.5 ex).

Session cells can be configured similarly. Lines can be placed around a *Maxima* session with the command

```
\maximasessionlinestrue
```

and they can be turned back off with

```
\maximasessionlinesfalse
```

The font can be changed by changing the value of `\maximasessionfont`. The color of the prompts when the session is in TEX form is controlled by `\maximapromptcolor`, by default red, the colors of the input lines and output lines are controlled by `\maximainputcolor` and `\maximaoutputcolor`, respectively. So the command

```
\def\maximainputcolor{green}
```

would make the input in a TEXed session green. The session can be TEXed without the colors by using the command `\maximasessionnocolor`. The baselineskip is set by `\maximasessionbaselineskip` for normal session cells, and by `\maixmatexsessionbaselineskip` for TEX sessions. The amount of space that appears before a session cell can be changed by changing the value of `\premaximasessionspace` (by default, 0pt), and that after a cell can be changed by changing the value of `\postmaximasessionspace` (by default, 1.5 ex).

C Maxima mode

Maxima mode is a major mode for writing *Maxima* code. For moving around in the code, *Maxima* mode provides the following motion commands:

`M-C-a`	Move to the beginning of the form.
`M-C-e`	Move to the end of the form.
`M-C-b`	Move to the beginning of the sexp.
`M-C-f`	Move to the end of the sexp.

and the following miscellaneous commands:

`C-c C-f`	Mark the current form.
`C-c )`	Check the current region for balanced parentheses.
`C-c C-)`	Check the current form for balanced parentheses.

Maxima mode has the following completion command:

`M-TAB`	Complete the Maxima symbol as much as possible, providing a completion buffer if there is more than one possible completion. (If the variable `maxima-use-dynamic-complete` is non-nil, then `M-TAB` will cycle through possible completions.

Portions of the buffer can be sent to a Maxima process. (If a process is not running, one will be started.)

`C-cC-r`	Send the region to Maxima.
`C-cC-b`	Send the buffer to Maxima.
`C-cC-c`	Send the line to Maxima.
`C-cC-e`	Send the form to Maxima.
`C-RET`	Send the smallest set of lines which contains the cursor and contains no incomplete forms, and go to the next form.
`M-RET`	As above, but with the region instead of the current line.
`C-cC-l`	Prompt for and load a Maxima file.

When something is sent to Maxima, a buffer running an inferior Maxima process will appear. It can also be made to appear by using the command `C-c C-p`. If an argument is given to a command to send information to Maxima, the region (buffer, line, form) will first be checked to make sure the parentheses are balanced. The Maxima process can be killed, after asking for confirmation with `C-cC-k`. To kill without confirmation, give `C-cC-k` an argument.

The behaviour of indent can be changed by the command `M-x maxima-change-indent-style`. The possibilities are:

Standard Standard indentation.

Perhaps smart Tries to guess an appropriate indentation, based on open parentheses, "do" loops, etc. A newline will re-indent the current line, then indent the new line an appropriate amount.

The default can be set by setting the value of the variable `maxima-newline-style` to either `'standard` or `'perhaps-smart`. In all cases, `M-x maxima-untab` will remove a level of indentation.

To get help on a Maxima topic, use `C-c C-d`. To read the Maxima info manual, use `C-c C-m`. To get help with the symbol under point, use `C-cC-h`. To get apropos with the symbol under point, use `C-cC-a`.

D Running Maxima Interactively

To run *Maxima* interactively in a buffer, type `M-x maxima`. In the *Maxima* process buffer, return will check the line for balanced parentheses, and send line as input. Control return will send the line as input without checking for balanced parentheses. The following commands are also available.

`M-TAB`	Complete the Maxima symbol as much as possible, providing a completion buffer if there is more than one possible completion. (If the variable `maxima-use-dynamic-complete` is non-nil, then `M-TAB` will cycle through possible completions.
`C-M-TAB`	Complete the input line, based on previous input lines.
`C-c C-d`	Get help on a Maxima topic.
`C-c C-m`	Bring up the Maxima info manual.
`C-cC-k`	Kill the process and the buffer, after asking for confirmation. To kill without confirmation, give `C-cC-k` an argument.
`M-p`	Bring the previous input to the current prompt.
`M-n`	Bring the next input to the prompt.
`M-r`	Bring the previous input matching a regular expression to the prompt.
`M-s`	Bring the next input matching a regular expression to the prompt.

