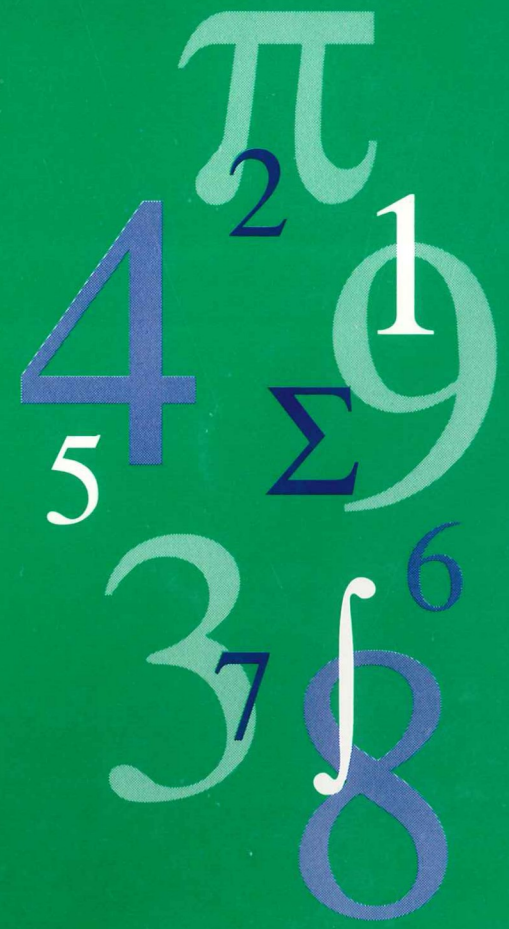
Volume 1, Number 1 Spring 1994

MAPLETECH

The Maple Technical Newsletter



Birkhäuser
Boston Basel Berlin

$14.50

THE MAPLE TECHNICAL NEWSLETTER

A Newsletter sponsored by Waterloo Maple Software for the communication of applications among creators and users of Maple Software

Subscription Information

The Maple Technical Newsletter is published twice annually in the Spring and the Fall. Issues Nine and Ten were published in 1993. With the appointment of an Editorial Board in 1994, a new numbering begins with Volume One

Annual subscription rate (1994): US $40.00 plus $6.00 postage and handling charges. (Canadian customers please add 7% GST to subscription rate, then add postage and handling. Birkhäuser Boston's GST registration number is 123394918.) Outside North America: Please place your order with your bookdealer or send directly to Birkhäuser Verlag, Journal Subscription Department, P.O. Box 133, CH-4010 Basel, Switzerland. Telephone: + + 41 (0) 61 271 7400; Fax: + + 41 (0) 61 271 7666; Telex: 963475 birk ch. A reduced rate is available to individuals. Please write to the publisher for details. All orders must be accompanied by payment, and if originating from the USA and Canada should be sent to:

Birkhäuser, Boston
c/o Springer-Verlag New York, Inc.
Journal Fulfillment Services
P.O. Box 2485
Secaucus, NJ 07096-2491
(201) 348-4033
Toll Free 1-800-SPRINGER
Fax (201) 348-4505

Change of Address:

Allow six to eight weeks for all changes to become effective. All communications should include both the old and new address (with Postal Codes) and should be accompanied by a mailing label from a recent issue.

Advertisements:

Correspondence concerning advertisements should be sent to:

Mr. Joe Barrett
Birkhäuser
175 Fifth Avenue
New York, NY 10010, USA
Telephone: (212) 460-1700

Office of Publication
Birkhäuser, Boston
675 Massachusetts Avenue
Cambridge, MA 02139
Telephone: (617) 876-2333

Copyright:

The Maple Technical Newsletter Staff
Editor: Dr. Tony Scott
Mathematical Institute
University of Oxford
24-29 St-Giles
Oxford OX1 3LB
United Kingdom
Phone: +44 (0) 865 (2) 73556
email:tcscott@maths.oxford.ac.uk
scott@cfa.harvard.edu

Assistant Editor:
Reid M. Pinchback
Academic Computing Services
Massachusetts Institute of Technology
email:reidmp@mit.edu

Editorial Board:
Jacques Carette, INRIA, Rocquencourt
Robert M. Corless, University of Western Ontario
Greg Fee, University of Waterloo
Philippe Flajolet, INRIA, Rocquencourt
Robert J. Lopez, Rose-Hulman Institute of Technology
Michael B. Monagan, ETH, Zurich
Kelly B. Roach, University of Waterloo
Alastair Rough, University of Waterloo
Marc Rybowicz, University of Limoges
Bruno Salvy, INRIA, Rocquencourt

Sponsored by

Waterloo Maple Software

MAPLE

Waterloo Maple Software
450 Phillip St.
Waterloo, Ontario
Canada
N2L 5J2
Phone: (519) 747-2373
FAX: (519) 747-5284
email:info@maplesoft.on.ca

European Office:
Waterloo Maple Software GmbH
Tiergartenstraße 17
69121 Heidelberg 1, Germany
Phone: +49-6221-487 180
FAX: +49-6221-487 184
email:100275.163@compuserv.com.

CONTENTS

A Note from the Editors

April 1994, Vol. 1, No. 1

If you have peeked at the front cover information, you may have noticed the categorizations of the articles within the Table of Contents and the list of members of our newly formed *editorial board*, and thus realized that the Maple Technical Newsletter has been "re-born" as an officially refereed publication. That's why we start fresh with this particular issue marked as "Volume 1, Number 1". In the past, we always had "refereeing" of the articles at a grass-roots level but our referee organization refined itself at each issue to the extent that the recognition of an official editorial board became necessary. The names in this list include people who have already made valuable contributions to the Newsletter.

As mentioned again in our new *guidelines for authors*, the Maple Technical Newsletter has been defined by essentially three types of articles:

(i) Articles on Maple functionality.

(ii) Education-level articles.

(iii) Articles on Research/Applications.

For educational-level articles, *accessibility* of the article is especially important as our readership includes people from various disciplines. This issue includes three such articles. The article entitled "Maple in Education, Part II" by Scott *et al.* shows the type of worksheets we are eager to make accessible to everyone from the share library (as discussed in both the article itself and the update on the share library within the *News and Announcements* section). Of course, we also continue our column entitled *Tips and Techniques* which presents yet a fourth category and which has received favorable response. In this way, we hope to cover a very broad range of articles.

For this issue, the section entitled *Features of Maple V Release 3* shows the capabilities of this most recent version of Maple. Our previous requests for applications of Maple in all areas seems to have born fruit. Besides the conventional areas of science such as nuclear magnetic resonance, this particular issue shows an unusual application of Maple, namely its use in a model for snow transport.

As before, the Newsletter includes updated reports on the share library, the availability of Maple books and manuals, recent conferences, and new platforms for the Maple system. Details on how to submit articles for publication in this Newsletter are available in the section entitled *Guidelines for Authors* on page 93. Everyone is encouraged to do so. We especially welcome technical and educational articles that are accessible and self-explanatory to the general Maple user. Applications in *all* areas are of interest, not just the areas of mathematics, science, medicine or engineering. Comments or inquiries should be addressed to the editors.

About the Articles and their Authors

To inform the reader about some of the new features of MapleV Release 3, the first two articles of this issue are devoted to extended functionality. The first article, written by Robert Corless and Michael Monagan describes Maple's `assume` facility. This article includes examples of how `assume` is used to solve problems and how it is integrated into Maple. It also discusses the limitations of this facility and proposes how Maple should evolve to a "lastproviso" model of computation. Robert Corless received his doctoral degree in mechanical engineering at the University of British Columbia and is presently an assistant professor at the University of Western Ontario. His background also includes computer science and applied mathematics, and his interests include computational aspects of dynamical systems, numerical solutions of ordinary differential equations, flow-induced vibration and perturbations. As a Commonwealth scholar from New Zealand, Michael received his Ph.D in Computer Science in 1990 from the University of Waterloo and is one of the most significant contributors to the Maple system. He is presently doing post-doctoral research at the Eidgenoessische Technische Höchschule (E.T.H) in Zürich, Switzerland, with the Institute for Scientific Computation. Besides finite and algebraic number fields, his research interests also include automatic program differentiation, Fortran/C code generation, and optimization. He has also made many contributions to this newsletter.

The second article, written by Zaven A. Karian and Rohit Goyal, is a functionality article in the area of random number generation which provides performance figures for the `rand` function. Zaven Karian obtained his doctoral degree in Mathematics from the Ohio State University and has a background in Computer Science. Since 1964, he has been at Denison University where he holds the Benjamin Barney Chair of Mathematics. His primary area of interest is the study of computer simulations. Rohit Goyal is a senior undergraduate Computer Science major at Denison University where he has been involved with the use and instruction of Maple. Currently, he is working on a project that uses the C-callable version of Maple to fit probability distributions to data. His other interests include artificial intelligence and object oriented programming. The third article, written by Paul Zimmermann, is an expository article on a Maple implementation called Gaïa which is a package that helps count and draw random combinatorial structures of various sorts. Paul Zimmermann got his doctoral degree in 1991 at l'École Polytechnique (France). His main research topic is the average-case analysis of algorithms. With Bruno Salvy, he designed the system Lambda-Upsilon-Omega (LUO) that gives an asymptotic expansion for the average-case complexity of programs, and the Maple package `gfun` for the manipulation of holonomic sequences.

The next article written by Nicola Pio Belfiore, Ettore Pennestrì and Rosario Sinatra, is in the area of mechanical engineering and presents a Maple-based approach to gear analysis which is relatively unique in mechanical design and could be of value to others in the field. Its two unique contributions are in the areas of graph-theoretic analysis of mechanical networks, and symbolic approaches to mechanical analysis. Ettore Pennestrì received his doctoral Engineering Sciences degree from Columbia University and is currently an Associate Professor at the University of Rome (Tor Vergata).. His research interests include kinematics and dynamics of machines. He is the author of a book (in three volumes) on kinematic synthesis of linkages and has been awarded with the A.M.R. International Kinematician Award (1993-1995). N.P. Belfiore completed the doctoral program in Applied Mechanics at the University of Rome in 1992 with a thesis on the structural synthesis of robotic wrists. He is teaching courses on Rational Mechanics and Mechanics applied to Machines at both the University of Cassino and the University of Rome (La Sapienza). Rosario Sinatra graduated in 1985 at the University of Catania where he is currently a researcher in the department of Mechanics Applied to Machines. His research interests include machine design, robotics and mechanical vibrations.

