Advances in Science and Technology Research Journal
Volume 8, No. 24, Dec. 2014, pages 44–50
DOI: 10.12913/22998624/566

Original Article

APPLICATION OF THE MAXIMA IN THE TEACHING OF SCIENCE SUBJECTS AT DIFFERENT LEVELS OF KNOWLEDGE

Anna Makarewicz[1]

[1] Department of Applied Mathematics, Lublin University of Technology, Nadbystrzycka 38, 20-618 Lublin, Poland, e-mail: a.makarewicz@pollub.pl

Received: 2014.10.18
Accepted: 2014.11.12
Published: 2014.12.01

ABSTRACT
In this paper I present a program Maxima, which is one of the best computer algebra system (CAS). The program is easy to use and offers many possibilities. In the text the interface, basic commands, and various examples to facilitate complex calculations in almost every field of mathematics are preented. I prove that the program can be used in secondary schools, upper secondary schools and college. Program's capabilities are presented, including, among others, differentiation and symbolic integration, symbolic solving equations (including differential), simplifying algebraic expressions, matrix operations, the possibility of drawing graphs of 2D / 3D and others.

Keywords: computer program, numerical computation, symbolic computation, symbolic differentiation, symbolic integration, graph, graph three-dimensional, function.

INTRODUCTION

Maxima is a computer program, performing mathematical calculations, both symbolic and numeric. The use of symbolic computation allows to solve many mathematical problems in an accurate manner. Maxima can work as a calculator, which can be used in secondary schools. It is able to create very complex parametric function graphs, implicit function graphs and graphs of function of several variables, (with such problems faced students of almost all departments), these are necessary basics that form the base for further development and specialization in a specific area, for example mechanics, electrical engineering, computer science, construction, aviation, broadly understood technologies, etc.

This program stems from program macsyma developed at the Massachusetts Institute of Technology for the Department of Energy. The software features include:

- performing numerical calculations with any accuracy,
- simplifying algebraic expressions and trigonometric,
- symbolic solving equations (including differential),
- symbolic solving systems of equations,
- symbolic differentiation and integration,
- matrix operations,
- drawing graphs of functions,
- perform calculations in the field of probability theory,
- define your own functions by the user,
- export of the results to the following formats: HTML, LaTeX, and graphic formats PNG, PostScript.

Thanks to William Schelter program is free. Since then, the Maxima has been developing quite intensively in versions for Windows and Linux. Since version 5.101 program already contains an integrated overlay wxMaxima (graphical user interface), which provides better formatting mathematical formulas and makes them easier to enter and modify. Maxima is a simple and intuitive software. Learning the basics of the Maxima does not require a large amount of time. Maxima facilitates better and faster understanding of mathematics. It allows you to experiment, consider a lot of alternative solutions and create visualization. It also helps to save time and to control and eliminate errors in solving complex accounting tasks.

BASICS MAXIMA

Maxima works at input %i and outputs %o. Inputs and outputs are numbered that in combination with the %i or %o creates a unique identifier. Maxima distinguishes quantity of letter, so the function f(x) and F(x) are not the same. Variable %i stores the result of the last command.

Each instruction must be completed: a semicolon (;) – displays the „result" or dollar ($) – do not display the result (but it is calculated). The program accepts standard arithmetic operators that are intuitive, so that the program can be used as a better class of the calculator in secondary schools:

- adding +
- subtraction –
- multiplication *
- dividing /
- exponentiation ^ or **
- matrix multiplication .

Comparison operators are also intuitive:

- equality =
- inequality <>
- greater than >
- less than <
- greater than or equal to >=
- less than or equal to <=
- to assign a value :
- defining a function :=

Constants:

- %e – base of natural logarithm
- %i – the imaginary unit
- %pi – π
- inf – ∞
- minf – $-\infty$
- %gamma – Euler's constant

Example:

Let's try to perform the first arithmetic operations (ie, Maxima will be used as a calculator) and show the interface (Figure 1).

