SCIENTIFIC VISUALIZATION WITH MAPLE

Jiří Hřebíček

Waterloo MAPLE Software GmbH, Heidelberg

1. INTRODUCTION

What is Maple. The name Maple is not an acronym of mathematical pleasure - great fun as it is to use a computer algebra system - but was chosen to draw attention to its Canadian origin [2]. Since November 1980 a lot of work has gone into the development of the Maple system by Symbolic Computation Group (partly in the University of Waterloo and partly in the ETH Zürich). In the simplest terms, Maple is a computer environment for doing mathematics [3] and solving scientific problems. Maple is powerful tool for mathematicians, physicists, chemist, engineers, teachers, ..., in short, for anybody who needs to do mathematical computations. Symbolic and numerical computations and computer visualization can all be done with Maple. The breadth of Maple's functionality is wide - topics from symbolic manipulation with expressions, calculus, linear algebra, differential equations, geometry, statistics and many other mathematical areas are covered, see [1], [2], [3] and [4]. Here is the short description of main Maple functions:

- **Symbolic computations**
  Maple's symbolic computations allow you the greatest freedom. Many mathematical computation such as differentiation, integration, and series expansion of functions, and inversion of matrices with symbolic entries, can be carried out quickly and precisely. By allowing variables to remain unknown (i.e., without numerical values) and in exact form (e.g., 2/3 and 2 as opposed to 0.666... and 1.41...) throughout consecutive steps of a calculation, Maple provides exact answers - with more accuracy than any numerical approximation method. If a floating-point results (number or set of numbers) are needed, they can be calculated at the end of the computation. Thus, roundoff error can be avoided.

- **Numerical calculations**
  Numerical calculations provide alternative methods of solution when a symbolic method would either be too slow or does not exist for a particular type of calculation. Maple supports many well-known numerical algorithms, see [2]. All symbolic mathematical constants and rational numbers can be evaluated numerically. Also, because of Maple's *infinite precision*, numerical calculations can be done to any number of digits accuracy.

- **Computer visualization**
  Maple's two- and three-dimensional graphics provide you with the power of scientific visualization. Functions and expressions in both one and two unknowns, as well as parametric equations and sets of points can be represented graphically. There are about thirty types of special plots as well as many available options for customising the way each graphic is displayed.

What is **Maple's internal structure**? Maple consists of three components: the **kernel**, the **external library**, and the **interface called the Iris**.

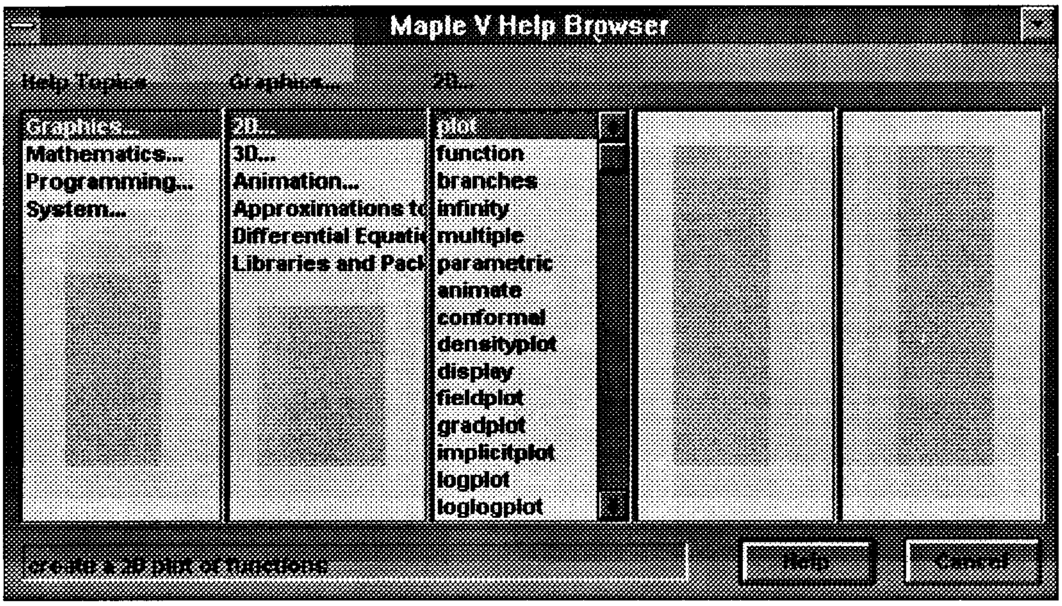
- The **kernel** is the *mathematical engine* behind Maple's calculations. The Iris and the kernel form the smaller part of the system which is highly optimised set of routines written and compiled in the C programming language. The kernel performs the large part of the basic computations done by the system.