The next article, written by Johannes Grotendorst, Paul Jansen, and Siegfried M. Schoberth, involves an application in the area of Nuclear Magnetic Resonance (NMR) spectroscopy. It demonstrates various ways of how the McConnell equations can be solved analytically using Maple and verifies solutions found in the literature. It also indicates computation techniques of mixed symbolic-numeric type for the determination of the formal parameters involved in the solutions. Johannes Grotendorst obtained his doctoral degree in theoretical

chemistry at the University of Regensburg and his background also includes studies of mathematics and physics at the University of Bielefeld. Since 1988, he has been at the Institute for Applied Mathematics at the Jülich research center (KFA). His current research interests include approximation of functions, series and integrals (he developed the Maple share library package `trans`), and integration of symbolic and numeric computation for mathematical problem-solving in applied science. Paul Jansen obtained his diploma in mathematics at the Technical University of Aachen (RWTH) and is currently at the the Institute for Applied Mathematics at the Jülich research center where he lectures in numerical and statistical mathematics. His interests include mathematical and statistical software and inverse problems. Siegfried Schoberth obtained his doctoral degree in microbiology at the University of Göttingen. He is currently a research scientist in the Institute for Biotechnology at the Jülich Research Center. His research interests include "in vivo" NMR spectroscopy and general microbiology.

The next article, aptly described by its title: "Modelling Snow Transport, An Application of Maple" and written by Christian Hoffmann, involves perhaps the most esoteric application of Maple seen so far. Christian Hoffmann received his mathematical training at the Polytechnical Institute at Darmstadt, Germany, and at the Federal Institute of Technology at Zürich, Switzerland. His doctoral thesis was completed in 1976 and he is currently working as a mathematical and statistical consultant at the Federal Institute of Forest, Snow, and Landscape Research near Zürich. His interests include numerical mathematics, the use of computer algebra in practical problems, statistical analysis, computer science, modelling, and astronomy.

The next three articles involve educational-level subjects. The first of this set presents some educational examples in Physics, Chemistry and Engineering prepared on *worksheets* using MapleV Release 3. We hope these examples will encourage Maple users to make similar types of worksheets and will show users the type of accessible applications we wish to include within the Maple Share Library. This article is a sequel to *Maple in Education* as presented in issue 7 of this newsletter and is co-authored with Michael Monagan, Richard Pavelle and Darren Redfern. Richard Pavelle, a leading figure in the area of symbolic computation, was on the research staff at the MIT Laboratory for Computer Science and MIT Lincoln Laboratory for more than 10 years, and has pioneered the development and use of symbolic systems since 1973. Among many other things, he is an inventor, the author of a *Scientific American* article on Computer Algebra, and an expert in using computer algebra systems to find analytical solutions for mathematical problems in science, engineering, and finance. Darren Redfern is a graduate of the Math Faculty of the University of Waterloo. After working as the Product Support Manager for Waterloo Maple Software, Darren started his own company, Practical Approach, in 1992. He is creating a third-party industry for products such as Maple, MATLAB, and others by providing books, training, and packages of code that supplement the basic functioning of these products. As one of his many duties, he is the Series Editor for all of Springer-Verlag's Maple material. He was also an assistant-editor of this Newsletter from issues 6 to 9 inclusive.

The next article is a Maple worksheet prepared by Robert Lopez and presents some simple applications of Maple to solving ordinary differential equations. With a doctoral degree in Relativistic Cosmology obtained in 1970 from Purdue University, Robert Lopez considers himself a classically trained applied mathematician. At the Rose-Hulman Institute of Technology, he pioneered the use of Maple in teaching mathematics to engineering students, won the Dean's outstanding Teacher Award and co-authored a numerical analysis text. He has also spent the last two years in association with Waterloo Maple Software giving workshops and presentations on the use of Maple in the classroom.

Ross Taylor joins us again with another contribution, namely a readable article in the area of thermodynamics which shows how Maple can be used to perform some simple phase equilibrium calculations, create phase diagrams and interpret data. Its contents could be also be useful to workers using Graph Theory in other areas. Ross Taylor obtained his doctoral degree in Chemical Engineering from the Institute of Science and Technology at the University of Manchester in 1980. He is a professor of chemical engineering at Clarkson University in Potsdam, New York and his particular interests are in mass transfer and design.

News and Announcements

Maple Connections

The New Maple/Prescience Info Server

In our last issue, we had mentioned the merger of Waterloo Maple Software with Prescience Corporation, a developer of products such as Theorist and Expressionist. Now available by anonymous ftp from ftp.maplesoft.on.ca is the "Info Server", an ftp site containing the Maple Share Library, demos, general information about Maple, Theorist and Expressionist, and support information. Some of the things you can find in the Info Server are:

- Maple V Release 3 for DOS and Windows demo.
- Theorist demo.
- Expressionist demo.
- Information about the Theorist Gopher Server.
- Maple Sample Engineering Worksheets.
- Answers to common support questions.
- Maple availability lists.
- List of Maple books and manuals.

Information is updated regularly. If you have any questions about the Maple Info Server, please contact Waterloo Maple Software Technical Support at support@maplesoft.on.ca.

Shows and Conferences

An Open Invitation to MSWS 94

Maple Summer Workshop and Symposium
August 9-13, 1994, Rensselaer Polytechnic Institute (RPI)

Waterloo Maple Software (WMS) is currently making plans for MSWS '94. The conference will be held at RPI on August 9-13, 1994. This year's event includes a two-day workshop with courses in:

- Discovering Maple V.
- Maple V Programming Fundamentals.
- Mathematical Applications with Maple V.
- Advanced Maple V Programming Techniques.
- Maple V in the Classroom.
- Algorithms for Computer Algebra.

The applications symposium will contain invited talks from leading personalities, panel discussions, workshops, open forums, and paper presentations from researchers, educators and other innovative users from all fields.

Also offered is a Spousal Program which includes a scenic and historic cruise of the Hudson or Mohawk Rivers, guided walking, driving or trolley tours of historic Albany and Troy, the Henry Hudson Planetarium, Shopping Excursions, bowling, and the Saratoga Thoroughbred Races.

Dr. William H. Graves from the University of North Carolina at Chapel Hill is this year's MSWS'94 banquet speaker. William Graves is the Associate Provost for Information Technology and the Director of the Institute for Academic Technology at UNCCH. He is also involved at this year's EDUCOM conference as the co-leader for a project entitled "Toward a National Learning Infrastructure".

Further details on the program will be released as they become available. If you would like to be on the mailing list for MSWS '94 information, please contact:

Paola D'Alessandro
Events Coordinator at WMS
Waterloo Maple Software
450 Philip St.
Waterloo, Ontario, N2L-3L3, Canada
Phone: (519) 747-2373
FAX: (519) 747-5284
E-mail: pdalessa@maplesoft.on.ca

Maple Publications

- *Solving Problems in Scientific Computing Using Maple and MATLAB* by W. Hrebicek Gander, (Springer-Verlag), ISBN 0-387-57329-1 (1993).

  A first comprehensive applications book of Maple and MATLAB to solving non-trivial problems in scientific computing. 250 pages.

  Contents: The Tractix and Similar Curves, Trajectory of a Spinning Tennis Ball, The Illumination Problem, Orbits in the Planar Three-Body Problem, The Internal Field in Semiconductors, Some Least Squares Problems, The Generalize Billiard Problem, Mirror Curves, Smoothing Filters, The Radar Problem, Conformal Mapping of a Circle, The Spinning Top, The Calibration Problem, Heat Flow Problems, Penetration of a Long Rod into a Semi-infinite Target, Heat Capacity of System of Bose Particles, Compression of a Metal Disk, Gauss Quadrature, Symbolic Computation of Explicit Runge-Kutta Formulas.

- *Exploring Calculus with Maple* by H.H. Holmes, J.G. Ecker, W.E. Boyce and W.L. Siegmann (Addison-Wesley), ISBN 0-201-52616-6 (1993).

  This easy-to-use manual enhances the fundamental concepts of calculus using Maple. Intended for use in a laboratory setting, a wide range of concepts are developed and investigated through a series of exploratory activities.

- *Calculus and Maple V* by R. McLaughlin, (Saunders College), ISBN 0-03-096778-3 (1993).

  A collection of calculus lab exercises on the uses of Maple.

- *Maple Computer Manual for Advanced Engineering Mathematics* by E. Kreyszig and E.J. Norminton (John Wiley & Sons, Inc.), ISBN 0-471-31126-X (1994).

  A manual which presents a selection of about 250 worked-out examples and nearly 800 problems, covering almost all fields discussed in Advanced Engineering Mathematics.

- *Precalculus Investigations using Maple V* by D.M. Mathews and K.E. Schwingendorf, (Harper Collings College), ISBN 0-673-99410-4 (1994).

  This book provides an exciting and effective alternative to the traditional lecture-listen format of precalculus, algebra and trigonometry courses. In the same spirit as the national calculus reform movement, this book is intended to supplement any standard text with twelve carefully structured interac-

tive learning environments for students in mathematics courses preceding calculus. Through the use of Maple, the laboratory sessions provide students with conceptual understanding of core topics which traditionally constitute precalculus courses and courses in college algebra and trigonometry.

- *Erste Schritte mit Maple* by W. Burkhardt, (Springer-Verlag), ISBN 3-540-56649-X (1994).

  Eine leichte Einführung mit Aufgaben und Lösungen. 130 Seiten.

  Inhalt: Einführung in die Benutzung, Termumformungen, Listen, Tabellen und Funktionen, Lösen von Gleichungen, Lineare Algebra und Gleichungssysteme, Grafiken, Analysis, Einfache Programme, Installation auf PCs, Befehlsübersicht, Lösungen der Aufgaben.

- *Maple V Release 2* by M. Kofler (Addison-Wesley Verlag Deutschland), ISBN 3-89319-635-8 (1994).

  Eine Einführung und Leitfaden für den Pratiker. 500 Seiten.

