DonnishJournals 2041-1189

Donnish Journal of Educational Research and Reviews.
Vol 2(1) pp. 064-070 January, 2015.
http://www.donnishjournals.org/djerr

Original Research Paper

Plotting with MAXIMA

Niyazi Ari and Savaş Tuylu*

Computer Science, Nigerian Turkish Nile University, Nigeria

Accepted 15th January, 2015.

The methods of calculus lie at the heart of the physical sciences and engineering. Maxima can help you make faster progress, if you are just learning calculus. The examples in this research paper will offer an opportunity to see some Maxima tools in the context of simple examples, but you will likely be thinking about much harder problems you want to solve as you see these tools used here. This research paper includes plotting with MAXIMA.

Keywords: Plotting Functions, Maxima.

INTRODUCTION

Maxima is a system for the manipulation of symbolic and numerical expressions, including differentiation, integration, Taylor series, Laplace transforms, ordinary differential equations, systems of linear equations, polynomials, sets, lists, vectors, matrices, tensors, and more. Maxima yields high precision numeric results by using exact fractions, arbitrary precision integers, and variable precision floating point numbers. Maxima can plot functions and data in two and three dimensions.

Maxima source code can be compiled on many computer operating systems, including Windows, Linux, and MacOS X. The source code for all systems and precompiled binaries for Windows and Linux are available at the SourceForge file manager.

Maxima is a descendant of Macsyma, the legendary computer algebra system developed in the late 1960s at the Massachusetts Institute of Technology. It is the only system based on that effort still publicly available and with an active user community, thanks to its open source nature. Macsyma was revolutionary in its day, and many later systems, such as Maple and Mathematica, were inspired by it.

The Maxima branch of Macsyma was maintained by William Schelter from 1982 until he passed away in 2001. In 1998 he obtained permission to release the source code under the GNU General Public License (GPL). It was his efforts and skills that made the survival of Maxima possible. MAXIMA has been constantly updated and used by researcher and engineers as well as by students.

2. PLOTTING

2.1. Introduction

Maxima uses an external plotting package to make the plots. The plotting functions calculate a set of points and pass them to the plotting package together with a set of commands. Currently, the external plotting programs are Gnuplot and Xmaxima. Gnuplot is default for Windows applications. Xmaxima is a Tcl/Tk graphical interface for Maxima that can be used to display created when Maxima is run from the console or from other graphical interfaces.

Corresponding Author: savastyludf@hotmail.com

Table 2.1: The following table gives the Maxima functions for plotting:

Function	Description
contour_plot *(expr,x_range, y_range, options, …)*	It plots the contours (curves of equal value) of *expr* over the region *x_range* by *y_range*. Any additional arguments are treated the same as in plot3d.
plot2d *(plot,x_range, …, options,…)* **plot2d** *([plot_1, …,plot_n], …, options,…)* **plot2d** *([plot_1, …,plot_n], x_range, …, options, …)*	Where *plot*, *plot_1*, …, *plot_n* can be either expressions, function names or a list with the any of the forms: [discrete, [*x1*, ..., *xn*], [*y1*, ..., *yn*]], [discrete, [[*x1*, *y1*], ..., [*xn*, ..., *yn*]] or [parametric, *x_expr*, *y_expr*, *t_range*]. Displays a plot of one or more expressions as a function of one variable or parameter. plot2d displays one or several plots in two dimensions. When expressions or function names are used to define the plots, they should all depend on only one variable *var* and the use of *x_range* will be mandatory, to provide the name of the variable and its minimum and maximum values; the syntax for *x_range* is: [*variable*, *min*, *max*]. A plot can also be defined in the discrete or parametric forms. The discrete form is used to plot a set of points with given coordinates. A discrete plot is defined by a list starting with the keyword *discrete*, followed by one or two lists of values. If two lists are given, they must have the same length; the first list will be interpreted as the x coordinates of the points to be plotted and the second list as the y coordinates. If only one list is given after the *discrete* keyword, each element on the list should also be a list with two values that correspond to the x and y coordinates of a point. A parametric plot is defined by a list starting with the keyword *parametric*, followed by two expressions or function names and a range for the parameter. The range for the parameter must be a list with the name of the parameter followed by its minimum and maximum values: [*param*, *min*, *max*]. The plot will show the path traced out by the point with coordinates given by the two expressions or functions, as *param* increases from *min* to *max*. A range for the vertical axis is an optional argument with the form: [y, *min*, *max*] (the keyword *y* is always used for the vertical axis). If that option is used, the plot will show that exact vertical range, independently of the values reached by the plot. If the vertical range is not specified, it will be set up according to the minimum and maximum values of the second coordinate of the plot points.
plot3d *(expr,x_range, y_range, …,options, …)* **plot3d** *([expr_1, …,expr_n], x_range,y_range, …, options,…)*	Displays a plot of one or more surfaces defined as functions of two variables or in parametric form. The functions to be plotted may be specified as expressions or function names. The mouse can be used to rotate the plot looking at the surface from different sides.
psfile *[psfile, string]*	Saves the plot into a Postscript file with name equal to *string*, rather than showing it in the screen. By default, the file will be created in the directorydefined by the variable *maxima_tempdir*; the value of that variable can be changed to save the file in a different directory.