Note that you can refer to the previous result with % (percent), eg:

(%i4) % – 20; (33–20)

(%o4) 13;

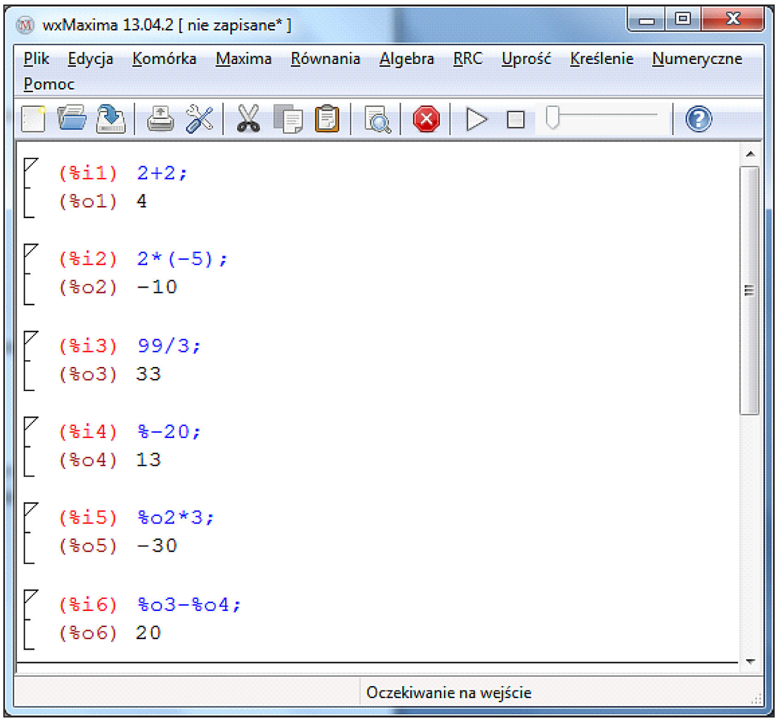
You can also refer to any previous result, which we received in a given session, which greatly facilitates the work and calculations. For this purpose we use the term %o or %i. Let us see how this works in practice:

(%i5) %o2 * 3; (-10*3)

(%o5) -30

(%i6) %o3 – %o4; (33–13)

(%o6) 20



```
(%i1) 2+2;
(%o1) 4

(%i2) 2*(-5);
(%o2) -10

(%i3) 99/3;
(%o3) 33

(%i4) %-20;
(%o4) 13

(%i5) %o2*3;
(%o5) -30

(%i6) %o3-%o4;
(%o6) 20
```

Oczekiwanie na wejście

Fig. 1. First arithmetic operations

To receive a number in decimal form, use the function float, here is an example:

```
(%i1) float(2/7);
(%o1) 0.28571428571429
```

Numerical calculations in Maxima can be performed with any precision, the command fpprec: n will be performed with an accuracy of n-significant digits (standard setting fpprec: 16). We can also change the accuracy of the displayed results by typing fpprintprec: n, here is an example:

```
(%i1) float(23/112);
(%o1) 0.20535714285714
(%i2) fpprintprec:4$
(%i3) float(23/1121);
(%o3) 0.021
```

Maxima handles well with large numbers and is better than the calculator. Let us try to calculate the 2^{128} and 30! Factorial operation is very difficult to imagine, especially for high school students and ordinary calculators cannot handle such large numbers, Maxima by calculating "the factorial of any value" helps you to realize how much the number is changing under its influence, which greatly helps to develop student's imagination. To do this in Maxima, we write:

```
(%i1) 2 ^ 128;
(%o1) 340282366920938463463374607431768
211456
(%i2) 30!;
(%o2) 265252859812191058636308480000000
```

Operator ' (apostrophe)

Operator ' prohibits the calculation of the value of its argument.

Example:

```
 (%i1) diff(sin(x),x)
(%o1) cos(x)
(%i2) 'diff(sin(x),x);
(%o2) (sin(x))
```

Symbolic calculations are very good and useful, as during subsequent operations the computer does not make the mistake associated with an approximation of the number. Nested mathematical operations are often so complicated that it is impossible to estimate it's value. Then you can ask the program to give the approximate result of the calculation. To get the value of the expression we place the comma after it and call the command number. It requires the return of the approximate value of the expression which is located before the comma. So far, Maxima was used as a better class of calculator. Now we will present what is beyond the reach of a calculator.