  Dieses Buch, dass ursprunglich für Mathematica geschrieben war, ist eine sehr umfassende Einführung in Release 2. Die Gliederung in zahlreiche überschaubare Kapital macht das Buch als Lehrbuch und als Nachschlagwerk geeignet. Systematisch und mit realitätsnahen Beispielen wird im weiteren Verlauf auf verschiedene Anwendungsgebiete von Maple eingegangen. Jedes Kapitel endet mit einer knappen Syntaxzusammenfassung. Sie werden dabei von der Definition eigener Funktionen, dem Vereinfachen mathematischer Ausdrücke, und dem Lösen von Gleichungen und Gleichungssystemen bis zu Integration, Differentialgleichungen, Fourier- und Laplace-Transformation geführt. Auch auf die ausgezeichneten Grafikfähigkeiten von Maple, das Programmieren in Maple und das Erstellen von Maple-Dokumenten mit LaTeX wird eingegangen.

  Dabei werden auch die Probleme mit Maple nicht verschwiegen. Ein eigenes Kapitel beschreibt typische Bedienungsfehler und ihre Ursachen. Eine Gegenüberstellung von Maple und Mathematica sowie eine kurze Beschreibung der Besonderheiten der Windows-Version von Maple runden das Buch ab.

- *Mathematik lernen mit Maple V* by W. Werner, (ELBI-Verlag GmbH), ISBN 3-929694-03-4 (1993).

  Ein Lehr- und Arbeitsbuch für das Grundstudium mit zahlreichen Beispielen und Übungen sowie einer Kurzeinführung in Maple V. Dieses Buch ist ein Lehrbuch für Mathematik. Die Beispiele werden mit Maple gelöst. Auch für die Übungsaufgaben kann Maple verwendet werden. Circa 500 Seiten.

  Inhalt: Mengen, Logik, Summen und Produkte, Trigonometrische Funktionen, Vektorrechnung, Natürliche Zahlen und vollständige Induktion, die reellen Zahlen, die komplexen Zahlen, Abbildungen und Funktionen, Folgen und Grenzwerte, Reihen, Potenzreihen, Grenzwert einer Funktion, Differenzierbare Funktionen, Taylorpolynom and Taylorreihe, Integralrechnung, Gewöhnliche Differentialgleichungen, Laplace-Transformation, Matrizen und lineare Gleichungssysteme, Determinenten, Ausgleichsrechnung, Eigenverte und Eigenvektoren, Einführung in Maple V.

- *Maple V Software für Mathematiker* by H. Gloggengieber (Markt & Technik), ISBN 3-87791-439-X (1993).

- *Computeralgebra mit Maple* by M. Kamminga-van Hulsen (Academic Service), ISBN 90-6233-923-9 (1993).

- *Self-Tutor for Computer Calculus Using Maple* by D.C.M. Burbulla and C.T.J. Dodson, (Prentice Hall Canada Inc.), ISBN 0-13-063926-5 (1993).

Maple V Release 3

Michael Monagan[1]

It's here, it's good, it's solid. It's available for the Macintosh, the PC (DOS + Windows), DEC Alpha (OSF/1), and most Unix platforms. It is the most thoroughly tested version of Maple yet, having gone through approximately 3 months of alpha-testing and 3 months of beta-testing.

Maple V Release 3 brings new capabilities and improvements in all areas of Maple. This includes enhancements to the user interface, worksheets, graphics, mathematics, and the Maple language and system. Release 3 also includes a new share library with many application worksheets and a demo directory with introductory tutorial and application worksheets. We've summarized the most important new facilities and changes below.

When you get Maple V Release 3, you'll receive a not-too-technical document called the "Release notes" which summarizes all the important changes in Release 2 *and* Release 3. A complete summary of the changes and improvements in Release 3 is provided on-line under `?updates,v5.3` and also as a Maple worksheet. Enjoy the new version of Maple and remember to take a look at what's in the new share library.

User Interface

Export to LaTeX

Perhaps the most useful new tool with Maple is this simple facility for outputting Maple worksheets as LaTeX files. It includes a faithful rendering of Maple mathematical expressions with line breaking for large expressions.

Keyword Search

Forgotten the name of a Maple command? Looking for Maple routines for solving problems in such and such an area? Then just type in the topic or keyword and Maple will search its on-line help database for all Maple commands that mention the word. A simple and very useful facility.

Subscripts and Assumed Variables

Subscripted variables now print as subscripts and assumed Greek variables print as Greek variables!

```
> assume(theta>0): a[i]*theta+a[i+1] = 1-theta;
```

$$a_i\theta^{\sim} + a_{i+1} = 1 - \theta^{\sim}$$

Toolbars for Windows

The Windows version of Maple V Release 3 includes special toolbars for the worksheet and plot windows. The user can change font sizes, change axes styles, change surface drawing styles, etc., with a simple mouse click.

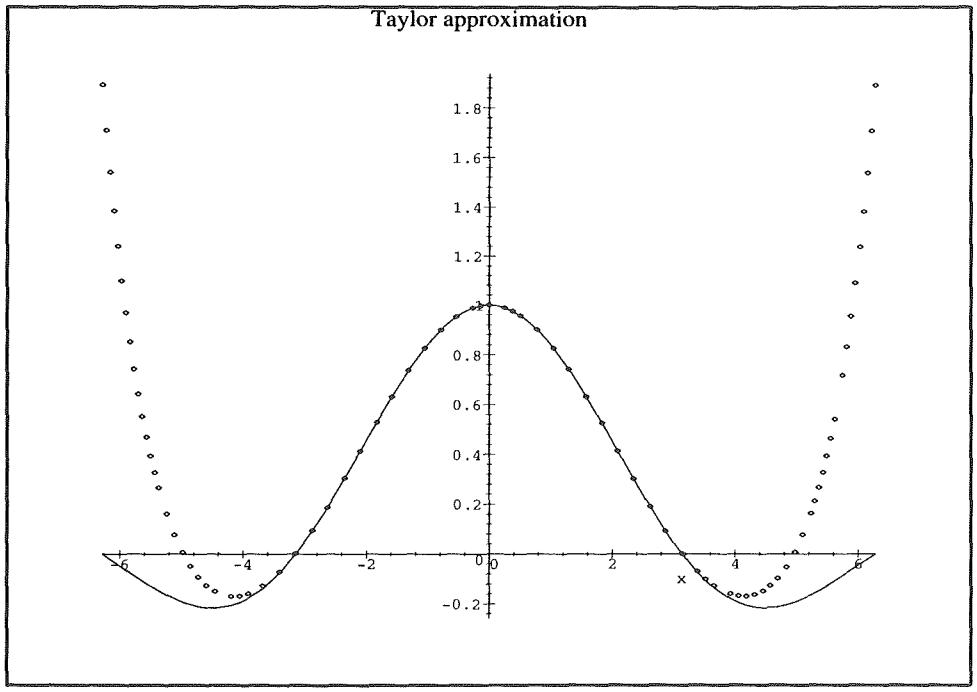
[1] Institute für Wissenschaftliches Rechnen, ETH-Zentrum, CH 8092 Zürich, Switzerland.

Graphics

New Plotting Options

There are now 5 new options for point styles in plots, namely, BOX, CROSS, DIAMOND, CIRCLE and POINT. There are several new line style options including DASHED, THICK, THIN, DOTDASHED, etc. Font options (type, style, and size) have been added for text regions in plots. For example:

```
> p1 := plot( sin(x)/x, x=-2*Pi..2*Pi ):
> approx := convert( series( sin(x)/x, x=0, 10 ), polynom ):
> p2 := plot( approx, x=-2*Pi..2*Pi, style=POINT, symbol=DIAMOND ):
> plots[display](p1,p2, title='Taylor approximation', titlefont=[TIMES,ROMAN,18]);
```



Plotting Discontinuities

A new option to `plot` has been added to search for discontinuities algebraically so that the resulting plot does not interpolate across discontinuities and singularities.

Mathematics

Radical Simplification

Good news! Problems with `sqrt` simplifications have been addressed. What were the problems? Maple simplified `sqrt(x^2)` to be `x` and `sqrt(x*y)` to be `sqrt(x)*sqrt(y)` automatically. These transformations are not always valid, for example when `x` and `y` are negative. Of course, simply turning off these simplifications is not satisfactory because users want these transformations to be done when provably correct, e.g. when `x` is known to be positive.

```
> radicals := [ sqrt(a^2), sqrt(4*a^2*y), (a^4*y)^(1/3) ];
```

$$radicals := [\sqrt{a^2}, 2\sqrt{a^2 y}, (a^4 y)^{1/3}]$$

Simplifying the power of a in these examples can be done by either using the `symbolic` option to `sqrt` or `simplify` or `combine`, or making appropriate assumptions. For example:

```
> simplify(radicals,radical,symbolic);
```

$$[a, 2a\sqrt{y}, a\,(ay)^{1/3}]$$

```
> assume(a>0);
> simplify(radicals,radical);
```

$$[a\tilde{}, 2a\tilde{}\sqrt{y}, a\tilde{}\,(a\tilde{}\,y)^{1/3}]$$

Various new facilities for simplifying nested radicals have been added. Simple nested square roots are de-nested where possible by `sqrt` and `simplify`, and the more powerful `radnormal` command is provided.

```
> sqrts := [ sqrt(2*I), sqrt(4+2*sqrt(3)), sqrt(2*(3-sqrt(2)-sqrt(3)+sqrt(6))) ];
```

$$sqrts := \left[1 + I, \sqrt{3} + 1, \sqrt{6 - 2\sqrt{2} - 2\sqrt{3} + 2\sqrt{6}}\right]$$

```
> readlib(radnormal)(sqrts);
```

$$\left[1 + I, \sqrt{3} + 1, -1 + \sqrt{3} + \frac{1}{3}\sqrt{6}\sqrt{3}\right]$$

New Integration Capabilities

Indefinite integration has been extended to handle classes of special functions (e.g. Bessel functions).

```
> Int( x^9*BesselY(3,x^2), x ) = int( x^9*BesselY(3,x^2), x );
```

$$\int x^9 \mathrm{BesselY}(3, x^2)\,dx = \frac{1}{2}x^8 \mathrm{BesselY}(4, x^2)$$

This second example shows a Hermite reduction:

```
> f := exp(-z)/z^2/(z^2+6*z+6)^2:
> Int(f,z) = int(f,z);
```

$$\int \frac{e^{(-z)}}{z^2 (z^2 + 6z + 6)^2}\,dz = -\frac{1}{12}\frac{e^{(-z)}\,(2 + 5z + z^2)}{(z^2 + 6z + 6)\,z} + \frac{1}{12}\mathrm{Ei}(1, z)$$

