Linear Algebra with Maxima

Minh Van Nguyen
`nguyenminh2@gmail.com`

1 Is Maxima a car?

No, we're not talking about *that* kind of Maxima from Nissan. Instead, we'll concentrate on a computer program that can help us study mathematics. Maxima [1] is a general purpose computer algebra system (CAS) for symbolic manipulation. It is available free of charge and can be downloaded at

`http://maxima.sourceforge.net`

Maxima can be used to perform routine mathematics, symbolic and numerical computations, differentiate and integrate, solve differential equations, plotting graphs of functions, and a wide array of advanced mathematics. Students of mathematics can benefit in various ways from using Maxima. Maxima allows students to check their answers to exercises against that of a computer program. However, one important advantage of Maxima, or indeed any general purpose CAS, is that students can delegate routine mathematics to this program and devote more time to problem solving.

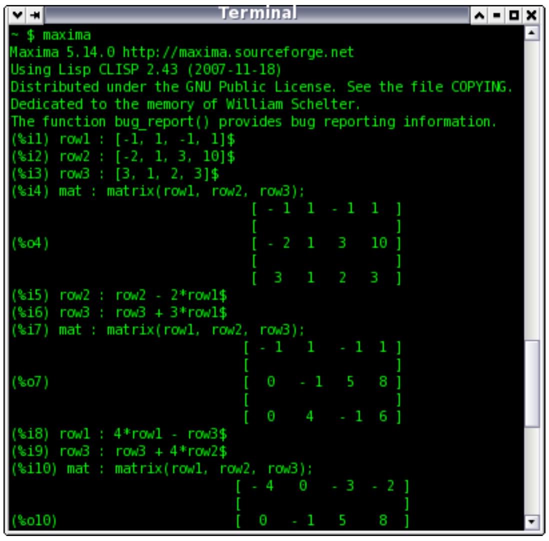
This article presents a brief introduction to Maxima. Due to limited space, we won't introduce all features of Maxima that are relevant to undergraduate mathematics. Instead, our primary aim is to present a short tutorial on using Maxima for solving problems in elementary linear algebra.

2 How sexy is Maxima?

Maxima is available under Windows, Linux, Mac and various other exotic operating systems. It is usually run within a command line window, with little or no graphical interface. Figure 1 shows a command line based session under Linux.

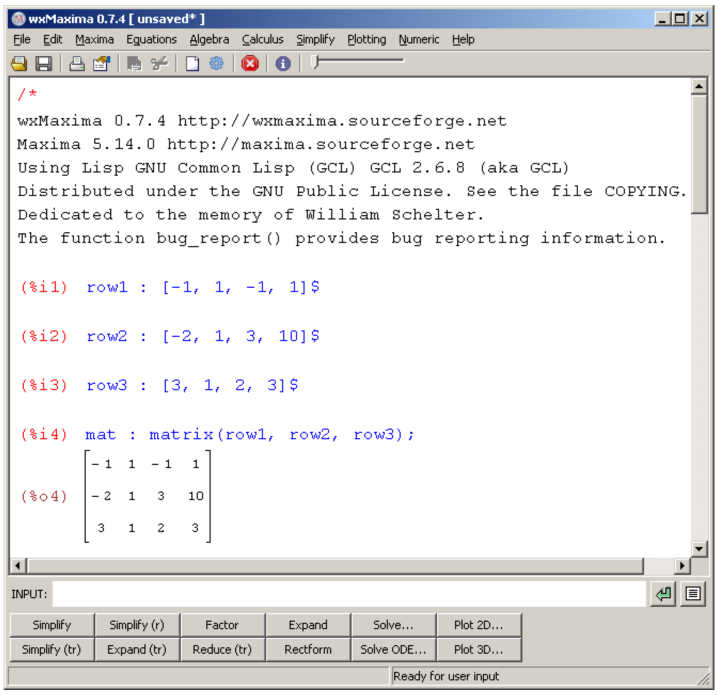
OK, let's admit it: the matrix in Figure 1 is rather difficult to read and there's little fancy graphics to make Maxima easier to use. But don't be discouraged until you've heard the full story. There are a number of graphical user interfaces (GUI) under Linux, but TeXmacs [2] is known to produce LaTeX quality output. Under Windows, we can use a GUI called wxMaxima [3] to make the output from Maxima easier to read. Figure 2 shows a session using wxMaxima under Windows.

Published in the University of Western Australia (UWA) Maths Society student magazine *AfterMath*, issue 4, 2008, pp.3–7.
`http://www.maths.uwa.edu.au/students/undergraduate/MS/AfterMath`



```
~ $ maxima
Maxima 5.14.0 http://maxima.sourceforge.net
Using Lisp CLISP 2.43 (2007-11-18)
Distributed under the GNU Public License. See the file COPYING.
Dedicated to the memory of William Schelter.
The function bug_report() provides bug reporting information.
(%i1) row1 : [-1, 1, -1, 1]$
(%i2) row2 : [-2, 1, 3, 10]$
(%i3) row3 : [3, 1, 2, 3]$
(%i4) mat : matrix(row1, row2, row3);
                              [ - 1  1  - 1  1  ]
                              [                 ]
(%o4)                         [ - 2  1   3   10 ]
                              [                 ]
                              [  3   1   2   3  ]
(%i5) row2 : row2 - 2*row1$
(%i6) row3 : row3 + 3*row1$
(%i7) mat : matrix(row1, row2, row3);
                             [ - 1   1   - 1  1 ]
                             [                  ]
(%o7)                        [  0   - 1   5   8 ]
                             [                  ]
                             [  0    4   - 1  6 ]
(%i8) row1 : 4*row1 - row3$
(%i9) row3 : row3 + 4*row2$
(%i10) mat : matrix(row1, row2, row3);
                            [ - 4   0   - 3  - 2 ]
                            [                    ]
(%o10)                      [  0   - 1   5    8  ]
```

