Computer Algebra and Technical Computing with Maple 8

The Mathematician's Apprentice

Whether you need to perform calculations to a specific degree of accuracy, or without numbers based purely on symbols, computer algebra systems are capable of both tasks. They display the results in numeric formats, as formulae or 3D graphics. Modern programs, such as Maple 8, which we will be discussing in this article, are well-suited for technical computing tasks. BY HOLGER PERLT

Computer algebra systems (CAS) are amongst the most interesting and sophisticated programs around – and not only in the eyes of the mathematician. CAS are completely different from numerical programming languages, such as Fortran, C/C++, or Pascal/Delphi. The latter are designed to work with numbers as solutions to equations or relationships, and use exact calculated procedures or approximations to this end.

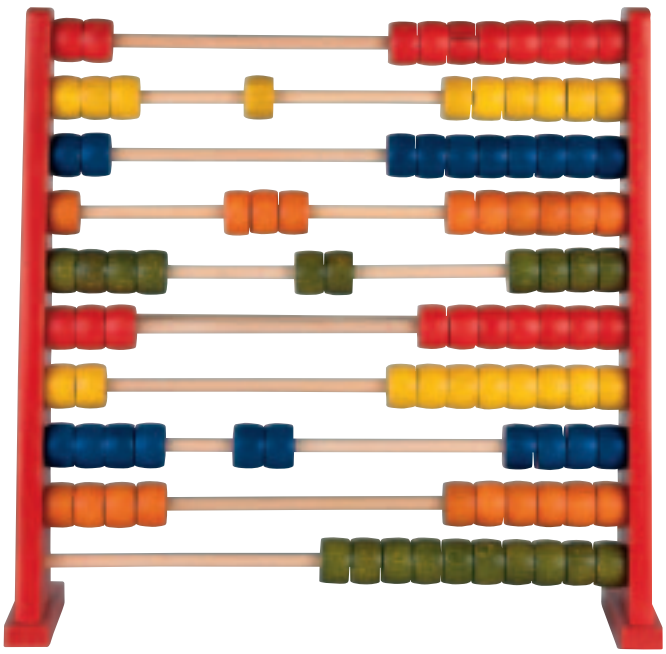
The precision of the results will depend both on the procedure and the type of number you are working with (an integer, or a floating point). Thus, the results of a procedure involving floating point numbers will commonly be an approximation, with speed being the main advantage of this kind of computational processing.

CA systems use both symbols and numbers and are capable of representing numbers to any given degree of accuracy. 1/3 is not simply 0.333333333… to a CAS, but the fraction 1/3. This has far-reaching consequences for solving algebraic problems, and an equally dramatic effect on processing times. Fractions of this type prove a headache in the case of algorithms where enumerators and denominators can reach considerable dimensions. But at least the results will be accurate.

Calculations with an Arbitrary Degree of Precision

Numeric languages will tend to truncate the results radically after every processing step to achieve a specific or the maximum available level of precision. Rounding or truncation errors can add up to produce completely misleading results. Although it must be said that this will hardly affect problems with an average level of complexity. The Maple worksheet in Figure 1 provides an example showing the kind of processing steps that will lead to issues with floating point numbers.

The ability to perform calculations on the basis of symbols is another important feature: The results of a calculation can be symbols (user-definable parameters, for example) and need not necessarily contain only numbers. Symbols are often critical to interpreting the results. They provide us with deeper insight into fundamental theories than a purely numeric program ever could. Over the years a number of CA systems have emerged to form two distinct groups.



Large and Small

Small-scale CA systems are often designed for special areas and will work quickly and efficiently within these areas. Additionally, they are mostly freeware or shareware by-products of university research – or at least not too expensive. This group includes MuPAD [1], Fermat, Cocoa, Singular, Form, and Reduce, although MuPAD is a borderline case, as regards its functionality and marketing.