Also, there are extensions to the classes of definite integrals that Maple can handle:

```
> Int( t^(-1/3)*ln(t)*sqrt(1-t), t=0..1 );
```

$$\int_0^1 \frac{\ln(t)\sqrt{1-t}}{t^{1/3}}\,dt$$

```
> simplify(value(")");
```

$$\frac{3}{98}\left(-42\gamma + 7\pi\sqrt{3} - 63\ln(3) - 42\Psi\left(\frac{1}{6}\right)\right)\Gamma\left(\frac{2}{3}\right)\Gamma\left(\frac{2}{3}\right)/\sqrt{\pi}$$

New ODE Algorithms

Two new algorithms for solving linear ODE's of order higher than 2 have been implemented. Maple is using these methods to reduce the order of an ODE so that other methods for handling first and second order ODE's can be applied. In the example below the new algorithm is able to find the exponential solution.

```
> ode := (-2*x^2+x+n^2)*y(x)+(4*x^2-2*x-n^2)*diff(y(x),x)+
         (-3*x^2+x)*diff(y(x),x$2)+x^2*diff(y(x),x$3);
```

$$ode := (-2x^2+x+n^2)\,y(x) + (4x^2-2x-n^2)\left(\frac{\partial}{\partial x}y(x)\right) + (-3x^2+x)\left(\frac{\partial^2}{\partial x^2}y(x)\right) + x^2\left(\frac{\partial^3}{\partial x^3}y(x)\right)$$

```
> dsolve(ode,y(x));
```

$$y(x) = \_C1\, e^x + \_C2\, e^x \int \mathrm{BesselJ}(n, x)dx + \_C3\, e^x \int \mathrm{BesselY}(n, x)dx$$

In the special case n=1 these integrals can now be computed. Another new capability is the possibility of outputting the solution of an ODE as a basis of independent solutions as shown in this example.

```
> n := 1;
```

$$n := 1$$

```
> dsolve(ode,y(x),output=basis);
```

$$[e^x,\ e^x\mathrm{BesselJ}(0, x),\ e^x\mathrm{BesselY}(0, x)]$$

The DESol Data Structure

This is a new data structure to represent the solution of a differential equation when an exact analytical solution cannot be found. This permits the user to manipulate a differential equation symbolically, e.g. by differentiating it, computing a series, etc. Here is an example:

```
> de := DESol( diff(x(t),t,t) = g/l*sin(x(t)), x(t), x(0)=0 );
```

$$de := \mathrm{DESol}\left(\left\{\left(\frac{\partial^2}{\partial t^2}x(t)\right) - \frac{g\sin(x(t))}{l}\right\},\ \{x(t)\},\ \{x(0) = 0\} \right)$$

```
> series(de,t=0);
```

$$D(x)(0)t + \frac{1}{6}\frac{gD(x)(0)}{l}t^3 + \frac{1}{120}\frac{gD(x)(0)\,(-D(x)(0)^2+g)}{l^2}t^5 + O(t^6)$$

Multivariate Factorization over GF(q)

Polynomial factorization over finite fields has been extended to handle multivariate polynomials. In this example we factor a polynomial in x and y over GF(9). The finite field is represented by polynomials over GF(3) modulo the given polynomial below. This representation is quite general and permits very large finite fields.

```
> alias(alpha=RootOf(x^2+2*x+2)):
> f := x^3+alpha*x^2*y+alpha^2*x*y^2+(2*alpha+1)*y^3;
```

$$f := x^3 + \alpha x^2 y + \alpha^2 x y^2 + (2\alpha+1)\,y^3$$

```
> Factor(f) mod 3;
```

$$(y + (\alpha+2)\,x)\ (y+\alpha x)\ (y+2\alpha x)$$

New Statistics Packages

The `stats` package has been completely redesigned. The functionality is broken down into subpackages which are listed when the `stats` package is loaded.

```
> with(stats);
```

[*describe*, *fit*, *importdata*, *random*, *statevalf*, *statplots*, *transform*]

The basic data structure of the stats package is the statistical list. It is a list of values which may contain ranges and weighted values. For example:

```
> s11 := [1,3,8,7,8,9,3,10,5,6,8];
```

$$s11 := [1, 3, 8, 7, 8, 9, 3, 10, 5, 6, 8]$$

```
> s12 := [1,3..5,7,6..9, Weight(2,5), 6, Weight(7..10,2)];
```

$$s12 := [1, 3 .. 5, 7, 6 .. 9, \text{Weight}(2, 5), 6, \text{Weight}(7 .. 10, 2)]$$

```
> describe[mean](s11), describe[mean](s12);
```

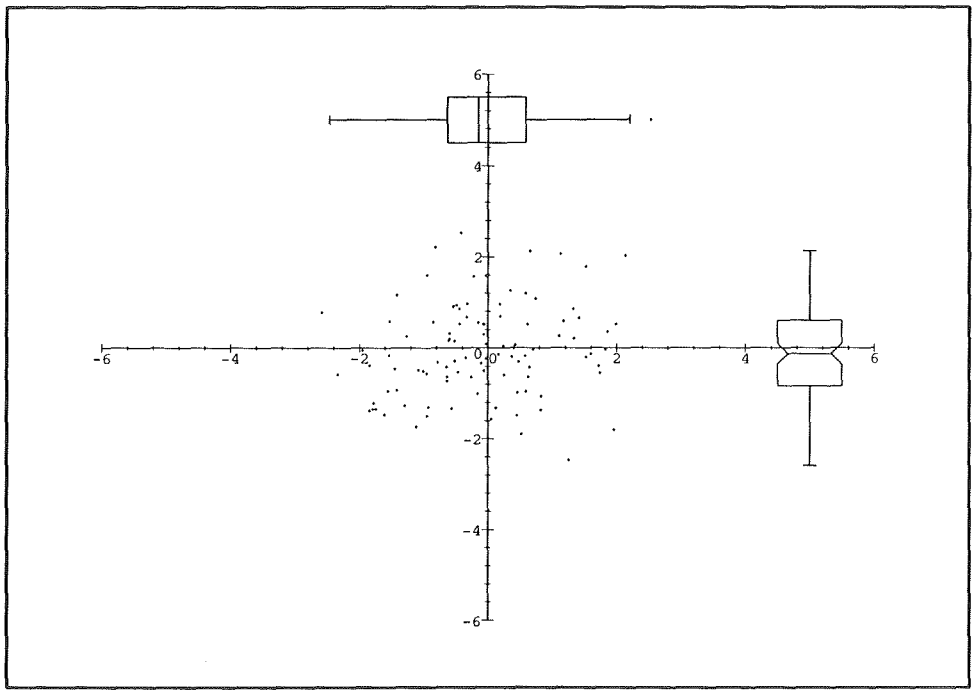
$$\frac{68}{11}, \frac{35}{8}$$

There are many new distributions supported, facilities for computing random variates, data fitting routines, and many utility plotting routines. Here we plot 100 random points from two normal distributions with mean 0 and standard deviation 1 against each other.

```
> with(statplots);
```

[*boxplot*, *histogram*, *notchedbox*, *quantile*, *quantile2*, *scatter1d*, *scatter2d*, *symmetry*, *xscale*, *xshift*, *xyexchange*]

```
> N01 := random[normald[0,1]]: X := [N01(100)]: Y := [N01(100)]:
> splot := scatter2d(X,Y): bplot := boxplot[5](Y): nplot := notchedbox[5](X):
> plots[display]( splot,xyexchange(bplot),nplot, view=[-6..6,-6..6] );
```



Language and System

Protected Names

A protection facility has been added to prevent the user from accidentally assigning to Maple system variables. For example, the names `list` and `lhs` are now protected. Attempting to assign to them will result in an error. For example:

```
> lhs := x^2-1;

Error, attempting to assign to `lhs` which is protected
```

Names may be protected with the `protect` function and unprotected with the `unprotect` function.

Global Statement

Procedures may now include an optional `global` statement which follows the `local` statement.

```
> proc(n) local y; global A;
    y := 1; for i to n do y := x*y; A[i] := y od;
  end;

Warning, `i` is implicitly declared local

proc(n)
local y,i;
global A;
  y := 1; for i to n do y := x*y; A[i] := y od
end
```

In this procedure, `n` is explicitly declared as a parameter, `y` is explicitly declared as a local variable, and `A` is explicitly declared as a global variable. The variables `i`, and `x` are not explicitly declared. In previous versions of Maple the variables `i` and `x` would be implicitly declared global variables. In this example, this is almost certainly a mistake as `i` should be a local variable. In the new version of Maple we see that `i` is implicitly declared local and the user is told about this implicit declaration. The rules for what is implicitly declared local are:

(i) the variable appears on the left hand side of an assignment statement, or:

(ii) it is used as an index to a `for` loop or `seq` statement.

Fortran Command

The `fortran` function now accepts an optional argument `mode = <modetype>` where the mode type must be one of `single` (the default), `double`, `complex`, `generic`. This controls how Maple function names are translated. Various other improvements have been made including outputting of vectors and matrices in row major order. For example:

```
> A := array(symmetric,1..2,1..2,[(1,1)=ln(x),(2,1)=sin(x)]):
> fortran(A,mode=generic,optimized);

      t2 = sin(x)
      A(1,1) = log(x)
      A(1,2) = t2
      A(2,1) = t2
      A(2,2) = undefined
```

New Share Library

The new version of the share library that comes with Release 3 contains:

1. About 25 applications packages and 75 applications routines;
2. Over 50 applications worksheets and 20 educational worksheets in the sciences, engineering and mathematics;
3. About 25 additional PostScript, TEX and LATEX documentation files.