EXPRESSIONS

The program Maxima is possible to declare algebraic expressions. You can save even the most complex expression-patterns. These expressions can be freely converted, and you can find their values at certain, fixed parameter values. Maxima has many features that allow modifying, simplifying and developing expressions and these are the main problems of students from secondary schools. Thanks to Maxima, they may check their thinking because Maxima allows you to control the partial results and calculations and all the results are displayed in symbolic form which enables precise analysis of reasoning. Let us present selected functions on algebraic expressions:

- ev(expression,condition) – modifies the expression based on a condition,
- subst(b,a,expression) – after the command variable a in the expression will be replaced by variable b,
- expand(expression) – develops algebraic expressions,
- ratsimp(expression) – simplifies algebraic expressions,
- radcan(expression) – simplifies expression,
- factor(expression) – decomposes expression factors,
- trigexpand(expression) – develops trigonometric expression,
- trigreduce(expression), trigsimp(expression) – simplifies trigonometric expression
- rectform – shows the expression in the form A + B i.
- polarform – shows the complex expression in the polar form re $^{i\theta}$
- logcontact – simplifies the expression such as log x + b log y.

Example:

(%i1) expand((b+c)^5); // expands the expression in brackets

(%o1) c^5+5*b*c^4+10*b^2*c^3+10*b^3*c^2+5*b^4*c+b^5

Now we will factorize the expression by enter the command:

(%i2) factor(%);

(%o2) $(c+b)^5$

DIFFERENTIATION AND INTEGRATION

The most important command to derive is diff (f (x), x); it requires the variable as the second argument at which the function f(x) is differentiated and can be used to calculate the partial derivatives. Diff command can have a third argument, defining the degree of derivative. These are selected features of the Maxima for the differential and integral calculus:

$f^{(n)}(x)$ – diff(f(x),x,n)

$\int f(x)dx$ – integrate(f(x),x)

$\int_a^b f(x)dx$ – integrate(f(x),x,a,b) or romberg(f(x), x,a,b) (calculate the integral numerically)

$\frac{\partial f}{\partial x}$ – diff(f(x,y),x)

$\frac{\partial f}{\partial y}$ – diff(f(x,y),y)

$\frac{\partial^2 f}{\partial x^2}$ – diff(f(x,y),x,2)

$\frac{\partial^2 f}{\partial y^2}$ – diff(f(x,y),y,2)

$\frac{\partial^{k+n} f}{\partial x^k \partial y^n}$ – diff(f(x,y),x,k,y,n)

Examples:

1. Let f(x) = x^x be the function whose derivative we try to calculate, then we obtain:
   (%i1) diff(x^x,x);
   (%o1) x^x*(log(x)+1)
2. Let the function be given by f(x) = $\frac{\sin(x)}{x}$. Calculation derivative of the second degree in Maxima looks as follows:
   (%i1) diff(sin(x)/x,x,2);
   (%o1) $-\frac{\sin(x)}{x}+\frac{2\sin(x)}{x^3}-\frac{2\cos(x)}{x^2}$
3. The function be given by f(x, y) = $x*y+\frac{x^2}{y^3}$. Let calculate its partial derivative:
   (%i1) diff((x*y)+(x^2)/(y^3),x, 1, y, 1);
   (%o1) $1-\frac{6x}{y^4}$

To calculate the integrals both definite and indefinite we use integrate() command.

Example 1. We calculate $\int \frac{7-2x}{x^2+4x+13}dx$

(%i1) integrate((7-2*x)/(x^2+4*x+13),x);

(%o1) $\frac{11}{3}\operatorname{atan}\left(\frac{2x+4}{6}\right)-\log(x^2+4x+13)$

Each student can easily check the result by differentiating it and writing the following commands:

(%i2) diff(%,x);

(%o2) $\frac{11}{9*\left(\frac{(2x+4)^2}{36}+1\right)}-\frac{2x+4}{x^2+4x+13}$

(%i3) ratsimp(%); // by this command we get a simplified result

(%o3) $-\frac{2x-7}{x^2+4x+13}$

Example 2. Let calculate the definite integral $\int_0^\infty e^{-x^2}dx$ by writing command integrate(expression, variable, lower limit, upper limit):

(%i1) integrate(exp(-x*x),x,0,inf);

(%o1) $\frac{\sqrt{\pi}}{2}$

Expressions can be calculated numerically by using the float(expression) command, continuing the previous example, we get:

(%i2) float(%);

(%o2) 0.88622692545276

The function that performs the integration in a numerical way is romberg. The Romberg () and integrate () handles well with multiple integrals (not only with the constant limits), which will surely be useful for students during calculating fields, the volumes of solids and their surfaces. In addition, anyone can do two or three-dimensional graphs, (we will explain it in chapter 9), this in combination with symbolic calculations made in Maxim actively growing and developing spatial imagination. We calculate: $\int_1^2\int_4^6 \frac{x}{y^2}dydx$