2.2. Examples

1.
```
(%i1) eq:x^2-y^2;
      contour_plot(eq,[x,-5,5],[y,-5,5])$
```
(%o1) x^2-y^2

2.
```
(%i3) eq:exp(-x)*sin(x);
      plot2d(eq,[x,0,5])$
```
(%o3) $\%e^{-x}\sin(x)$

3.
```
 -->  A plot with a logarithmic scale:
(%i5) eq:exp(-x);
      plot2d(eq,[x,-5,5],[logy])$
```
(%o5) $\%e^{-x}$

4.
```
 -->  Parametric plot:
(%i7) r:(exp(cos(t))-sin(3*t))$
      plot2d([parametric,r*sin(t),r*cos(t),
       [t,-5*%pi,5*%pi],[nticks,1000]])$
```

5.
```
 -->  Discrete plot
(%i9) plot2d([discrete,[[1,.1],[2,0.2],
 [3,0.3],[4,0.4],[5,0.5]]],[style,points])$
```

6.
```
 -->  Two fuctions in the same plot:
(%i10) eq1:exp(-x)*sin(x)$
       eq2:exp(-x)*sin(2*x)$
       plot2d([eq1,eq2],[x,-5,5])$
```

7.
```
(%i13) eq:x^2-y^2$
       plot3d(eq,[x,-5,5],[y,-5,5])$
```

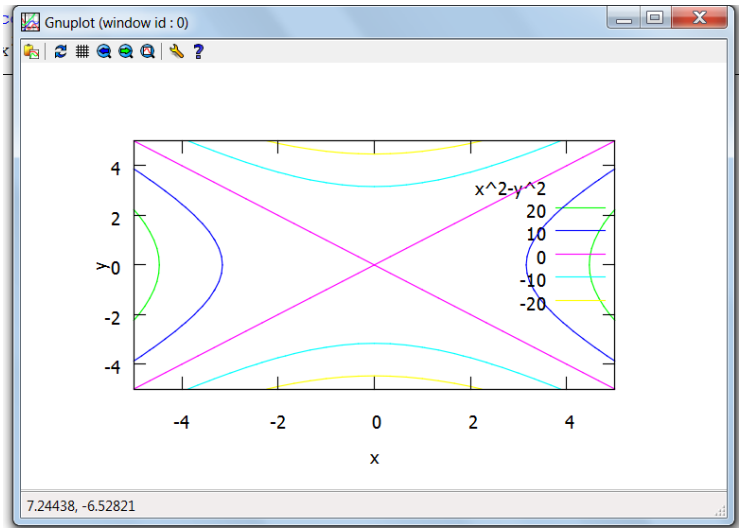

Figure 1. How to use the function "contour_plot"