- Most of the mathematical knowledge are written in Maple's own programming language and resides as function in the **external Maple library**. *Function written in Maple is* not compiled, but *interpreted* as it is read or entered, allowing you to create your own Maple procedures interactively within the program. When a user needs library function, Maple is in most cases smart enough to load routine itself. Only infrequently used Maple procedures must be loaded explicitly by user. Maple must also be informed about the use of separate package like plots, linear algebra, and statistics packages.

- The **Iris is Maple's eyes** to the world and defines, to a large extent, how you interact with the commands and procedures. It handles input of mathematical expression (parsing and notification of errors), display of expressions, plotting functions, and support of other user communication with the system. Depending on the quality of your terminal, and the version of Maple you are running, the interface may fluctuate between a simple alphanumeric terminal version to a *sophisticated interface* with *Maple documents* (called *worksheets*) containing input, output, text, and graphics.

2. MAPLE'S HELP FACILITIES IN VISUALIZATION

Very useful tool of Maple is its help system. It is explained to you on some examples connected with graphics. The most important is the hierarchical Maple Help Browser for accessing the on line help information. A typical search in the Help Browser is shown on the picture with screen dump of the Help Browser searching for information about two-dimensional graphics and **plot** command.



Selecting the Browser... item in the Help menu opens a help-page browser dialogue box. Maple's help files are organised here in a logical hierarchy up to five levels deep. A topic with three dots (...) following it has related subtopics beneath it. The left-most listing is the highest level of reference and contains several general topics, i.e., **Mathematics, Graphics.** Each list to the right is one level of reference lower. Selecting a topic with three dots, with a single click, opens a list of the related subtopics. Also, selecting any topic updates the synopsis region, located at the bottom of the browser, with a brief description of the help file for that topic. To open a Maple help file from the dialogue box, either double-click the left mouse button directly on a topic, or select a topic and click the

Help button. The browser is particularly useful when you are not sure of the exact name of the command you require. Multiple Maple help pages can be opened in this way and each displayed in a separate window Maple has an in-depth help facility that includes help files for each command, data type, and construct in the Maple language and library. The standard way to call up one of these help files is simply enter the name of the command preceded by a **?** character, and Maple will display the help file if it exists. If no such help file exists Maple will provide you with a list of other possible choices to try. Maple's help facilities are shortly demonstrated via base plotting commands **plot** and **plot3d**.

> ?plot; # only part of help

FUNCTION: plot - create a 2-dimensional plot of functions
CALLING SEQUENCE:
  plot(f, h, v)
  plot(f, h, v,...)
PARAMETERS:
  f - function(s) to be plotted
  h - horizontal range
  v - vertical range (optional)
SYNOPSIS:
- A typical call to the plot function is plot(f(x),x=a..b), where f is a real function in x and a..b specifies the horizontal real range on which f is plotted.
- The plot function provides support for two-dimensional plots of one or more functions specified as expressions, procedures, parametric functions, or lists of points. (When plotting a procedure, operator notation must be used.) See ?plot[function)] for more information on plotting functions. For three-dimensional plots, see ?plot3d. A call to plot produces a PLOT data structure, which is then printed. For information on the PLOT data structure, see ?plot[structure].
- The horizontal and vertical range arguments h and v define the axis labels and the range over which the function(s) are displayed. They take one of the following forms: string, low..hi, or string=low..hi, where low and hi are real constants. See ?plot[range] for further information.
- Remaining arguments are interpreted as options which are specified as equations of the form option = value. In particular, the style option allows one to plot the points as points only, or to interpolate them using line mode. See ?plot[options] for more information.
- Setting the global variable plotdevice (through the interface command) to one of the values listed under plot[device] controls the type of plot which will be produced. See ?plot[device] and ?plot[setup] for information on how to set up plots on a particular device (printer or terminal).