(%i1) g(x,y):=x/(y^2);

(%i2) romberg(romberg(g(x,y),y,4,6),x,1,2);

(%o2) 0.12500002874223

SYSTEMS OF EQUATIONS

Some useful functions for solving equations and systems of equations:

- solve (equation, variable) – solves the equation in the set of complex numbers,
- linsolve ([equations], [variables]) – solves a linear system of equations,
- algsys ([equations], [variables]) – solves algebraic system equations in the set of complex numbers, if you want to receive solution in the real numbers , first you should write the following command: realonly: true,
- find_root (f (x), x, a, b) – this command find (through the approximation) zeros of the function f in the interval [a, b] (notice that f (a), f (b) should have different signs).

For example, to find a solution to the system of equations $\begin{cases}9x-8y=4\\7x+2y=3\end{cases}$ we write:

(%i1) r1: $9*x-8*y=4$;

(%o1) $9x-8y=4$

(%i2) r2: $7*x+2*y=3$;

(%o2) $7x+2y=3$

```
(%i3) s:solve([r1,r2], [x,y]);
(%o3) [[x=16/37, y=-1/74]]
```

Checking the solution

The following command substitute the result into to both equations. The subst() command put the expression in the first argument into the expression the second argument

```
 (%i4) [subst(s,r1),subst(s,r2)];
(%o4) [4=4,3=3]
```

Solutions do not have to be a real. For equation x^4-1 = 0, we obtain:

```
(%i1) solve(x^4-1);
(%o1) [x=%i,x=-1,x=-%i,x=1]
```

Functions realroots (W), realroots (W, accuracy) find by bisection method all real roots of a polynomial. Polynomial coefficients must be rational numbers. The second parameter defines the accuracy with which the root is searched- default is 10^{-7}. In addition, each student can calculate roots e.g. quadratic equation (with parameters a, b, c: $ax^2 + bx + c = 0$), and thus verify that the formulas learned in school are rely true, and they are zeros above quadratic function. Additionally, you can also check the correctness of Viete's formulas, which improve memorization and provide impetus for deepening self-knowledge.

DIFFERENTIAL EQUATIONS

The function that solves ordinary differential equations of the first and second order is ode2 (eqn, zvar, nvar), where zvar is the dependent variable and nvar is the independent variable. To solve the differential equation we will use the operator: 'diff – which introduces a symbol of the derivative and does not perform differentiation, but prepare derivative which later is part of the differential equation.

For example, we solve the differential equation $\frac{dy}{dt} = 3t^2y^2 - \sin(t)y^2$

(%i1) rr:'diff(y,t,1)=3*(t*y)^2-y^2*sin(t);

(%o1) $\frac{dy}{dt} = 3t^2y^2 - \sin(t)y^2$

(%i2) ro:ode2(rr,y,t); // solution of the differential equation

(%o2) 1/y=-cos(t)-t^3+%c //notice that %c is constant value

(%i3) rs:ic1(ro,t=0,y=1); // initial condition y(0)=1

(%o3) 1/y=-cos(t)-t^3+2

(%i4) method; // We can ask about the type of the equation and we get:

(%o4) separable // variable-separated

If Maxima cannot solve a differential equation, as a result returns false.

MATRICES

The Maxima has many functions related to matrices. Matrices are introduced to the program by using the function matrix, to perform the matrix product we use a dot (not *). The use of other features we show on an example:

If we use the function genmatrix of undefined parameter, we obtain a general matrix with elements a_{ij}.

(%i1) genmatrix(a,2,2);

(%o1) $\begin{bmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{bmatrix}$

We calculate the determinant by writing the following command:

(%i2) determinant(%);

(%o2) $a_{1,1}a_{2,2}$-$a_{1,2}a_{2,1}$

We can also (symbolically) designate the inverse matrix:

(%i3) invert(%o1) ; // Simplified result which we obtain by using the following function:

(%i4) ratsimp(%); //then we have:

(%o4) $\begin{bmatrix} \frac{a_{2,2}}{\mathrm{a1,1a2,2-a1,2a2,1}} & \frac{-a_{1,2}}{\mathrm{a1,1a2,2-a1,2a2,1}} \\ \frac{-a_{2,1}}{\mathrm{a1,1a2,2-a1,2a2,1}} & \frac{a_{1,1}}{\mathrm{a1,1a2,2-a1,2a2,1}} \end{bmatrix}$

These symbolic calculations also make remembering easier and preserve the fundamental rights that are used by the students in all science, they also inspire to continue learning mathematics.