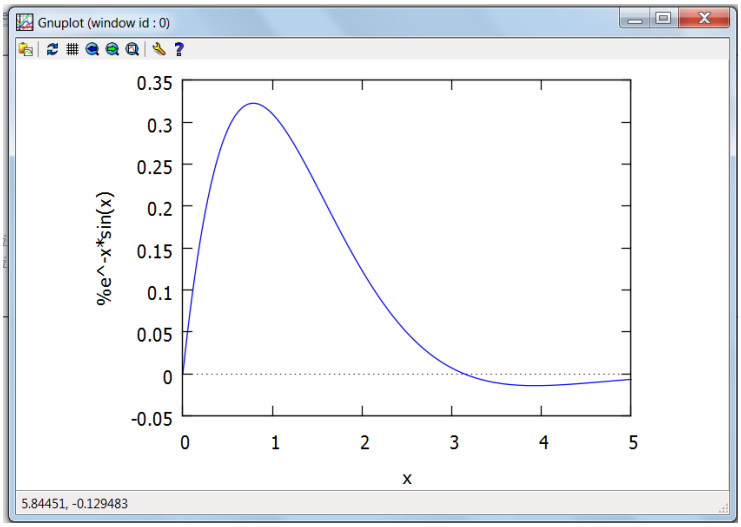

Figure 2. How to use the function "plot2d"

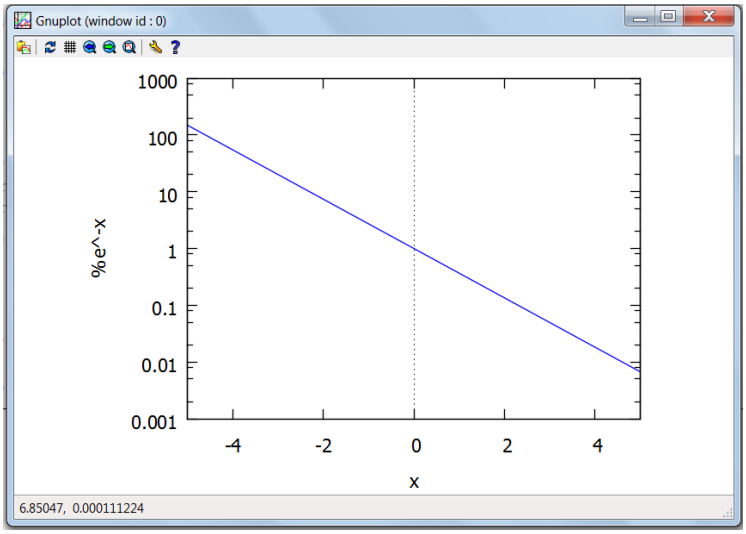

Figure 3. How to use logarithmic scale with the function "plot2d"

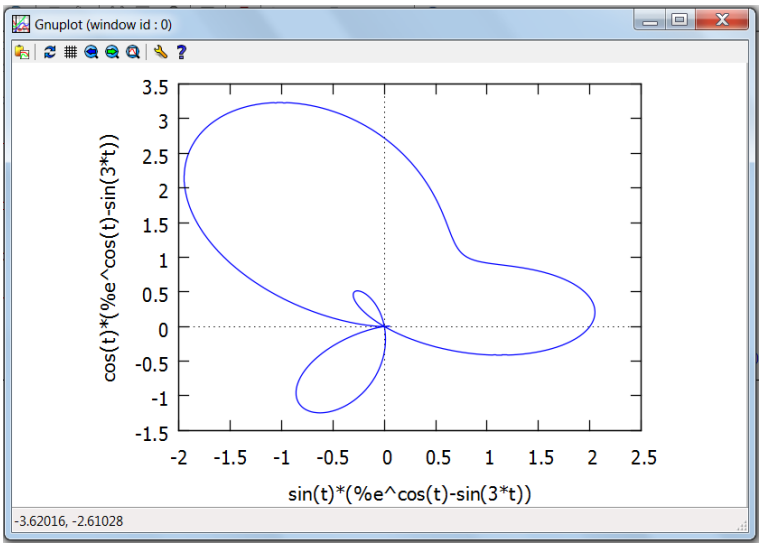

Figure 4. How to use parametric plot with the function "plot2d"