Figure 1: Maxima running within a command line window under Linux.



```
/*
wxMaxima 0.7.4 http://wxmaxima.sourceforge.net
Maxima 5.14.0 http://maxima.sourceforge.net
Using Lisp GNU Common Lisp (GCL) GCL 2.6.8 (aka GCL)
Distributed under the GNU Public License. See the file COPYING.
Dedicated to the memory of William Schelter.
The function bug_report() provides bug reporting information.

(%i1) row1 : [-1, 1, -1, 1]$

(%i2) row2 : [-2, 1, 3, 10]$

(%i3) row3 : [3, 1, 2, 3]$

(%i4) mat : matrix(row1, row2, row3);
```

$$(\%o4)\quad \begin{bmatrix} -1 & 1 & -1 & 1 \\ -2 & 1 & 3 & 10 \\ 3 & 1 & 2 & 3 \end{bmatrix}$$

Figure 2: A graphical user interface to Maxima under Windows.

3 How can I get started?

Introductory courses in linear algebra begin by showing us how to solve systems of linear equations. Small systems such as (1) can easily be solved by hand, but such method is not suitable for larger systems. In this section, we'll present various ways of using Maxima to solve (1). The techniques presented here can be applied to larger systems.

$$\begin{array}{rcrcrcr} -x & + & y & - & z & = & 1 \\ -2x & + & y & + & 3z & = & 10 \\ 3x & + & y & + & 2z & = & 3 \end{array} \tag{1}$$

3.1 Can I use row-echelon form?

Let's start off by using the method of row-echelon form to solve (1). We put the coefficients of each row, including the term on the right-hand side of the equal sign, into a vector:

```
row1 : [-1, 1, -1, 1]$
row2 : [-2, 1, 3, 10]$
row3 : [3, 1, 2, 3]$
```

Now stack each row vector on top of each other to get the augmented matrix for (1). For this, we use the command

```
mat : matrix(row1, row2, row3);
```

which produces the following output:

```
[ - 1  1  - 1  1  ]
[                 ]
[ - 2  1   3   10 ]
[                 ]
[  3   1   2   3  ]
```

To get a zero in the first position of the second row, we subtract twice row one from row two and store the result in row two. Similarly, to get a zero in the first position of the third row, we add three times row one to row three and store the result in row three. In Maxima, these can be accomplished as follows:

```
row2 : row2 - 2*row1$
row3 : row3 + 3*row1$
```

Now use the command

```
mat : matrix(row1, row2, row3);
```

to view our newly updated matrix, which is:

```
[ - 1   1   - 1  1 ]
[                  ]
[  0   - 1   5   8 ]
[                  ]
[  0    4   - 1  6 ]
```

Carrying this process through to the end, we end up with the following reduced row-echelon form:

```
[ 1  0  0  - 1 ]
[              ]
[ 0  1  0   2  ]
[              ]
[ 0  0  1   2  ]
```

The set of solution to (1) can be read off from the right-most column in this matrix, i.e. $x = -1,\ y = 2,\ z = 2$.

3.2 Is there a quicker way?

The previous section mimics the process whereby we would solve (1) by hand. As we work through a system by hand, we can use Maxima to check on our progress, to verify our workings and our intermediate results.

However, we can also use the Maxima command `linsolve` to verify our answer. This command can be used to solve systems of linear equations. Here are the commands to solve system (1) using `linsolve`:

```
eq1 : -x + y - z = 1$
eq2 : -2*x + y + 3*z = 10$
eq3 : 3*x + y + 2*z = 3$
linsolve([eq1, eq2, eq3], [x,y,z]);
```

4 Where can I find out more about Maxima?

A good place to start is the following website:

`http://maxima.sourceforge.net`

It has many links to tutorials and manuals for using Maxima. Tutorials cater to Maxima novices as well as advanced users. The manual is an excellent source of information on using Maxima, but be warned that it can be difficult for beginners. You can subscribe to the Maxima mailing list at

`http://maxima.sourceforge.net/maximalist.html`

You can post support questions about using Maxima to this list and people would be happy to help you getting the most out of Maxima.

References

[1] Maxima computer algebra system
`http://maxima.sourceforge.net/`
Viewed 2008-03-24

[2] GNU TeXmacs
`http://www.texmacs.org/`
Viewed 2008-03-24

[3] wxMaxima
`http://wxmaxima.sourceforge.net/wiki/index.php/Main_Page`
Viewed 2008-03-24