EXAMPLES:
```
plot(cos(x) + sin(x), x=0..Pi);
plot(tan(x), x=-Pi..Pi);
plot([sin(t), cos(t), t=-Pi..Pi]);
plot(sin(t),t); # since no domain is specified, it is used t=-10..10
# Same four plots, but as procedures or operators
plot(cos + sin, 0..Pi);
plot(tan, -Pi..Pi);
plot([sin, cos, -Pi..Pi]);
plot(sin);
# for expressions having discontinuities one can do
plot(tan(x),x=-2*Pi..2*Pi,y=-4..4, discont=true);
plot( -1 + 2*Heaviside(x-1) , x = -1..2, discont = true );
# multiple plots (in a set)
plot({sin(x), x-x^3/6}, x=0..2);
plot(sin(x), x=0..infinity); # infinity plots
# point plots
l := [0,0,1,1,2,1,2,0,1,-1,0,0];
plot(l, x=0..2, style=point);
l := [[ n, sin(n)] $n=1..10];
plot(l, x=0..15, style=point,symbol=circle);
```
SEE ALSO: plot3d plot[<spec>] where <spec> is one of infinity, polar, parametric, multiple, ranges, functions, options, structure, setup, device, replot, style, color

As you can see, a help file contains several different types of information, such as CALLING SEQUENCE, SYNOPSIS, and EXAMPLES.

3. HOW TO USE MAPLE IN VISUALIZATION

One of the most interesting and useful features of computer algebra is scientific visualization. Displaying expressions and functions of one or two unknowns, sets of points pictorially extend our understanding of their nature and significance. Maple provides a large range of plotting commands that accommodate most of your scientific visualization needs. This chapter is meant to give you a overview in how to use Maple in visualization. Plots are created, named and manipulated much like any other algebraic object in Maple, but are displayed in a separate window. These windows can be moved about the screen, resized, and closed in standard Windows fashion. Resizing a plot window redraws the plot to fill the new window size. Multiple plot windows can be opened concurrently or closed at any time. If a plot window has been closed, the plot can be redisplayed either by referring to the plot object by name or by re-entering the appropriate plot command from the worksheet.
The basic concept to plotting is to provide an expression or function in one or two unknowns, and also provide desired ranges for each unknown to be evaluated over. Maple then samples a meaningful set (or grid) of points and displays the results for you. Of course, the quality with which these images appear on your screen are directly proportionate to the graphics capabilities of your monitor. When Maple is installed, the device type for your monitor should be set for you, making the transition to plotting as seamless as possible.

3.1 Two-Dimensional Plots

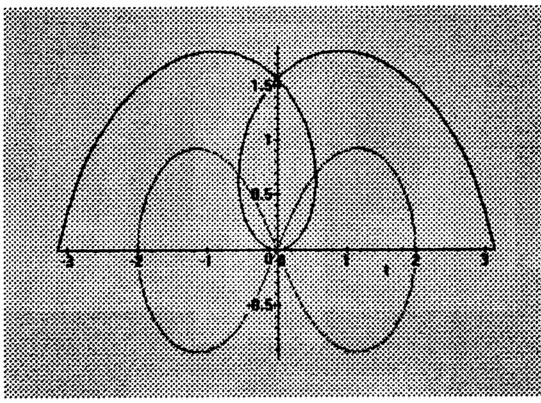
Two-dimensional plots are usually created with the Maple **plot** command. Changes to the appearance of a two-dimensional plot are controlled through parameters of the **plot** command or through the menu items available in the two-dimensional plot window. There are several types of two-dimensional plots available. A few of the more popular ones you could see in the previous chapter in EXAMPLES. Besides these standard plots, there are several other types of two-dimensional plots available in the plots package:

- Polar plots can be produced in the same way as parametric plots except that the option **coords = polar** should be added or by the command **polarplot** from plots package:

```
> with(plots):
> polarplot({t,[2*cos(t),sin(t),t=-Pi..Pi]},t=-Pi..Pi,numpoints=50); # Figure 1
```

- Two-dimensional vector fields can be plotted:

```
> fieldplot( [x/(x^2+y^2+4)^(1/2),-y/(x^2+y^2+4)^(1/2)],x=-2..2,y=-2..2); # Figure 2
```



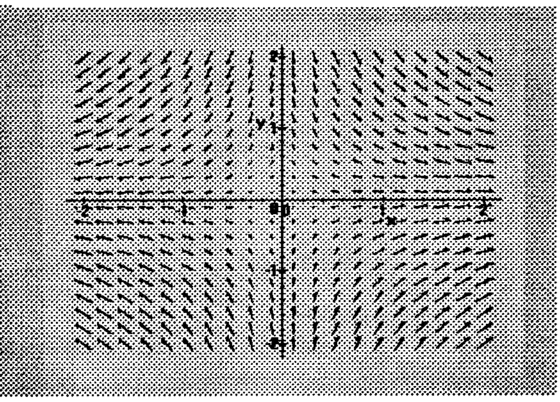


Figure 1 - Polar plot **Figure 2 - Vector field plot**

- Implicit curves defined by an equation can be plotted:

```
> implicitplot(x^2/25+y^2/9=1, x=-6..6, y=-6..6, scaling=CONSTRAINED); #Figure 3
```

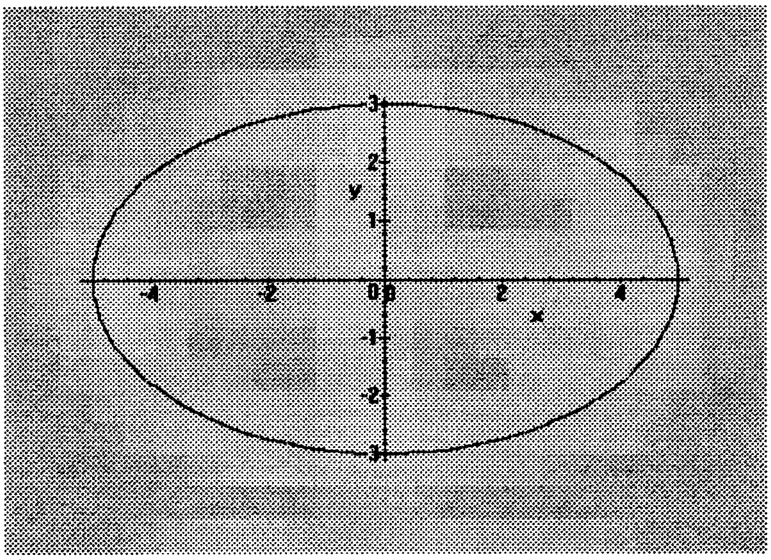


Figure 3 - Implicit function plot

3.1.1 Options of plot command

When Maple plots a graph, it makes many choices. For example, it chooses the depicted range of the graph, sample points to make a smooth curve, which tickmarks to show, and so on. Maple takes care of choosing values for these options that are most convenient in daily use. However you can customise it to your needs. There are several additional options available for two-dimensional plots. The **numpoints** option specifies that more sample points be taken, which gives a smoother curve. (The default value is 49.) Specify what color the plot is drawn in with the **color** option. The **thickness** option specifies the thickness of the curves drawn. The **symbols** option specifies how a point is drawn (one of **BOX, CROSS, CIRCLE, POINT**, or **DIAMOND**). The **axes** option specifies what type of axes (**FRAME, BOXED, NORMAL**, or **NONE**) are used. With the **xtickmarks** and **ytickmarks** options, you control the number of marks that Maple uses along each axis. The **style** option chooses between different styles of interpolation between sampled points (i.e., **LINE, POINT**). A title is added to the plot with the **title** option. Under the Windows or X Window System you can change many options by selecting menu items and/or mouse action.