Large, universal CA systems often provide useful numeric functions and professional graphics as well as symbolic (algebraic) processing features. They are also capable of creating quality documentation, and can make best use of web technologies. Developers will commonly refer to these products as systems

Maple Users	
Target Groups	**Per cent**
Universities and Colleges	60%
Industry	20%
Research Institutes	10%
Schools	10%
Source: Scientific Computers, Aachen, Germany	

for technical computing, rather than Computer Algebra Systems.

Products of this kind are normally beyond the scope of university groups. At some stage in product development a business enterprise is formed to take care of development and coordination. Maple, Mathematica, Macsyma, or Axiom are examples of large-scale CAS. Maple (Waterloo Maple) in particular, and also Mathematica (Wolfram Research) have achieved a high level of market penetration thanks to their well organized marketing and sales structures.

The target groups for these software sytems are universities, colleges and technological enterprises, with a large range of applications. Prices at the top end of the scale tend to prevent use in schools, although this is where systems of this type could be most useful. The user base figures for Maple in Germany and Austria show this situation clearly (see Table 1).

Maple 8 Feature Overview

Universal CA systems have been moving towards more complete solutions in recent years, with development work concetrating on numeric operations and graphics, in addition to symbolic operations. The aim is to allow the user to solve a complex problem using the following steps, and without needing to switch to another program:

- Draft an algorithm
- Test the algorithm
- Apply the algorithm to a problem
- Document the solution professionally

The requirements for these steps differ. The first two depend on a high level of flexibility and transparency, since sophisticated algorithms and functions are involved. The user will probably want to test the draft algorithm in every imaginable scenario possible – and this is often impossible with purely numerical procedures.

The third step requires enormous processing power, and is often the achilles heel of the CAS. Since a CAS is not a compiler language its numeric processing speed will be slower than that of C/C++ or Fortran. The last point requires the features of a top-notch word processor – and a lot of work has gone into this area over the last few years.

This concept becomes evident when you consider the fact that a user can spend a whole session within a sheet or notebook, using these formal pages to author program code and documents, and perform calculations.

This is also where the symbolic, numeric, and graphic results will be available. You could write whole books with this user interface. Purists may tend to stick to the command-line version – especially if they only need to compute some results.

Central Features

Maple's central features include the symbolic (algebraic), numeric, and graphic modes. However, assessments should be based on the symbolic mode, as this is the primary benchmark for a CAS. Symbolic mode can be further broken down into the following:

- Basic operations, substitutions and simplifications
- Analysis (Calculus)

Maple 8 offers the standard you would expect from other CA systems in this area. Let's look at the simplification of expressions as an example:

The function has a set of algorithms that have been enhanced over the years. Using efficient routines for simplification are critical to solving complex problems.

Maple is specifically capable of taking assumptions concerning value ranges and other conditions into consideration when performing the simplification tasks.

Calculus is concerned with solving equations and (partial) differential equations (DE), integrating and solving problems involving limiting values. This is the core issue for a large number of scientific and technical users. Almost any problem you look at within these fields will produce a differential equation. If you have access to a generic symbolic solution, you can investigate the problem at hand from various viewpoints – and this is ideal for technicians, scientists, or students.

Numeric Mode

Providing a numeric mode cannot be considered a traditional task of symbolic calculation. The idea of developing a system for universal use in various areas of science and technology certainly provided ample incentive for the development of this mode. Maple entered into a strategic alliance with NAG to avoid losing out to tried and tested numeric routines. All the most important routines providing numeric solutions for problems in the area of linear algebra derive from the well-known NAG program library. This allows the CA system to solve standard problems involving vectors or matrices at an acceptable speed.