In this version of the share library we have divided the share library up into the following subject areas to make it easier for the users to find what they are looking for and provided a contents listing for each area with the given `?` command:

Algebra	`?share,algebra`
Analysis (Calculus)	`?share,analysis` or `?share,calculus`
Automatic Differentiation	`?share,autodiff`
Combinatorics	`?share,combinat`
Engineering	`?share,engineer`
Graphics and Geometry	`?share,graphics` or `?share,geometry`
Linear Algebra	`?share,linalg`
Number Theory	`?share,numtheory`
Numerics	`?share,numerics`
Programming	`?share,programming`
Science	`?share,science`
Statistics	`?share,statistics`
System Tools	`?share,system`

Also new is an alphabetical index of the contents (see `?share,index`). Additionally, the new `readshare` command simplifies the loading of routines and packages from the share library. For example, to load the `gfun` package that is in the calculus directory of the share library:

```
> with(share):
See ?share and ?share,contents for information about the share library
> readshare(gfun,calculus);
```

gfun

The Maple Share (Applications) Library

The Maple *share* library is a means for the distribution of Maple routines, packages, worksheets, and other Maple materials written by Maple users to the Maple community. The share library is distributed with each new version of Maple (professional version only, not the student version) but we also maintain an electronic version (updated quarterly) which is available via anonymous ftp at the following sites

Internet number	Symbolic address	Machine location	directory
129.132.101.33	neptune.inf.ethz.ch	Switzerland	maple
129.97.140.58	daisy.uwaterloo.ca	Canada	maple
192.93.2.54	ftp.inria.fr	France	lang/maple
192.16.184.250	canb.can.nl	Holland	pub/maple-ftplib

Please read the file `README` there. For users who do not have access to the internet, the share library is also distributed via electronic mail from CAN in the Netherlands. To access it, send a mail message containing the text "send info" to `maple-netlib@can.nl` . Instructions will be mailed back to you.

What's New in Maple V Release 3?

The share library distributed with Maple V Release 3 contains approximately 50 Maple worksheets, 75 routines, and 25 packages, over double that of Release 2. We have included a special command `readshare` to make it easier to load routines and packages from the share library. Also, `?share[index]` gives you an alphabetical index of the facilities into the share library for quick inspection, and `?share[contents]` points you to more detailed topic indexes. For example, `?share[statistics]` lists all facilities in statistics.

Where to Send Contributions

See `?share[contrib]` for instructions for submitting Maple code or worksheets to the share library. Submissions must be made by electronic mail to make it feasible for us to assist authors in testing, preparing documentation, and maintaining the code. Questions about contributions should be sent to Dr. Michael Monagan at `monagan@inf.ethz.ch` .

New Share Library Contributions

The following worksheets are available for Release 2 and Release 3.

- doubpend.ms – uses the ODE package to approximate the solution to the differential equation associated with a double pendulum, i.e. a pendulum attached to a pendulum. The motion of the double pendulum is then animated.
- education (science) – These seven worksheets show different capabilities of Maple in solving seven problems in science at the educational level. For further details, see the article by Scott et al. in this issue.

  1. bohratom.ms (chemistry): shows Maple solving three non-linear equations symbolically arising from semi-classical mechanics, namely the Bohr theory applied to the Hydrogen atom.

2. chemeqn.ms (chemistry): uses Maple's `isolve` command to balance the coefficients in a chemical reaction.
3. heatcap.ms (statistical mechanics): obtains the mean energy and heat capacity of an Einstein solid. Maple is used to compute a symbolic infinite sum and compute symbolic limits.
4. Maxgas.ms (statistical mechanics): uses Maple to find the most probable speed of the Maxwell-Boltzmann distribution. This involves differentiation and solving a simple non-linear equation.
5. planck.ms (statistical mechanics): uses Maple to compute a definite integral, the Stefan-Boltzmann Law (of Blackbody radiation).
6. quantopt.ms (quantum optics): uses Maple to symbolically integrate a triple integral and simplify the resulting formula.
7. wheatsto.ms (basic electronics): uses Maple to solve six linear equations (derived from applying Kirchoff's laws to the "Wheatstone Bridge" electrical circuit) *symbolically* for the currents which are functions of the resistances $R1$, $R2$, $R3$, $R4$, $R5$.

- flash.ms – flash calculations are used to determine the phase condition of a mixture at a specified temperature and pressure. Such computations lie at the heart of many process engineering calculations and are of central importance in many other applications as well (oil reservoir simulation for example).
- group.ms – an application of Maple's group package to answering various questions about the small Rubik's cube.
- lapack.ms – on the front cover of the LAPACK User's Guide is a 6 by 6 matrix whose entries are the letters L A P A C K appearing in a particular pattern. This worksheet shows how we can use Maple's symbolic linear algebra tools to find out exactly what is so special about this matrix.
- logistic.ms – solves the logistics law of population growth $P' = a\,P - b\,P^2$. Given 3 populations $P(t0)$, $P(t0+\delta)$, and $P(t0+2*\delta)$ Maple can solve for a and b.
- logmap.ms – explores the period-doubling bifurcation sequence of the discrete logistic map. Although numerical tools can be used, we believe it is useful to examine the polynomial algebra here in some detail.
- McConnel.ms – a Maple worksheet on how to use matrix algebra, in particular matrix exponentials, for the symbolic solution of a system of first order differential equations in biochemistry, namely the McConnell equations. The McConnell equations are used to analyze data obtained from solute-cell transport experiments using the nuclear magnetic resonance (NMR) technique.
- parfrac.ms – shows how to use Maple to perform each step of a partial fraction decomposition for rational function integration.
- pendulum.ms – models the motion of a pendulum in a fluid as a second order ODE, solves the ODE analytically using Maple, and then plots the motion for different fluid resistances.
- phase.ms – simple Thermodynamic Calculations including bubble and dew point determination, creating and plotting phase diagrams, flash calculations, etc.
- shottraj.ms – uses Maple to solve symbolically and plot the solution of a second order ODE which describes the motion of a bullet shot straight up into the air.

The following Maple routines and packages are available; the superscript indicates for which release of Maple V they are available.

- pade2[1,2,3] – computes a generalized Pade approximation of $f_1(x), \ldots, f_n(x)$ at $x=p$.

- intpak[1,2,3] – an interval arithmetic package.
- IntSolve[1,2,3] – an integral equation solver.
- fft, fht[2,3] – Maple routines for the calculation of the Fast Fourier and Hartley transforms.
- fields[1,2,3] – let *N* be a field extension of *L*, a field extension of *K*, such that *N* is finitely generated over *K*. This package uses Grobner basis methods to calculate the transcendence degree of *N*|*L* and the degree [*N*:*L*] if the field extension is algebraic, and related questions.
- sffge[1,2,3] – reduces a matrix of polynomials to upper triangular form and (optionally) computes the determinant. This routine is designed specially for sparse matrices.
- genus[2,3] – (an update to the integral basis package) for computing the genus of an algebraic curve.
- trans[1,2,3] – update includes special numerical versions of the approximation algorithms to use the performance of the hardware floating point arithmetic.
- ratinterp[1,2,3] – computes the rational function which interpolates given data points.
- kinetics[1,2,3] – Maple procedures for determining the system of differential equations, the associated conservation laws, and some of the species that have a zero steady state of a reaction scheme. This update includes a better algorithm plus notational changes.
- MatPade[2,3] – Maple procedures to compute matrix-type Pade approximants.

Tips for Maple Users

Michael B. Monagan[1]

In this month's column, we answer three common programming questions that users have often asked, and we begin a discussion on simplification and the problem of zero recognition which we will continue in the next issues of the MTN. Testing whether an expression is equal to zero, and simplification are perhaps the two most important issues facing all computer algebra systems. They are complicated issues both from a system design point of view, and from the algorithmic point of view. Let us begin with the programming questions.

Unapply: the Inverse of Application

A user asked why the following didn't work:

```
> f := x*sin(x);
```

$$f := \sin(x)\, x$$

```
> diff(f,x);
```

$$x\cos(x) + \sin(x)$$

```
> g := x -> ";
```

$$g := x \rightarrow "$$

```
> g(1);
>
```

That is, the output of $g(1)$ is "nothing". What the user clearly wanted to do was to assign to g the function `x -> x*cos(x)+sin(x)`. So why didn't Maple do the obvious thing?! Well, consider this program:

```
> diff(f,x);
```

$$x\cos(x) + \sin(x)$$

```
> h := proc(g) int(g,x); " end;
```

Now, does the " in the h procedure refer to the derivative computed in the session? Or does it refer to the previous value `int(g,x);` in the procedure? Of course it refers to the value of `int(g,x);` in the procedure. When you input a procedure either using the `proc ... end` notation or the arrow notation, the body of the procedure is NOT executed. It is only executed when the procedure is called. Hence when the `g` procedure is called, since " has no value, `NULL` is returned.

That leaves us with the user's problem: "How can I create a procedure from a formula?" The answer is to use the `unapply` function as follows:

```
> diff(f,x);
```

$$x\cos(x) + \sin(x)$$

```
> g := unapply(",x);
```

$$g := x \rightarrow x\cos(x) + \sin(x)$$

```
> g(1);
```

$$\cos(1) + \sin(1)$$

[1] Institute für Wissenschaftliches Rechnen, ETH-Zentrum, CH 8092 Zürich, Switzerland.

Scoping Rules for Procedures

A user recently asked why the following program didn't work:

```
> try1 := proc(x:list) local v;
    v := x[nops(x)];
    map(y->y/v, x);
  end:
```

Note: the construct `y->y/v` is Maple shorthand for the procedure `proc(y) y/v end`. What is the program meant to compute? The input to the program is a list of values x. It assigns v the last value in the list x. The `map` statement is meant to return a new list where the values in the list x have been divided by v. For example, if the input is:

```
> x := [16,2,4,8];
```

$$x := [16, 2, 4, 8]$$

then the `try1` program is supposed to return:

$$\left[2, \frac{1}{4}, \frac{1}{2}, 1\right]$$

where all entries have been divided by 8. Instead, the program returns:

```
> try1(x);
```

$$\left[16\frac{1}{v}, 2\frac{1}{v}, 4\frac{1}{v}, 8\frac{1}{v}\right]$$

The problem is that the variable v in the procedure `y->y/v` does not refer to the local variable v in the `try1` procedure. What does it refer to? In Maple, because the v in `y->y/v` is neither a local variable (there are no local variables) nor a parameter (y is the only parameter), it refers to the global variable v (the one in the Maple session). In some other languages, e.g. Pascal, the v variable would refer to the local v variable in the `try1` procedure. Note, the Maple people are right now considering changing Maple to be like Pascal on this issue of "scoping rules for variables". Here are several ways to solve the problem now:

```
> proc(x:list) local v,y;
    v := x[nops(x)];
    [seq(y/v, y=x)];
  end
```

This solution solves the problem by avoiding the use of a nested procedure. It is therefore the fastest solution. Note, in the `seq` function here, the variable y iterates over the elements of the list x. Here is another solution:

```
> proc(x:list) local v;
    v := x[nops(x)];
    map((y,v)->y/v, x,v);
  end
```

This solution avoids the problem by passing the value v as an additional parameter to the `map` function. Yet another solution:

```
> proc(x:list) local v,y;
    v := x[nops(x)];
    map(unapply(y/v,y), x);
  end
```

What does `unapply(y/v)` do? First, the argument y/v is evaluated. In our example, the result is the formula $y/8$. Now the `unapply` command creates a procedure from the formula $y/8$ with y being a parameter:

```
> unapply(y/8,y);
```

$$y \rightarrow \frac{1}{8}y$$

Thus this solution solves the problem by creating a procedure with the value of v in the procedure. The final solution is a general way to simulate nested scoping in Maple. It relies on the ability to be able create new procedures by substituting for dummy names into a procedure. We don't claim that this is an elegant solution! However ...

```
> proc(x:list) local v,f;
    v := x[nops(x)];
    f := subs(DUMMY='v', y->y/DUMMY);
    map(f,x);
  end
```

Instead of using the procedure `y->y/v`, the desired procedure is created by substituting the global variable `DUMMY` for the name of the local variable v into the procedure. It is not necessary to put quotes around the local variable v in this example. Without quotes, the value of v (8 in our example) would be substituted for `DUMMY`. We would end up with the same solution as the `unapply` solution.