3.2 Three-Dimensional Plots

Three-dimensional plots are generally created with the Maple **plot3d** command. Changes to the appearance of a three-dimensional plot can be controlled through parameters to the Maple **plot3d** command or through the menu items available in the three-dimensional plot window. More information we obtain using help:

> ?plot3d; #only part of help

```
FUNCTION: plot3d - three-dimensional plotting
CALLING SEQUENCE: plot3d
  plot3d(expr1, x=a..b, y=c..d)
  plot3d(f, a..b, c..d)
  plot3d([exprf,exprg,exprh], s=a..b, t=c..d)
  plot3d([f,g,h], a..b, c..d)
PARAMETERS:
  f,g,h           - function(s) to be plotted
  expr1            - expression in x and y.
  exprf,exprg,exprh - expressions in s and t.
  a,b             - real constants.
  c,d             - real constants, procedures or expressions in x
  x,y,s,t         - names
SYNOPSIS:
- The four different calling sequences to the plot3d function above all define surfaces. Facilities
  for plotting curves and other objects are in the plots package. The first two calling sequences
  describe surface plots in Cartesian co-ordinates while the second two describe parametric
  surface plots.
- In the first call, plot3d(expr1,x=a..b,y=c..d), the expression expr1 must be a Maple expression
  in the names x and y. The range a..b must evaluate to real constants. The range c..d must
  either evaluate to real constants or be expressions in x. They specify the range over which
  expr1 which will be plotted. In the second call, plot3d(f,a..b,c..d), f must be a Maple
  procedure or operator which takes two arguments. Operator notation must be used, i.e. the
  procedure name is given without parameters specified, and the ranges must be given simply
  in the form a..b, rather than as an equation. The second range c..d can have arguments
  evaluating to real constants or procedures of one-variable.
- A parametric surface can be defined by three expressions expr1, expr2, expr3 in two
  variables. In the third call, plot3d([expr1,expr2,expr3 ],s=a..b,t=c..d), expr1, expr2, and expr3
  must be Maple expressions in the names s and t. Finally, in the fourth call, plot3d([ f,g,h
  ],a..b,c..d), f, g and h must be Maple procedures or operators taking two arguments. Here
  again, operator notation must be used.
- Any additional arguments are interpreted as options which are specified as equations of the
  form option = value. For example, the option grid = [m,n] where m and n are positive integers
  specifies that the plot is to be constructed on an m by n grid at equally spaced points in the
  ranges a..b and c..d respectively. By default a 25 by 25 grid is used, thus 625 points are
  generated. See the help page for plot3d[options].
- The result of a call to plot3d is a PLOT3D data structure containing enough information to
  render the plot. The user may assign a PLOT3D value to a variable, save it in a file, then
  read it back in for redisplay. See the help page for plot3d[structure].
- Note: for other types of 3d plots, see ?plots.
EXAMPLES: plot3d
plot3d(sin(x+y),x=-1..1,y=-1..1);
plot3d(binomial,0..5,0..5,grid=[10,10]);
plot3d((1.3)^x * sin(y),x=-1..2*Pi,y=0..Pi,coords=spherical,style=patch);
plot3d(sin(x*y),x=-Pi..Pi,y=-Pi..Pi,style=contour);
# can have variable endpoints in some cases
plot3d(sin(x*y),x=-Pi..Pi,y=-x..x);
```

```
p:= proc(x,y) if x^2 < y then cos(x*y) else x*sin(x*y) fi end:
h:= proc(x) x^2  end:
plot3d(p,-2..2,-1..h);
plot3d([x*sin(x)*cos(y),x*cos(x)*cos(y),x*sin(y)],x=0..2*Pi,y=0..Pi);
plot3d(x*exp(-x^2-y^2),x=-2..2,y=-2..2,grid=[49,49]);
# can specify a color function (or procedure)
plot3d(x*exp(-x^2-y^2),x=-2..2,y=-2..2,color=x);
plot3d(p,-2..2,-1..h,color=h);
#multiple 3d plots can also be done
plot3d({sin(x*y), x + 2*y},x=-Pi..Pi,y=-Pi..Pi);
c1:= [cos(x)-2*cos(0.4*y),sin(x)-2*sin(0.4*y),y]:
c2:= [cos(x)+2*cos(0.4*y),sin(x)+2*sin(0.4*y),y]:
c3:= [cos(x)+2*sin(0.4*y),sin(x)-2*cos(0.4*y),y]:
c4:= [cos(x)-2*sin(0.4*y),sin(x)+2*cos(0.4*y),y]:
plot3d({c1,c2,c3,c4},x=0..2*Pi,y=0..10,grid=[25,15],style=patch);
plot3d({c1,c2,c3,c4},x=0..2*Pi,y=0..10,grid=[25,15],style=patch,color=sin(x));
SEE ALSO: plot, read, save, plot3d[options], plot3d[structure], plot3d[coords],plots
```