Maple differentiates betweeen two types of decimals: Software floating point numbers refers to the standard representation of decimals, and allows an arbitrary level of precision for numerical tasks, independently of the

Figure 1: This Maple worksheet demonstrates the difference between exact and approximate representation of numbers. Rounding of preliminary results often impacts the final result

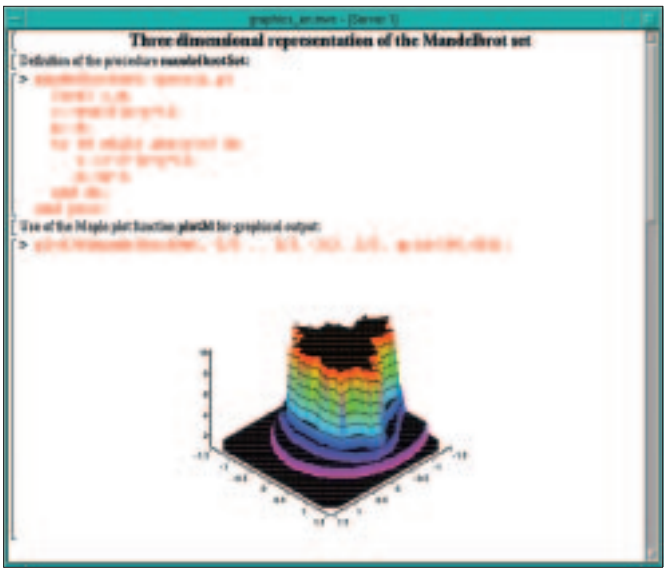


Figure 2: Maple 8 can display complex 3D graphics: *plot3d* shows a Mandelbrot set for a given set of values

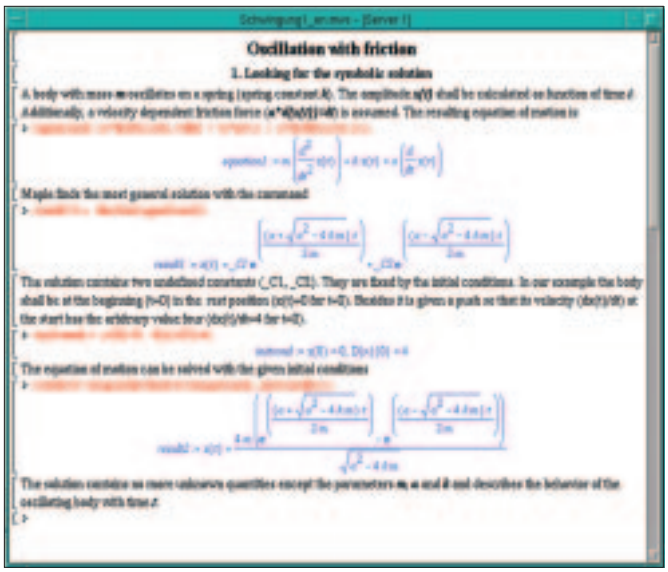


Figure 3a: A mass attached to a spring is pushed. How will the mass react? Maple finds the general, symbolic solution

computer involved. Precision is defined by the *Digits* variable. In contrast, hardware floating point numbers depend on the machine's internal facilities. Calculations with this type of notation are often quicker, however, their accuracy depends on the processor used.

Floating point calculations are invoked by the *evalf* instruction, which causes Maple to perform the operation in software floating point mode. Additionally, Maple uses different keywords for some symbolic and numeric functions that basically solve the same problem. As different routines need to be accessed, it often does not make sense to run a symbolic calculation for a purely numeric problem, and to then substitute numbers when the result is known. It is a lot quicker to use equally efficient, numeric algorithms.

The counterpart for hardware floating point calculations is *evalhf*. However, this command has several restrictions in comparison to *evalf*. A numerical solver for partial differential equations was introduced to Version 8. This involved considerably enhancing the functionality of the *pdsolve* function.

Graphics

Maple 8 offers a variety of graphic display features to suit all needs. However, if a feature you really need does happen to be missing, you can add your own Maple code to customize the program. The graphic routines include:

- Basic library routines, such as plot or plot3d
- Special graphic routines in the plots and plottools packages, including animated graphics
- The DEtools package for solutions to differential equations
- stats for statistical data

The last two packages are particularly valuable if you are required to complete complex tasks. If you have ever had to program variable graphic objects, you will appreciate the ease with which Maple can display high quality results: Figure 2 shows a Mandelbrot set, constructed using one of Maple's basic graphic routines.