Programming Operators and Rules

A user was trying to define standard statistical operators for expectancy, variance, covariance and asked how to implement `Cov`, a covariance operator in Maple. Note, it is not necessary that you know any statistics or even care about statistics to learn from this example. Let a be a constant, and X, Y, Z be random variables. We are given the following properties below for the `Cov` operator:

1. $\mathrm{Cov}(a,X) = 0$
2. $\mathrm{Cov}(X,X) = \mathrm{Var}(X)$ where Var is the variance operator
3. $\mathrm{Cov}(a\,X,Y) = a\,\mathrm{Cov}(X,Y)$ (linearity)
4. $\mathrm{Cov}(X+Y,Z) = \mathrm{Cov}(X,Z) + \mathrm{Cov}(Y,Z)$ (distributivity)
5. $\mathrm{Cov}(X,Y) = \mathrm{Cov}(Y,X)$ (commutativity)

and we want to apply them to "simplify" an expression involving these operators. What do we mean by "simplify" here? We mean apply the given rules in such a way that the input expression is transformed into a standard form. For example, on input of:

$$\mathrm{Cov}(x,x+2y+z)$$

after applying the properties from left to right, we want to obtain the result:

$$\mathrm{Var}(x)+2\mathrm{Cov}(x,y)+\mathrm{Cov}(x,z)$$

In the previous issue of the MTN, we showed how to program simplifications like this. Our code makes use of the `map` and `select` commands. To simplify our Maple code, I have written this utility function `HasConstants:`

```
> # This boolean procedure determines if a product has any constants
  # If it does, it assigns the parameter C the product of those constants

  HasConstants := proc(x:'*', C:name) local c;
    c := select(type,x,constant);
    if c <> 1 then C := c; true else false fi;
  end:

  Cov := proc(X,Y) local C;
    if type(X,constant) then 0
    elif type(Y,constant) then 0
    elif X = Y then Var(X)
    elif type(X,'+') then map(Cov,X,Y) # distributivity
    elif type(Y,'+') then map(Cov,Y,X) # distributivity
    elif type(X,'*') and HasConstants(X,'C') then C*Cov(X/C,Y) # linearity
    elif type(Y,'*') and HasConstants(Y,'C') then C*Cov(X,Y/C) # linearity
    elif sort([X,Y]) = [Y,X] then Cov(Y,X) # commutativity
    else 'Cov'(X,Y)
    fi
  end:
```

Observe that each rule or property has become one or two conditions followed by an action. This is coded as a single Maple `if` statement. If no rule is applicable the *unevaluated* expression `'Cov'(X,Y)` is returned. The quotes here are to prevent an infinite recursion.

Note also that the property distributivity has been implemented using the `map` command. If X is of sum of terms, i.e. $X = X_1 + X_2 + ... + X_n$, then:

$$\text{map}(Cov, X, Y) \rightarrow \text{Cov}(X_1,Y) + \text{Cov}(X_2,Y) + ... + \text{Cov}(X_n,Y)$$

Zero Recognition and Simplification

What is Maple's goal when I input an expression? The answer in one word is "simplify" it. Everything that Maple does can be viewed in this light. I.e, Maple can be viewed as a formula transformer where the user is happy when the output is *simpler* than the input. Even an integral can be viewed as a simplification. For example, consider:

```
> int( x^2/(x^2-a^2), x );
```

$$x + \frac{1}{2}a\ln(x-a) - \frac{1}{2}a\ln(x+a)$$

In what sense is this output "simplified"? It is simplified in the sense that the integral sign ($\int$) has been removed. Suppose we try to solve the system of equations:

```
> solve({x^3+y^3=1,x^4-y^4=1},{x,y});
```

Maple returns this:

$$\{y=0,\ x=1\},\ \{y=\%1,\ x=7/4\,\%1+5/4\,\%1^4+3\,\%1^5-9/4\,\%1^2-7/4\,\%1^3+2\}$$
$$\%1 \quad := \quad \text{RootOf}(4\_Z^6-6\_Z^3+4+3\_Z^5+3\_Z)$$

In what sense has this been simplified? The variables x,y have been isolated to the left hand sides of the equations. Notice that Maple did not completely solve the equations. The solutions have been left in an implicit form as the roots of a degree 6 polynomial. Consider the following table of expressions (table 1). All these expressions equal zero. Well, strictly speaking the ones containing integrals equate to constants. And the ones containing square roots equal zero *assuming* we are choosing the positive root or the "principle branch". Let's input them into Maple and see what happens:

1.	$\sqrt{4}-2$	6.	$\sqrt{6}-\sqrt{2}\sqrt{3}$
2.	$\sqrt{3+4i}-2-i$	7.	$\sqrt{4+2\sqrt{3}}-\sqrt{3}-1$
3.	$x^{3/2}-x\sqrt{x}$	8.	$\sqrt{x+1}^{3}-x\sqrt{x+1}-\sqrt{x+1}$
4.	$\sin(-x)+\sin(x)$	9.	$e^{-x}-\frac{1}{e^{x}}$
5.	$x-\ln(1+x)-\int\frac{dx}{1+x}$	10.	$\ln(x+\sqrt{x^3+1})-\int\frac{3x^2+2\sqrt{x^3+1}}{2x^3+2\sqrt{x^3+1}\,x+2}dx$

Table 1

```
> e1 := sqrt(4)-2;
```

$$e1 := 0$$

```
> e2 := sqrt(3+4*I)-2-I;
```

$$e2 := 0$$

```
> e3 := x^(3/2)-x*sqrt(x);
```

$$e3 := 0$$

```
> e4 := sin(-x)+sin(x);
```

$$e4 := 0$$

```
> e5 := x-ln(1+x)-int(x/(x+1),x);
```

$$e5 := 0$$

```
> e6 := sqrt(6)-sqrt(2)*sqrt(3);
```

$$e6 := 6^{1/2}-2^{1/2}3^{1/2}$$

```
> e7 := sqrt(4+2*sqrt(3))-sqrt(3)-1;
```

$$e7 := (4+2\,3^{1/2})^{1/2}-3^{1/2}-1$$

```
> e8 := sqrt(x+1)^3-x*sqrt(x+1)-sqrt(x+1);
```

$$e8 := (1+x)^{3/2}-x(1+x)^{1/2}-(1+x)^{1/2}$$

```
> e9 := exp(-x)-1/exp(x);
```

$$e9 := \exp(-x)-\frac{1}{\exp(x)}$$

```
> r := sqrt(x^3+1):
> e10 := ln(x+r)-int( (3*x^2+2*r)/(2*x^3+2*r*x+2), x );
```

$$e10 := \ln(x+(x^3+1)^{1/2})-\int\frac{3x^2+2(x^3+1)^{1/2}}{2x^3+2(x^3+1)^{1/2}x+2}dx$$

Maple determines that the first 5 are zero. We say that the first 5 were *automatically* simplified to 0. Why didn't Maple simplify the latter 5 to 0? Can Maple simplify the latter 5 to 0? There exist several commands in Maple to simplify formulae. The `simplify` command is the command most users will be familiar with.

The `expand` and `normal` commands also simplify formulae. The `normal` command, possibly the most important command in Maple, provides a "normal form" for "rational expressions". What is a normal form? What are rational expressions? We'll get to that later. Let's try the `simplify` command:

```
> for i from 6 to 10 do simplify( e.i ) od;
```

$$0$$

$$2^{1/2}\,(2+3^{1/2})^{1/2}-3^{1/2}-1$$

$$0$$

$$0$$

$$\ln\left(x+(1+x)^{1/2}\,(x^2-x+1)^{1/2}\right)-\int 1/2\,\frac{3x^2+2\,(x^3+1)^{1/2}}{x^3+(x^3+1)^{1/2}x+1}\,dx$$

It would appear that `simplify` is not powerful enough to recognize that the nested square root expression is zero. Actually, there is a function in the Maple library called `radnormal` which can de-nest radicals, and hence simplify this to zero. The integral `e10` is a case where Maple really just can't do it.

In this discussion, we are going to view all of Maple as a simplifier. Think of the integration command `int` as a simplifier. It is no different from the `sqrt` command or the `sin` command or the `Cov` operator that we implemented previously. When Maple gets an integral as an input, it tries to simplify the integral. If it can get rid of the integral sign, then it has succeeded.