There are several types of three-dimensional plots available. Here are a few of the more popular ones:

- Surfaces defined by functions of two variables and expressions in two unknown and multiple plots are described above in EXAMPLES.
- Space curves, tubes, and surfaces defined by parametric equations can be plotted:

```
> plot3d([x*sin(x),x*cos(y),x*sin(y)], x=0..2*Pi, y=0..Pi); #Figure 4
> plots[sphereplot]((1.3)^z * sin(theta), z=-1..2*Pi, theta=0..Pi); #Figure 5
> plots[cylinderplot](1, theta=0..2*Pi, z=-2..2); #Figure 6
> plots[spacecurve]([t*cos(t),t*sin(t),t], t=0..7*Pi);
```

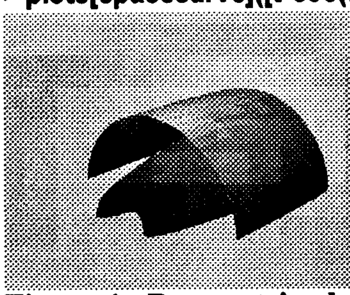


Figure 4 - Parametric plot

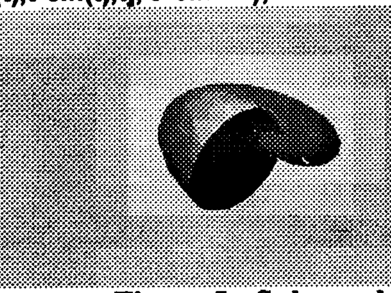


Figure 5 - Sphere plot

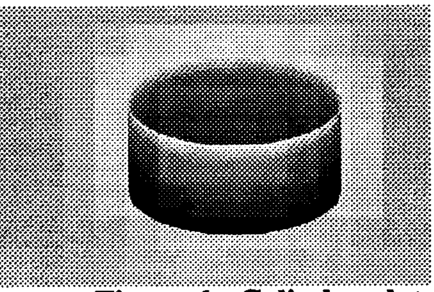


Figure 6 - Cylinder plot

- Some popular polyhedral such as tetrahedron, octahedron, and dodecahedron can be plotted:

```
> plot3d[polyhedraplot]([0,0,0], polytype=dodecahedron,
> style=patch,scaling=constrained,orientation=[71,66]); # Figure 7
```

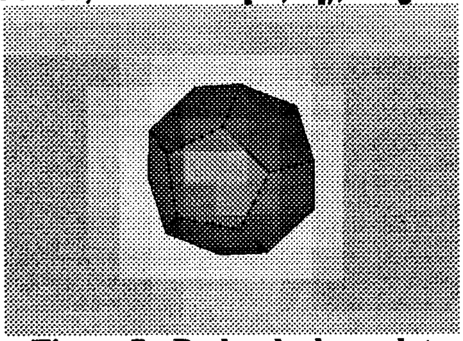


Figure 7 - Dodecahedron plot

- Contour plots and implicit surfaces defined by an equations can be plotted:

```
> plots[contourplot](sqrt(2-x^2-y^2), x=-1..1, y=-1..1); # Figure 8
> implicitplot3d( (x-1)^2 + (y+1)^2 + z^2 = 4,x=-1..3,y=-3..1,z=-2..2); # Figure 9
```