Version 8 includes a new package that demonstrates the concept of a unified approach to problem solving with Maple in the context of a large documentation base: *Student Calculus1* is aimed at first year students. The package includes a wealth of problems and solutions based on the analysis of the functions of a variable, prepared and presented to suit

Technical Computing in Action

Pictures 3a to 3c demonstrate Maple's uniform approach. Each picture shows a worksheet, where the user can enter texts, and define the problem to be solved. Sheets are extremely informative, interactive documents that can be exchanged across platform boundaries. Other users can repeat calculations at any time, and add their own modifications. This is useful for engineers wanting to exchange ideas, and of course for teachers and lecturers wanting to map out complete maths or physics courses.

Figure 3a shows how a basic differential equation is defined and solved algebraically. The code that needs to be run is shown in red, and Maple's answer in blue. Entries are made just like a user would make them on paper – the only exceptions being the cases where statements are terminated or the values substituted.

This display principle makes using the program a lot easier – and Maple always attempts to display the results in a fashion that allows for the maximum in readability.

Figure 3b shows two manipulations intended to characterise the behavior of the symbolic solution, one for smaller periods and the other for larger time values. Complex results will often prohibit a simple evaluation of their outcome.

Finally, Figure 3c shows a numerical solution to the differential equation in Figure 3a. Here, Maple allows the user to display the results as a graph, which is extremely useful. To do so, you simply add the routine from the DEtools package and let Maple take care of the rest.

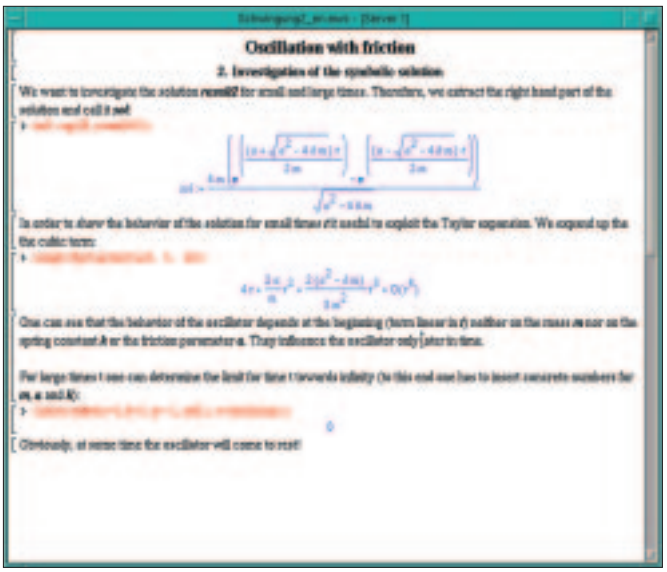


Figure 3b: The generic solution for an oscillating spring is quite clear. However, Maple can reduce the formula even further

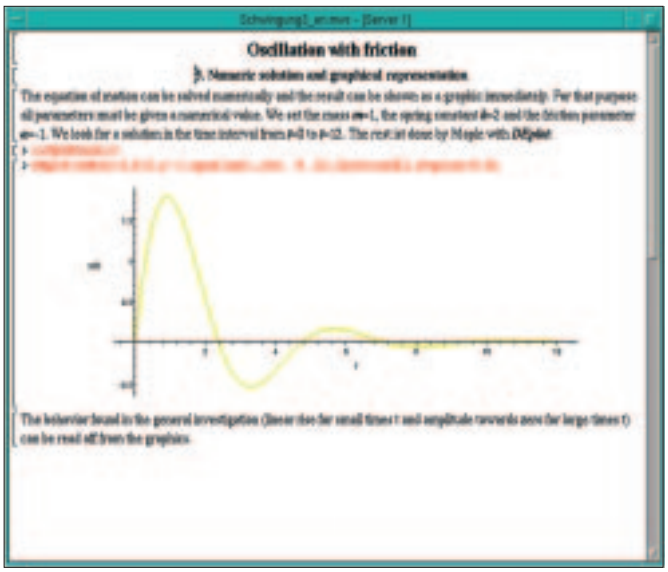


Figure 3c: The spring's behavior becomes more apparent, when Maple displays the motion as a graph. This requires a numeric solution

the requirements of a student entering higher education.