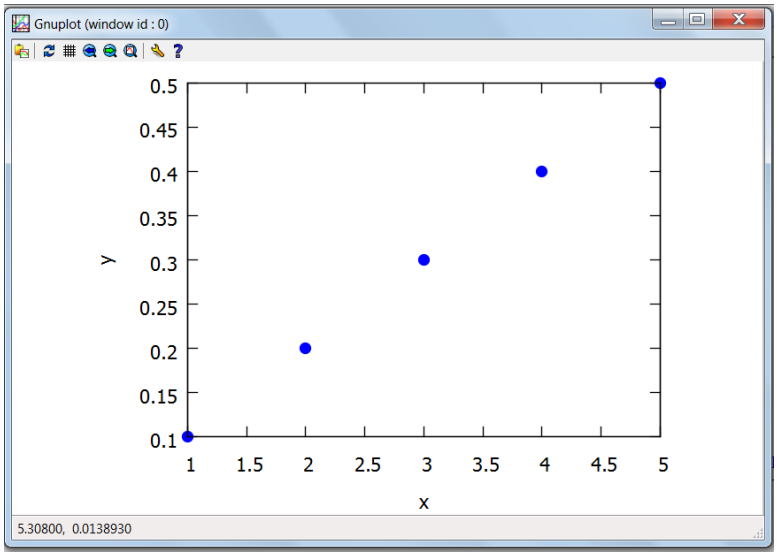

Figure 5. How to use discrete plot with the function "plot2d"

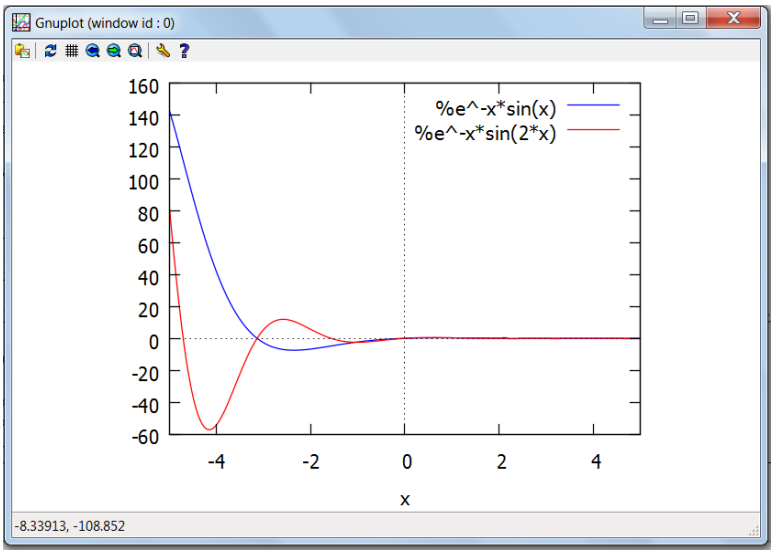

Figure 6. How to use two functions in the same plot with the function "plot2d"

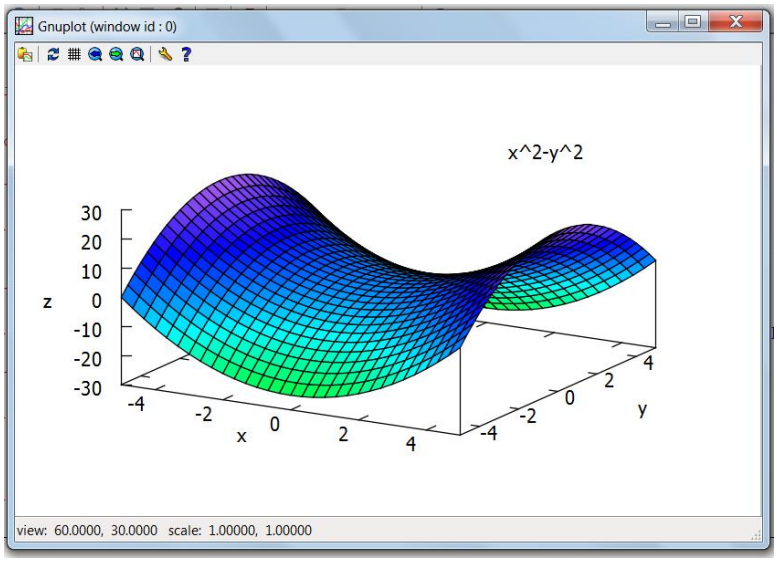

Figure 7. How to use the function "plot3d"