Although the Maxima is a symbolic algebra system also allow us for numerical operations, also with elements of linear algebra. We can calculate, for example:

- the eigenvalues of the matrix A and their multiplicities – eigenvalues(A),
- rank of a matrix A – rank(A),
- eigenvalues of a square matrix A, with their multiples and their eigenvectors – eigenvectors(A),
- transposition of matrix A – transpose(A)

GRAPHS

At the beginning of drawing graphs we will present and talk over some examples of functions and their parameters. Let f1(x):= $\frac{(x^2-1)}{(x^2+1)}$

(%i1) f1(x):=(x^2-1)/(x^2+1)$
(%i2) plot2d(f1(x),[x,-8,8]); \\ Drawing the function y:=f1(x) in the range -8 <= x <=8 (Figure 2).

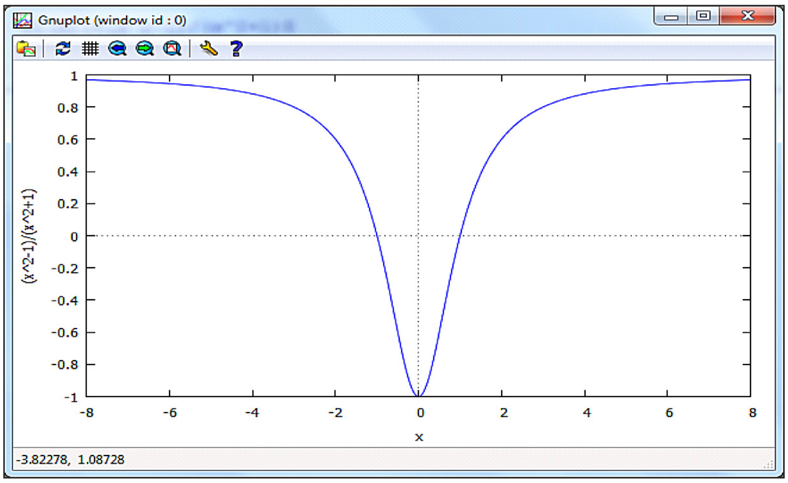
We can also do three-dimensional graphs in Maxima, here's an example (Figure 3):
Let fc(x,y):=2-x^2-y^2
(%i1) fc(x,y):=2-x^2-y^2$
(%i1) plot3d(fc(x,y), [x,-8,8], [y,-8,8]);

Thanks to 3D graph, students can see, sometimes even read (from graph) for example maximum and minimum value of functions in the range and extremums of function. The situation is very complicated, especially when function is a function of several variables, where, thanks to graph and visualize students can realy see the use of patterns and certain solving schemas in practice. 3D and 2D graphs also help in determining the limits of integration, not only single but also multiple integral. This allows the student to realize what are the limits of integration especially if they are infinite and illustrate the use of multiple integrals in practice. It is invaluable help for pupils and it proves that mathematics cannot learn "by heart".

To get the ability to rotate the graph (of course after running Maxima and write commands), we obtain:
(%i2) plot3d(fc(x,y), [x,-8,8], [y,-8,8], [plot_format, openmath]) (Figure 4).

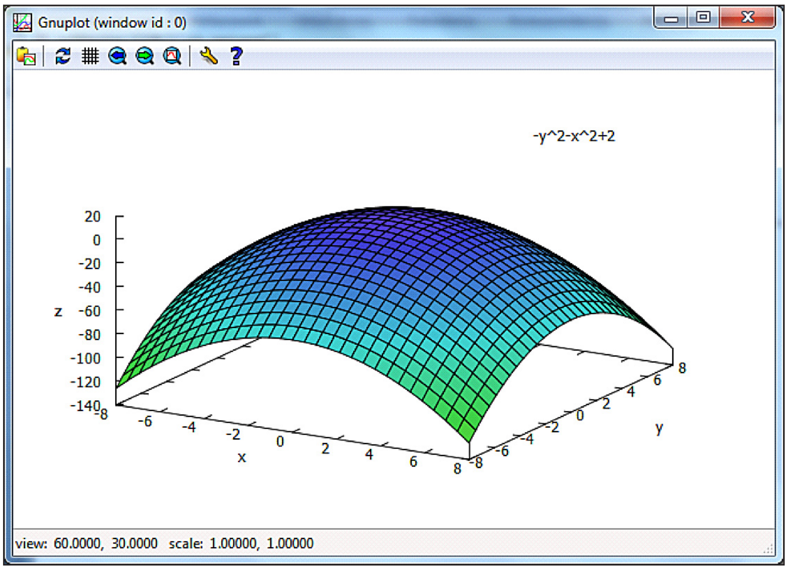
Draw.mac package through the command implicit () allows us to draw graphs of implicit functions both 2D and 3D, such as (Figure 5):
(%i1)load(draw);
(%i2)draw2d(implicit(sin(2*x)*cos(y)=0.2,x,-3,3,y,-3,3));