We want also to emphasize the issue of *zero recognition*. This is crucial because if we fail to recognize a zero, we may erroneously divide by zero or conclude that a system of equations is unsolvable. The following examples will help motivate the discussion further.

```
> int( (sqrt(6)-sqrt(2)*sqrt(3))*exp(x^3), x );
```

$$\int (6^{1/2}-2^{1/2}3^{1/2})\,\exp(x^3)\,dx$$

```
> simplify(");
```

$$0$$

Maple could not initially compute the integral because it did not realize that the integrand is equal to 0. It has a *hidden* zero. But it can simplify the integrand to 0 using the `simplify` command. A second example:

```
> eqns := {x=0,x=sqrt(6)-sqrt(2)*sqrt(3)};
```

$$eqns := \{x=0,\ x=6^{1/2}-2^{1/2}3^{1/2}\}$$

```
> solve(eqns,{x});
>
```

In this example, we fooled Maple so that it could not find the x=0 solution. Once we can hide a zero, it is not hard to force Maple to divide by it.

```
> 1/((sqrt(6)-sqrt(2)*sqrt(3))+x);
```

$$\frac{1}{6^{1/2}-2^{1/2}3^{1/2}+x}$$

```
> series(",x);
```

$$\frac{1}{\%1}-\frac{1}{\%1^2}x+\frac{1}{\%1^3}x^2-\frac{1}{\%1^4}x^3+\frac{1}{\%1^5}x^4-\frac{1}{\%1^6}x^5+\text{'O}(x^6)$$

$$\%1 := 6^{1/2}-2^{1/2}3^{1/2}$$

```
> simplify(");
Error, (in radsimp) division by zero
```

It is apparent from these three examples that none of the commands `int`, `solve` nor `series` use the `simplify` command. If they had, they would have eliminated the hidden zero $\sqrt{6}-\sqrt{2}\sqrt{3}$ expression. What do they use to simplify their inputs? I hope the reader is intrigued at this point and will learn as we answer the following questions in the coming issues of MTN:

1. What simplifications are automatic?
2. What simplifications are provided by `expand`, `normal` and `simplify`?
3. When can routines like `solve` and `int` fail or give wrong answers because they fail to recognize a zero?
4. What assumptions are made when simplifications are done?
5. Can I, and if so how, extend Maple's simplification knowledge?

Simplification and the `assume` facility

Robert M. Corless and Michael B. Monagan[1]

Simplification is difficult. For computer algebra systems, the problem is to make simplifications which are both 'correct' and 'useful'. The good news is that Maple is becoming more 'correct'. It no longer makes transformations as carelessly as it used to. For instance, Maple V Release 3 does not *automatically* simplify

$$\sqrt{x^2} \to x \quad \text{or} \quad \sqrt{xy} \to \sqrt{x}\sqrt{y}.$$

These incorrect transformations were made automatically in previous versions of Maple. That is, Maple made these simplifications as soon as it saw these expressions, without any instructions from the user to do so. This gave rise to many bugs in the cases where x and y can be negative.

More than just automatic simplification has since been improved. The commands `simplify` and `combine` are now more careful about applying these and other transformations.

The bad news is that Maple is now less helpful. Preventing incorrect simplifications is good. But this change to Maple means that *correct* simplifications have also been 'turned off' and Maple is now 'scared' to return answers because they might be wrong.

This is where the `assume` facility comes in. It can be used to tell Maple that $x > 0$ so that the above transformations can be 'legally' made. Maple's `assume` facility is based on the ideas and algorithms described in [6, 7]. Other systems (such as Macsyma) also have `assume` facilities. In [1], a description is given of an implementation of assumptions and simplifications of the absolute value function in Reduce. This present article is meant as an informal introduction to the use and philosophy of `assume`, and not as a detailed technical reference.

Why do we need an `assume` facility?

The `assume` facility is meant to address several issues. The first is that without assumptions on the values of certain parameters in your equations and formulae, Maple can make no rigorous progress in simplifying or evaluating functions. For example, without assuming that $x > 0$, we cannot correctly simplify the expression $\sqrt{x^2}$ to x. (In this paper, we will assume all variables are real unless stated otherwise. Think of this as a 'default assumption'.) In the 'bad old days' (Maple V Release 2 and earlier) Maple would *over*simplify $\sqrt{x^2}$ to x. This led to dreadful bugs.

Consider Table 1, which compares Maple with Reduce and Mathematica on some problems containing the parameter a. We see that the first two entries in the Maple V Release 2 column are wrong, for at least some values of a. Entries three and four are correct. Further, the third one tells the user that the answer depends on the sign of a. The fourth one suggests that Maple can almost get the answer. The last example is of course also correct, but it is not helpful. Does this means that Maple just can't compute this integral?

Mathematica's behavior is essentially similar. The first two answers are, like Maple's last answer, correct but not helpful. The square root has not been simplified. The last two answers, like Maple's first two answers, are wrong for some values of a.

[1] This article was written during a visit of RMC to ETH Zürich. This visit was supported by G. H. Gonnet, and indirectly made possible by the Organizing Committee of the IASTE Summer School "Let's Face Chaos through Non-linear Dynamics" at the University of Ljubljana.

Problem	Reduce	Maple V R 2	Mathematica 2.2	Maple V R 3
$\sqrt{a^2}$	a	a	$\sqrt{a^2}$	$\sqrt{a^2}$
$\sqrt{(1+a)^2yz}$	$(1+a)\sqrt{y}\sqrt{z}$	$(1+a)\sqrt{y}\sqrt{z}$	$\sqrt{(a+1)^2yz}$	$\sqrt{(a+1)^2yz}$
$\lim_{x\to\infty} ax$	NA	signum$(a)\infty$	indeterminate	signum$(a)\infty$
$\int_0^\infty e^{-at}dt$	NA	$\frac{1}{a}+\lim_{t\to\infty} e^{-at}$	$\frac{1}{a}$	$\frac{1}{a}+\lim_{t\to\infty} e^{-at}$
$\int_0^\infty e^{-at}\sqrt[3]{t}\,dt$	NA	$\int_0^\infty e^{-at}\sqrt[3]{t}\,dt$	$\frac{\Gamma[4/3]}{a^{4/3}}$	$\int_0^\infty e^{-at}\sqrt[3]{t}\,dt$

Table 1

The incorrect square root simplifications have been removed in Maple V Release 3, but this means in some sense that Maple is now less helpful than it was before, because it is too 'scared' to make simplifications even when they might be correct. In order for the system to make the desired simplifications, we need to `assume` that, for example, $a \geq 0$, or $a > 1$, or that $a > 0$.

The second reason for including an assume facility is that Maple can make even *more* simplifications than it did in the 'bad old days', if it knows that certain quantities are positive (for example). So in some sense, it is an 'efficiency' issue as well as a 'correctness' issue. The sooner simplifications can be made, the more likely it is the system will return with a useful solution. One can also expect that the computation will proceed more quickly.

Finally, an assume facility is needed because for a great many analysis problems, as opposed to algebra problems, we need to include geometric information to get a solution at all.

How do we use Assume?

Using `assume` is simple. We issue one of the commands

```
> assume(<inequality>);
> assume(<variable>,<property>);
```

if we wish to make new assumptions about a variable. We issue one of the commands

```
> additionally(<inequality>);
> additionally(<variable>,<property>);
```

if we wish to add to Maple's knowledge of the properties of a variable without disturbing what Maple already knows. How does this information help Maple? Consider the following examples.

```
> assume(a>0);
> assume(n,integer);
> additionally(n>0);


> Re(a);
```

$$a\tilde{}$$

Notice that Maple prints a with a tilde (or 'twiddle'). This is meant to remind you that assumptions about a have been made. We see below that Maple is able to make simple deductions about the signs of expressions from the assumptions we have made.

```
> signum(a);
```

$$1$$

```
> frac(n);
```

$$0$$

```
> signum(n*(n+1));
```

$$1$$

Maple's `if` statement does not know about the `assume` facility.

```
> if n>0 then n>0 else n<=0 fi;
Error, cannot evaluate boolean
```

The `about` command can be used to tell us what information has been assumed.

```
> about(a);
Originally a, renamed a~ : is assumed to be:
  RealRange(Open(0),infinity)
```

The following Maple library functions know about assumptions that have been made.

signum, csgn, abs	Im, Re	frac, trunc, ceil, floor

These routines directly query the `assume` facility. They use the low-level routine `isgiven` to query the assume facility, and encode some other 'smarts'. This routine is less powerful than the similar routine `is`, but is guaranteed to return an answer in a bounded amount of time.

As an example, if `signum` is given `a` as an argument, it will ask if `isgiven(a,positive)`, and also if `isgiven(a,negative)`. It also uses the rule

$$signum(x_1 + ... + x_n) = 1 \quad if \quad signum(x_i) = 1$$

Other Maple commands like `sqrt`, `int` and `limit` make use of the `assume` facility indirectly by using these functions. For example, we have

```
> assume(a>0);
> sqrt(a^2);
```

$$a\tilde{}$$

```
> sqrt((1+a)^2*x*y);
```

$$(1 + a\tilde{})\sqrt{xy}$$

```
> limit(a*x,x=infinity);
```

$$\infty$$

```
> int( exp(-a*t), t=0..infinity );
```

$$\frac{1}{a\tilde{}}$$

```
> int( exp(-a*t)*t^(1/3), t=0..infinity );
```

$$\frac{2}{9}\frac{\pi\sqrt{3}}{a^{-4/3}\Gamma\left(\frac{2}{3}\right)}$$

It is somewhat unfortunate that the twiddle in the $a^{-4/3}$ might be mistaken for a minus sign. The routines `int`, `limit`, and `sqrt` needed to know in these examples if $a > 0$. They discover this information by testing if `signum(a)=1`. Similarly, we would like to simplify $\sin(n\,\pi)$ to 0 when n is known to be an integer. Though not implemented yet, this could be done by asking if `Im(n)=0` and `frac(n)=0`.

The Maple philosophy for `assume` is that it is better if programs 'concentrate' their use of `assume` by confining interrogation of the `assume` facility to a limited number of routines. This makes maintenance of programs easier, in the face of possible future changes to `assume`.

Let us now consider a slightly more complicated example, taken from [5]. We consider the computation of the metric for *prolate spheroidal coordinates*, which is the coordinate change

$$x = a\sqrt{(\xi^2-1)\,(1-\eta^2)}\cos\phi$$
$$y = a\sqrt{(\xi^2-1)\,(1-\eta^2)}\sin\phi$$
$$z = a\xi\eta\,,$$

where $a \geq 0$, $1 \leq \xi < \infty$, $-1 \leq \eta \leq 1$, and ϕ is real and may be taken on the range $0 \leq \phi < 2\pi$.