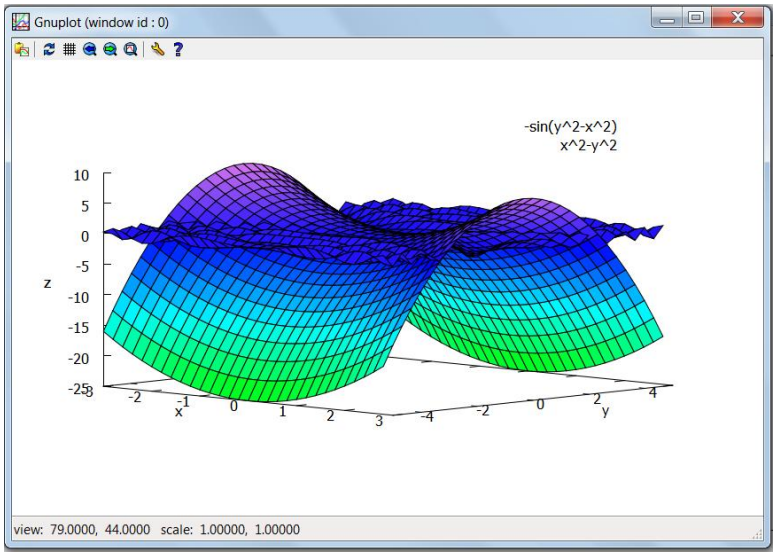

Figure 8. How to use two surfaces in the same plot

8.

```
  -->  Two surfaces in the vsame plot:
(%i15) eq1:(x^2-y^2);
       eq2:sin(x^2-y^2);
       plot3d([eq1,eq2,[x,-3,3],[y,-5,5]]);
(%o15) x^2-y^2
(%o16) -sin(y^2-x^2)
(%o17) [C:/Users/user/maxout.gnuplot]
```

9.

```
  -->  3 surfaces in the same plot:
(%i18) eq1:cos(x)*sin(y);
       eq2:cos(x/2)*sin(y/2);
       eq3:cos(x/4)*sin(y/4);
       plot3d([eq1,eq2,eq3],[x,-2*%pi,
       2*%pi],[y,-2*%pi,2*%pi],[grid,40,40]);
(%o18) cos(x) sin(y)
(%o19) cos(x/2) sin(y/2)
(%o20) cos(x/4) sin(y/4)
(%o21) [C:/Users/user/maxout.gnuplot]
```

10

```
(%i22) eq:sin(3*theta)*cos(phi);
       plot3d(eq,[theta,0,%pi],[phi,0,2*%pi],
       [transform_xy,spherical_to_xyz],[grid,30,60])$
(%o22) cos(φ) sin(3 θ)
```

11

```
(%i24) load(draw)$
       draw3d(
       colorbox="Magnitude",
       enhanced3d=true,
       explicit(cos(x^2-y^3),x,-5,5,y,-5,5))$
```