Using parametric() command allows us to create graphs of parametric functions.



-3.82278, 1.08728

Fig. 2. Drawing the function



view: 60.0000, 30.0000 scale: 1.00000, 1.00000

Fig. 3. Three-dimensional graphs

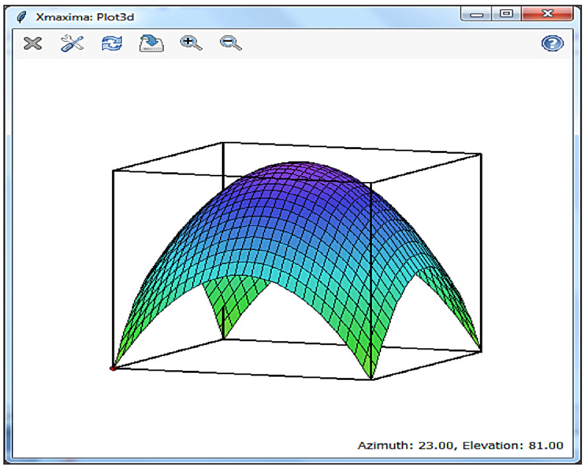


Fig. 4. Rotate the graph

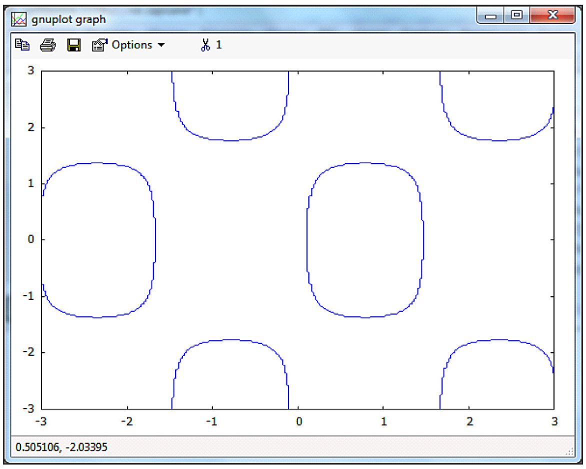


Fig. 5. Graphs of implicit functions

CONCLUSIONS

Very aesthetic visualization has the advantage that by its form encourages learning by creating an attractive environment. The student can freely rotate the graph to see the desired details. Manipulation of color allows, for example distinguish quickly negative number from positive number. The cursor allows reading values precisely. The use of different kinds of visualization explains thoroughly the course of the function, and is also invaluable help for student.

Computer graphs can approach the student to an unavailable reality, things which are outside his environment, phenomena and processes taking place in a great distance, things and phenomena occurring us around. The fundamental principles are more easily assimilated by students thanks visual effects and symbolic calculations.

The Maxima, thanks to its capabilities and appropriate selection tools, helps to solve common problems in different fields, such us economic analysis (e.g. compound), physics (e.g. mathematical pendulum), chemistry and engineering.

REFERENCES

1. Lachowicz C.T.: Matlab, Scilab, Maxima. Opis i przykłady zastosowań, Wydawnictwo Politechniki Opolskiej, Opole 2005.
2. Staranowicz A., Duda P., Orłowski A.: Technologie informacyjne. Wydawnictwo SGGW, Warszawa 2007.
3. Portal projektu MAXIMA http://maxima.sourceforge.net/
4. Portal projektu Otwarta Nauka http://otwartanauka.pl/
5. Portal matematyki obliczeniowej http://computerbasedmath.org/