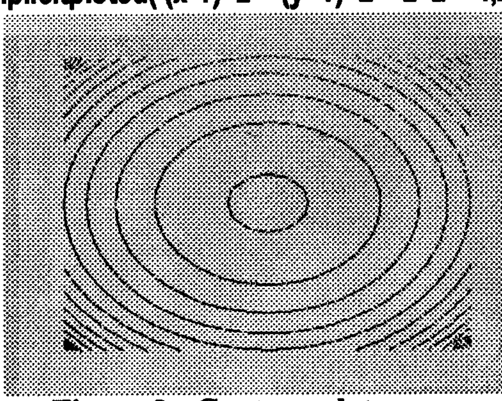


Figure 8 - Contour plot

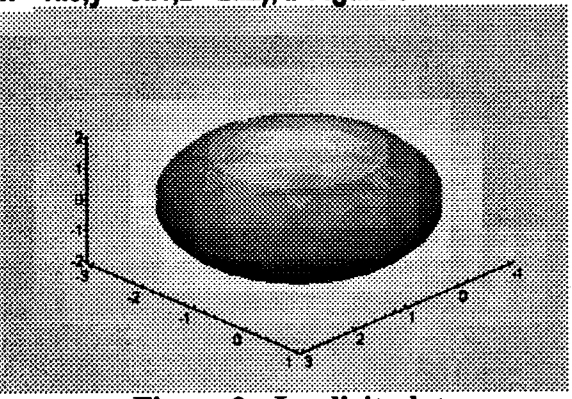


Figure 9 - Implicit plot

3.2.1 Option of **plot3d** command

Just as in two-dimensional case, there are many options to customize three-dimensional plots. The grid option specifies the rectangular grid size for the sample points. (The default value is 25 by 25). Specify the style with which the surface is rendered (e.g., **PATCH, WIREFRAME, POINT, CONTOUR**) with the style option. The number of contours used in a contour plot is specified with the contours option. The **thickness** option specifies the thickness of the curves drawn. The **symbols** option specifies how a point is drawn (one of **BOX, CROSS, CIRCLE, POINT**, or **DIAMOND**). Different coloring schemes are specified with the **color** and **shading** options. Either ambient or directional lights are applied to a surface with the ambientlight and light options, respectively. The **orientation** option specifies from which point in space you view the surface. Labeling of the plot are handled with the **title, labels**, and **tickmarks** options. Under the Windows or X Window System you can change many options by selecting menu items and/or mouse action.

3.3 Graphics packages

There are three special graphics packages for the scientific visualization in Maple:

- Plots package

To use the plots package function, either define that function alone by typing **with(plots,function>),** or define all plots functions by typing **with(plots).**

The functions available are:

animate	**animate3d**	**conformal**	**contourplot**	**cylinderplot**
densityplot	**display**	**display3d**	**fieldplot**	**fieldplot3d**
gradplot	**gradplot3d**	**implicitplot**	**implicitplot3d**	**loglogplot**
logplot	**matrixplot**	**odeplot**	**pointplot**	**polarplot**
polygonplot	**polygonplot3d**	**polyhedraplot**	**replot**	**setoptions**
setoptions3d	**spacecurve**	**sparsematrixplot**	**sphereplot**	**surfdata**
textplot	**textplot3d**	**tubeplot**		

Example:

The following vizualisation of two space functions demonstrates Maple posibility in combinations of more graphs.

```
> with(plots):
> p1:=spacecurve([cos(t)*(10+2*sin(15*t)),sin(t)+2*cos(15*t),t=0..2*Pi,numpoits=1000],
> color=black):
> p2:=tubeplot([10*cos(t),10*sin(t),0],t=0..2*Pi,numpoints=100,radius=1.8,
> style=PATCHNOGRID):
> display({p1,p2},scaling-CONSTRAINED); Figure 10
```