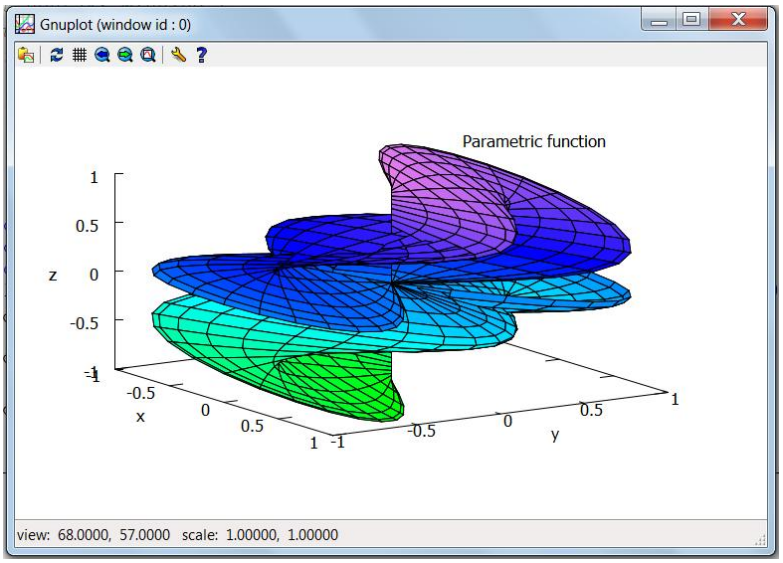

Figure 9. How to use three surfaces in the same plot

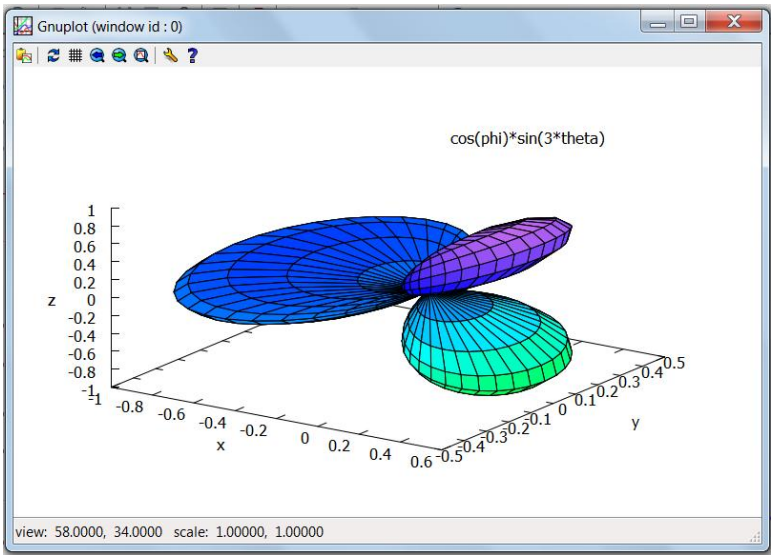

Figure 10. How to use three surfaces in the same plot

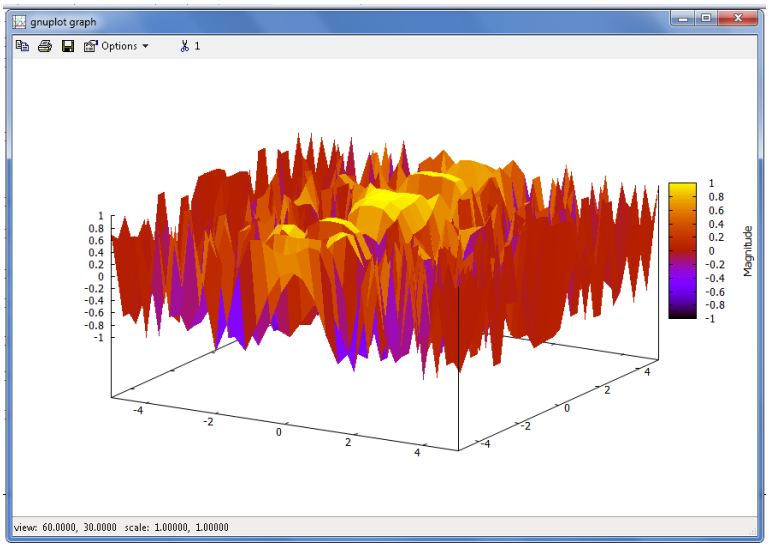

Figure 11. How to use draw package and explicit for plotting

3. CONCLUSION

The research paper can apply each and every part of plotting, help application of the physical sciences and engineering, make faster progress, and help to understand plotting faster. The paper particularity helps to understand parts of Calculus and is going to extend to other parts of the Calculus.

4. ACKNOWLEDGEMENT

We would like to thank the Maxima developers and Nigerian Turkish Nile University (NTNU) for their friendly help.

REFERENCES

[1] R. H. Rand, Introduction to Maxima,
[2] R. Dodier, Minimal Maxima, 2005
[3] https://www.ma.utexas.edu/maxima/maxima_19.html.
[4] N. Ari, G. Apaydin, Symbolic Computation Techniques for Electromagnetics with MAXIMA and MAPLE, Lambert Academic Publishing, 2009.
[5] Fred Szabo. Linear Algebra, An Introduction Using MATHEMATICA Academic Press 2000
[6] S. Lipschutz, M. Lipson, Linear Algebra Schaum's Outlines McGraw Hill2009
[7] S. Lipschutz, Linear Algebra Schaum's Solved Problems Series McGraw Hill 1989
[8] Minh Van Nguyen, Linear Algebra with MAXIMA http://www.maths.uwa.edu.au/students/undergraduate/MS/After Math
[9] N. ARI, Engineering Mathematics with MAXIMA SDU, Almaty, 2012
[10] Niyazi ARI, Lecture notes, University of Technology, Zurich, Switzerland.
[11] Niyazi ARI, Symbolic computation of electromagnetics with Maxima (2013).
[12] http://maxima.sourceforge.net/