Maple's Structure

Maple comprises three components: the kernel, the program library and the user interface. The kernel was programmed in C and is responsible for low-level operations. These include arithmetic, file input and output, executing the Maple programming language, and the efficient execution of basic mathematical operations (for example, the derivation of polynomials).

The program library comprises of almost every mathematical function. It is written in the Maple language and parts of it are loaded by the kernel, when needed. The user interface comprises of both a GUI and a command-line version.

Third-party programs can also use the Maple routines and provide their own user interfaces. The technological program, Matlab, is a good example of this. A recent addition, Maplets, even allow you to program a GUI of your own.

Maple distinguishes five internal functional groups:

- Evaluators
- Algebraic functions
- Algebraic auxiliary functions
- Data structure manipulators
- General auxiliary functions

Evaluators are responsible for various kinds of calculations. These include statements, algebraic expressions, boolean expressions, naming conventions, floating point calculations with arbitrary precision or hardware floating point arithmetic.

The algebraic functions include basic functions, such as *diff* (derivations), *divide* (division of polynomials) and *coeff* (which calculates the coefficients of polynomials).

The algebraic auxiliary functions can not usually be called directly, but are instead referenced by functions of the two preceding groups. They include simplifications of expressions and arithmetic packages.

Data structure manipulators can be applied both to mathematical objects and to data structures.

They include *op* (selects the operands for an expression), and *length* (which ascertains the length of an expression). The final group – general auxiliary expressions – is at the base of the hierarchy. It takes care of storage, internal input/output management, and program exceptions.

The Maple user has access to more than 3000 commands, from Maple routines, through auxiliary functions, to evaluators. This makes Maple's feature

Physics and Computer Algebra

One area of application for CAS is the theory of elementary particles. The complex rules of perturbation theory can be described in the programming language of a CAS. This allows physicists fundamental insight into the complex world of subatomic interaction. This field provided considerable impulses toward the development of CAS in the 70s.

The Dutch physicist, J. Vermaseren, has made considerable contributions towards the development of an efficient CAS geared to the requirements of investigations into the perturbation theory. This program is FORM. Stephen Wolfram and Mathematica are also prominent examples for the symbiosis between physics and CAS.

At the same time mathematicians have been working on developments to allow the use of computer algebra for applied research in the fields of Group Theory and differential equations.

Maple 8

Adept Scientific plc

Stand Alone Commercial Version: approx £1,300

Student Version: approx £125

Special licenses for research, universities, and schools are availabe; individual pricing on request

http://www.adeptscience.co.uk

list one of the fullest among programming systems of this kind.

Programming with Maple

Maple's own programing language allows the user to write complex programs, with a clear structure. If you are familiar with C, Fortran, or Pascal/Delphi, you should have no difficulty in mastering Maple. The Maple language is even educative for beginners: If you are familiar with Maple, you will soon come to terms with numeric programing languages. As Maple is not a compiler language, you can test your program code immediately after writing it. This is particularly useful for beginners. Procedures, modules and packages are some of the central aspacts of the Maple language. This kind of structuring is essential to more complex applications. But Maple leaves virtually nothing to be desired. Modules and procedures are even platform independent. Packages are sets of procedures and data that permit calculations in specific fields. Figure 2 shows a simple example of a procedure in the Maple programing language.

Open for Other Languages

It often makes sense to combine Maple with other programming languages – and this is possible in both directions. You can use the *CodeGeneration* function to translate native Maple code to its counterpart in the numeric compiler languages C, Fortran, or Java.