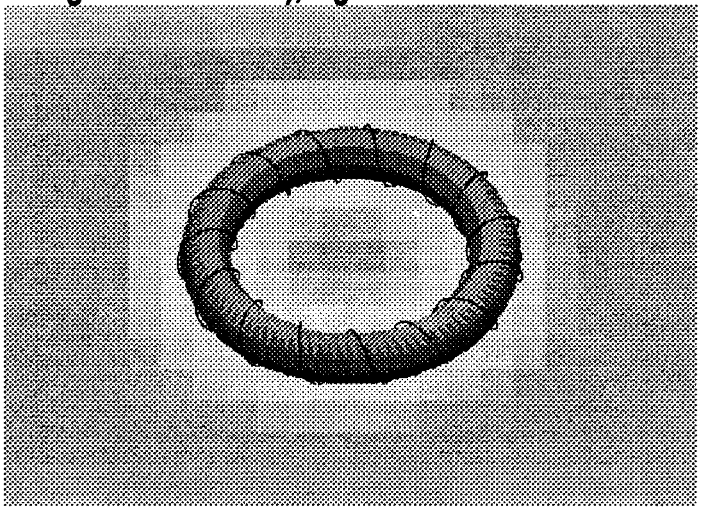


Figure 10 - Space curve combine with tubeplot

- Statplots package
The subpackage **stats[statplots, <...>]** provides various statistical plotting facilities.
The functions available are:

boxplot	**histogram**	**notchedbox**	**quantile**	**quantile2**
scatter1d	**scatter2d**	**symmetry**		

- Differential Equations Tools package DEtools
To use the package DEtools function, either define that function alone using the command **with(DEtools, <function>),** or define all DEtools functions using the command **with(DEtools).**
The functions available are:
Deplot Deplot1 Deplot2 Dchangevar PDEplot dfieldplot phaseportrait

4. ANIMATION
Two- and three-dimensional animation are created with the commands **animate** and **animate3d** from the plots package. Maple animation are a sequence of picture (frames) which is displayed in rapid succession in an animation window. To have animation, there must be an extra unknown in the expression being animated - the variable of animation. Give this variable a range of its own (specified as the last range), and also specify a number of frames. Other than these two things, the calls to **animate** and **animate3d** are very similar to the ones for **plot** and **plot3d**, respectively. Each animation appears in its own window, complete with motion controls much like those found on a animation window.

4.1 Two-Dimensional Animation
A typical call to the animate function **F(x,t)** is:

```
> animate(F(x,t),x=a..b,t=c..d);
```

where **F** is a real function in **x** and **t**, and **a..b** specifies the horizontal real range on which **F** is plotted while **c..d** specifies how the frame co-ordinate varies from one frame to the next. A third range specification will provide the vertical axes information if desired. The animate function provides support for two-dimensional plots of one or more functions specified as expressions, parametric functions, or lists of points. Horizontal and vertical range arguments define the axis labels and the range over which the function(s) are displayed. Remaining arguments are interpreted as options which are specified as equations of the form **option = value**. In particular, the frames option allows one to specify the number of frames to be displayed. The default is 16. The rest of the options are the same as those available for the **plot** command.
Example:

```
> with(plots):
> animate([u*sin(t),u*cos(t),t=-Pi..Pi], u=1..8);
```

4.2 Three-Dimensional Animation

A typical call to the animate3d function **F(x,y,t)** is:

```
> animate3d(F(x,y,t),x=a..b,y=c..d, t=p..q);
```

where **F** is a real function in **x, y** and **t**, and **a..b** and **c..d** specifies the range on which **F** is plotted while **p..q** specifies how the frame co-ordinate varies from one frame to the next. The animate3d function provides support for three-dimensional plots of one or more functions specified as expressions or parametric functions. In particular, the frames option allows one to specify the number of frames to be displayed. The default number of frames is 8. The other options are the same as those available for the **plot3d** command.
Example:

```
> with(plots):
> animate3d(cos(t*x)*sin(t*y),x=-Pi..Pi, y=-Pi..Pi,t=1..2);
```

Graphs of above examples of two- and three-dimensional animation will be presented on the conference together with solving chosen non-linear scientific problems, see [1].
Using Maple visualization in teaching will be also discussed on the conference.

5. REFERENCES

[1] Gander W., Hřebíček J.: Solving Scientific Problems Using Maple and Matlab, Springer Verlag, Berlin, 1994.
[2] Heck A.: Introduction to Maple, Springer Verlag, Berlin, 1993.
[3] Char B.W. and others: First Leaves. A Tutorial Introduction to Maple V. Springer Verlag, Berlin, 1991.
[4] Redfern D.: The Maple Handbook, Springer Verlag, Berlin, 1993.