E EMaxima mode commands

Key	Description
`C-c C-o`	Create a (standard) cell.
`C-c C-p`	Create a session cell.
`C-c C-n`	Create a noshow cell.
`C-c +`	Go the the next cell.
`C-c -`	Go to the previous cell.
`C-c C-u a`	Update all of the cells. With an argument, don't ask before updating.
`C-c C-u A`	Update all of the cells in TEX form. With an argument don't ask before updating.
`C-c C-u t`	Evaluate all of the initialization cells.
`C-c C-u i`	Update all of the initialization cells. With an argument, don't ask before updating.
`C-c C-u I`	Update all of the initialization cells in TEX form. With an argument, don't ask before updating.
`C-c C-u s`	Update all of the session cells in TEX form. With an argument, don't ask before updating.

Commands only available in cells.

Key	Description
`C-c C-v`	Send the current cell to the *Maxima* process.
`C-c C-u c`	Update the current cell.
`C-c C-u C`	Update the current cell in TEX form.
`C-c C-d`	Delete the output from the current cell.
`C-c C-t`	Toggle whether or not the current cell is an initialization cell.
`C-c C-x`	Insert a heading for the cell indicating that it's part of a package.
`C-c @`	Assemble the references contained in the cell. With an argument, assemble the package that the cell defines.
`C-c C-TAB`	Complete a reference within a cell.

Commands only available outside of cells.

Key	Description
`C-c C-u l`	Send the current line to *Maxima*, and replace the line with the *Maxima* output.
`C-c C-u L`	Send the current line to *Maxima*, and replace the line with the *Maxima* output in TEX form.

F **Maxima** mode commands

Motion

Key	Description
`M-C-a`	Go to the beginning of the form.
`M-C-e`	Go to the end of the form.
`M-C-b`	Go to the beginning of the sexp.
`M-C-f`	Go to the end of the sexp.

Process

Key	Description
`C-cC-p`	Start a *Maxima* process.
`C-cC-r`	Send the region to the *Maxima* process.
`C-cC-b`	Send the buffer to the *Maxima* process.
`C-cC-c`	Send the line to the *Maxima* process.
`C-cC-e`	Send the form to the *Maxima* process.
`C-cC-k`	Kill the *Maxima* process.
`C-cC-p`	Display the *Maxima* buffer.

Completion

Key	Description
`M-TAB`	Complete the *Maxima* symbol.

Comments

Key	Description
`C-c ;`	Comment the region.
`C-c :`	Uncomment the region.
`M-;`	Insert a short comment.
`C-c *`	Insert a comment environment.

Indentation

Key	Description
`TAB`	Indent line.
`M-C-q`	Indent form.

Maxima help

Key	Description
`C-c C-d`	Get help on a (prompted for) subject.
`C-c C-m`	Read the manual.
`C-cC-h`	Get help with the symbol under point.
`C-cC-a`	Get apropos with the symbol under point.

Miscellaneous

Key	Description
`M-h`	Mark the form.
`C-c)`	Check the region for balanced parentheses.
`C-c C-)`	Check the form for balanced parentheses.

G AUCTEX commands

Inserting commands

Key	Description
C-c C-e	Insert an environment.
C-c C-s	Insert a section.
C-c]	Close an environment.
C-c C-j	Insert an item into a list.
"	Smart quote.
$	Smart dollar sign.
C-c @	Insert double brace.
C-c C-m	Insert TEX macro.
M-TAB	Complete TEX macro.

Formatting

Key	Description
C-c C-q C-r	Format region.
C-c C-q C-s	Format section.
C-c C-q C-e	Format environment.
C-c .	Mark an environment.
C-c *	Mark a section.

Commenting

Key	Description
C-c ;	Comment a region.
C-u C-c ;	Uncomment a region.
C-c %	Comment a paragraph.
C-u C-c %	Uncomment a paragraph.

Font selection

Key	Description
C-c C-f C-b	Bold.
C-c C-f C-i	Italics.
C-c C-f C-r	Roman.
C-c C-f C-e	Emphasized.
C-c C-f C-t	Typewriter.
C-c C-f C-s	Slanted.
C-c C-f C-d	Delete font.
C-u C-c C-f	Change font.

Running TEX

(Commands: `TeX`, `TeX Interactive`, `LaTeX`, `LaTeX Interactive`, `SliTeX`, `View`, `Print`, `BibTeX`, `Index`, `Check`, `File`, `Spell`.)

Key	Description
`C-c C-c`	Run a command on the master file.
`C-c C-r`	Run a command on the current region.
`C-c C-b`	Run a command on the buffer.
`` C-c ` ``	Go to the next error.
`C-c C-k`	Kill the TEX process.
`C-c C-l`	Center the output buffer.
`C-c C-^`	Switch to the master file.
`C-c C-w`	Toggle debug of overful boxes.