Maple can also process code compiled in other languages, provided it is accessible as a library. You will need a Shared Library for Linux: *lib*XYZ.*so*. You can then use the Maple *define_external* function to call the external routine just like you would call a native Maple procedure. This may allow quicker processing or permit the use of special numeric algorithms. Maple 8 sees the introduction of a new feature – the Maplet.

Maplets are graphic user interfaces that run within a Maple session. They allow the user to combined packages and procedures with interactive windows and dialogs, thus producing a tailor-made desktop. Unfortunately, this feature is only available within a Maple session. As the name suggests, the Maplet package is based on the Java Runtime Environment. Version 8 introduces enhancements for processing XML files with more flexible functions. Users must bear in mind that Maple uses its own conventions.

User Community

As Maple has been around for several years now, a large user community has grown. Also, a large number of books dealing with special interest topics in science and technology have been published – not to forget the numerous procedures and packages that users can download on the Internet free of charge.

Waterloo Maple provides a lot of support here (of course, it is in their own interest to do so): The company has set up a so-called Application Center. The website at [7] provides users with hundreds of sample solutions, Maple worksheets, and program code for dozens of fields. You will also find links to innumerable books, reports, and articles that refer directly, or indirectly to Maple. This makes life easier for newcomers, but even experienced users continually discover new applications. ■

Tests and Benchmarks

It is not easy to evaluate the capability of a CA system, and the reviewer's subjective viewpoint is often apparent. But experts from various universities have put some thought into this matter and come up with three major benchmarking areas:

- Solution of algebraic and transcendental equations
- Solution of differential equations
- Calculation of integrals

Nearly every research task will boil down to one of these areas sooner or later. Current versions are occasionally benchmarked along these guidelines and the results are available on the Web. CAS developers do take this seriously. Michael Wester [2], Laurent Bernardin [3], Hans-Gert Graebe [4], and Stefan Steinhaus [5] are probably the most highly regarded benchmarkers.

Without looking at each of the test results individually, one can still say that Maple performs extremely well in all tests. This is true of the major problem areas. Maple users have access to a state-of-the-art tool that will allow them to solve the most complex of problems in a majority of cases.

Kamke's Manual of Ordinary Differential Equations provides an almost classic test suite for ordinary differential equations. It comprises nearly every kind of standard DEQ occurring in applied mathematics. As E. S. Cheb-Terrab [6] reported, Maple 7 was capable of solving 1273 of the 1316 examples – that is, a grand total of 96.7 per cent.

Maple has always played a leading role as regards solutions for standard and partial differential equations and this also applies to algorithms for solving equations.

THE AUTHOR

Dr. Holger Perlt is a physicist who has worked in the field of theoretical, elementary particle physics. Dr. Perlt started using computer algebra in the late 70s. He has spent the past few years working on the implementation of modern approaches to self-learning optimization in software for complex technological processes.

INFO

[1] MuPAD 2.0 : *http://www.mupad.de*

[2] Michael Wester, "A Critique of the Mathematical Abilities of CA Systems": *http://math.unm.edu/~wester/cas_review.html*

[3] Laurent Bernardin, "A Review of Symbolic Solvers": *http://www.inf.ethz.ch/personal/bernardi/solve/*

[4] Hans-Gert Graebe, "About the Polynomial System Solve Facility of Axiom, Macsyma, Maple, Mathematica, MuPAD and Reduce": *http://dol.uni-leipzig.de/pub/1998-11/en*

[5] Stefan Steinhaus, "Comparison of mathematical programs for data analysis": *http://www.scientificweb.de/ncrunch/*

[6] E. S. Cheb-Terrab, "Comparison of Performances in solving ODEs using Maple 7 and Mathematica 4.1": *http://lie.uwaterloo.ca/odetools/comparison.html*

[7] Maple Application Center: *http://www.mapleapps.com*