To compute the metric, we first compute the Jacobian matrix of this transformation, and the volume element in the new coordinates is (from multivariate calculus) the absolute value of the determinant of this Jacobian. We will have $dV = |\det(J)|\, d\xi\, d\eta\, d\phi$. These coordinates will be positively oriented if $\det(J) > 0$, and this is sometimes important to know.

```
> with(linalg):

Warning: new definition for norm
Warning: new definition for trace

> T := [a*sqrt((xi^2-1)*(1-eta^2))*cos(phi),
  a*sqrt((xi^2-1)*(1-eta^2))*sin(phi),
  a*xi*eta];
```

$$T := [a\sqrt{(\xi^2-1)\,(1-\eta^2)}\cos\phi,\ a\sqrt{(\xi^2-1)\,(1-\eta^2)}\sin\phi,\ a\xi\eta]$$

```
> J := jacobian(T,[xi,eta,phi]):
  d := simplify(det(J));
```

$$d := -a^3\xi^2 + a^3\eta^2$$

Now we can make our assumptions.

```
> assume(a>0);
> assume(xi>=1);
> assume(eta>=-1); additionally(eta<=1);
> assume(phi,real);
```

Now we can ask if $\det(J)$ is positive or negative:

```
> signum(d);
```

$$\text{signum}\,(-a^{\sim 3}\xi^{\sim 2} + a^{\sim 3}\eta^{\sim 2})$$

This fails because `signum` is not powerful enough to do this. The `signum` command uses the `isgiven` command rather than the `is` command which is expensive. However, we can test d ourselves using the `is`

command, which is a more powerful test. This routine is not used by `signum` because of efficiency considerations.

```
> is(d<=0);
```

true

For a final example, consider trying to decide if a function $f(x,y)$ is convex. This problem occurs in the calculus of variations (and in other contexts). If we wish to write a calculus of variations package in Maple (see [2, 3]) it turns out to be handy to automatically decide if f is convex. This can be done by seeing if the Hessian matrix of second partial derivatives of f is positive semidefinite (or negative semidefinite if we are looking for concavity). We begin the example below with some assumptions on x and y, which are supposed to reflect some known restrictions.

```
> assume(x,real);
> assume(y,real);
> additionally(x<0);
> additionally(x<y);
> additionally(x<-y);
```

Now we see if the following matrix is negative semidefinite.

```
> A := matrix([[x,y],[y,x]]);
```

$$A := \begin{bmatrix} x\tilde{} & y\tilde{} \\ y\tilde{} & x\tilde{} \end{bmatrix}$$

```
> definite(A,'negative_semidef');
```

$$x\tilde{} \le 0 \text{ and } -x\tilde{}^2 + y\tilde{}^2 \le 0 \text{ and } x\tilde{} \le 0$$

```
> is(");
```

true

We see that the linear algebra routine `definite` returns a boolean expression that is true if and only if the input matrix is definite (of the type specified). We can ask `is` if this expression is true or false.

When do we need to `assume` things, and how do we know?

The best way to proceed with assumptions is to make them *first*, if you know them. If you know that $\eta < 1$, then tell Maple so before you begin your computations. This may save you lots of time wasted wondering why Maple didn't give you a useful answer.

However, you may not know that you need to make an assumption. In that case, Maple makes some attempts to hint that you need to do something—currently, Maple's ability to tell you what it needs to find out is less than ideal, but progress is being made. The best clues that Maple gives at the moment are shown below.

1. Maple returns an unevaluated limit for an integral. For example,

   ```
   > int( exp(-s*t),t=0..infinity);
   ```

   $$\lim_{t \to \infty -} -\frac{e^{(-st)}}{s} + \frac{1}{s}$$

   This unevaluated limit hints that Maple can make some progress but needs to know more (for example, that $s > 0$) before it can properly evaluate the limit.

2. Maple refuses to simplify the answer. For example,

```
> exp(ln(x));
```

$$x$$

```
> ln(exp(x));
```

$$\ln(e^x)$$

In the first case, Maple knows that no matter what branch of the logarithm you take, $\exp(\ln(x)) = x$ and this simplification happens automatically. In the second case, however, if x is complex then $\ln(\exp(x)) = x + 2\pi ik$ where k is the *unwinding number* of x. Maple should *not* simplify this unless told that, for example, x is real. This type of return is difficult to distinguish from an inability on Maple's part to do the computation.

3. Maple returns an unevaluated `signum` or `csgn` function. Incidentally, the function `csgn` deserves to be better known. It is a generalization of the `signum` function to the complex plane, and allows nice encodings of information about functions with branch cuts. See `?csgn` for details.

```
> sqrt(4*q^2);
```

$$2\sqrt{q^2}$$

```
> simplify(");
```

$$2\,\mathrm{csgn}(q)q$$

Consider again the integral

```
> integrate( exp(-u*t)*t^(1/3), t=0..infinity );
```

$$\int_0^\infty e^{(-ut)} t^{1/3} dt$$

We have seen that Maple can do this integral if we tell Maple that $u > 0$. The main problem with this in more complicated examples is that often you won't know what you need to assume. For example, consider trying to solve $y'' + ay' + by = 0$. This equation has oscillatory solutions if $a^2 - 4b < 0$, and so expressions containing a division by y might conceivably need to know the sign of $a^2 - 4b$ (which did not appear in the input). If you are not well grounded in the theory of differential equations, how are you to tell *in advance* the right assumption to make? If no assumptions are made, then the computation may simply stop, and you may conclude that Maple can't do the computation at all.

There are several possible solutions to this problem that are currently being explored by Maple and other computer algebra systems.

1. **The 'pass the buck to the user' approach.** When in doubt, ask the user. This solution is used by Macsyma. It works nicely for the integration example above. The user would be asked a simple question like

```
> is u positive, negative, or zero ?
```

However, we have elsewhere seen examples where the question is not whether u is positive, negative, or zero, but whether a huge formula over a page long is positive, negative or zero. Worse, the question may contain symbols that did not appear in the user's original input (admittedly this reflects poor programming, because any intermediate variables must have some meaning in terms of the user's original variables). In these cases you really have no idea what to answer. A Macsyma user once told us

that one should "just answer 'positive', that usually gives you the answer you want." This is not very satisfactory, and one expects that at least some automation is possible here.

2. **The 'compute all solutions' approach.** The computer algebra system might compute all solutions. If the answer depends on whether u is positive, negative or zero, then the computer algebra system should give all three answers. For example, we might be satisfied with the following answer for the integral

```
> int( exp(-u*t), t=0..infinity );
```

piecewise(0 < u, 1/u, infinity)

This appears to be an attractive solution. It works well in simple examples like the ones in this article. But this approach becomes hopeless in more complicated cases where the number of cases that need to be examined blows up. [Combinatorial growth in the number of answers as the number of parameters increases is the norm.] Additionally, determining what the different cases are may be a harder problem than solving the cases. There is also a potential for disaster in returning all the solutions like this. For example, we had better not divide by

piecewise(0 < u, 1/u, infinity) - 1/u

This expression is zero if $u > 0$. This complicates the (already difficult) zero-recognition problem. Actually, at this point in a computation, the computer algebra system must split the computation into two paths. This approach forces a major redesign of the computer algebra system if it is going to handle this correctly.

3. **The 'encode all solutions' approach.** This approach is essentially the same approach as the previous one but instead of formally returning a piecewise function consisting of several cases, a single formula encoding the cases with the aid of the `signum` or `csgn` function is returned. Maple sometimes uses this approach. Again, this works well for simple examples but it fails for exactly the same reasons as described above in more complicated cases.

4. **The 'proviso' approach.** Another approach, which we believe to be the right approach, is the 'proviso' approach, where Maple will make an appropriate assumption, record it, and proceed to get as simple an answer as possible [4]. Later the user can check to see what assumptions were made. If the assumptions made were not the desired assumptions, the user can repeat the computation by first making the desired assumptions. This approach solves the problem when the user doesn't know or forgets to make the right assumptions, and it solves the problem for the computer algebra system of having to compute potentially a huge number of cases.

Future Plans

We outline here some ideas for improving the `assume` facility.

1. Put 'global' assumed domains for variables in place: the default should be that all variables are real. Thus $\sqrt{x^2}$ should simplify to signum(x)x or $|x|$. One should be able to specify instead that all variables are complex, or all integers, or all positive, or that all variables are purely formal or 'symbolic' and have *no* value, or only 'nice' values. In this case, any simplification which is valid for *any* value is acceptable.

2. Implement a 'proviso' mechanism which will tell the user directly when assumptions are necessary, and which assumptions are necessary. If intelligently designed, it will also lessen the need for the user to make assumptions explicitly. The simplest way to implement it would be to make assumptions on variables during the computation. For example, when computing $\int_0^\infty e^{-ut}dt$ the integration routine that

tests whether u is positive would assume, if u contains parameters and no assumptions have been made, that u is positive. The user would see

```
> int( exp(-u*t), t=0..infinity );
```

$$\frac{1}{u\tilde{}}$$

The twiddle reminds the user to check:

```
> about(u);

Originally u, renamed u~ :
  is assumed to be: RealRange(Open(0),infinity)
```

3. The use of the twiddle character to show that assumptions have been made is ugly and error prone. It can be mistaken for a minus sign, and different assumptions are displayed in the same way. The user interface needs to become more sophisticated here. A better solution would be to display the assumptions made beside the output expression, perhaps in this way:

```
> int( exp(-u*t), t=0..infinity );
```

$$\frac{1}{u} \text{ where } u > 0$$

Concluding Remarks

The current `assume` facility in Maple is a step in the right direction. It is not complete, or even particularly robust, and it is not used consistently. But it allows correct computation of solutions to problems that were beyond Maple until recently, and will improve.

References

[1] H. Caprasse: A complete simplification package for the absolute value function in the real domain. *SIGSAM Bulletin*, **20:** pp. 18-21, no. 1 & 2, (1986).

[2] Robert M. Corless, *Essential Maple, an introduction for scientific programmers*. Springer-Verlag, *to appear*, (1994).

[3] Robert M. Corless: *Symbolic Recipes*, vol. 1. Springer-Verlag, *to appear*, (1994).

[4] Robert M. Corless and David J. Jeffrey: Well, It isn't Quite that Simple. *SIGSAM Bulletin*, **26:** no. 3, pp. 2-6, (1992).

[5] André Heck: *Introduction to Maple*. Springer-Verlag, ISBN 0-387-97663-0, New York, (1993).

[6] Trudy Weibel and Gaston Gonnet: An Algebra of Properties. *Proc. ISSAC '91, Bonn*, pp. 352-359, (1991).

[7] Trudy Weibel and Gaston Gonnet: An assume facility for CAS with a sample implementation in Maple. *Conference presentation at DISCO '92, Bath*, (1992).

Random Number Generation and Testing[1]

Zaven A. Karian and Rohit Goyal[2]

Simulation studies of discrete event models almost always require the use of random observations from empirically or theoretically specified distributions. There are a variety of methods (see, for example, Chapter 4 of Karian and Dudewicz [3]) of producing such observations from random numbers (i.e., random observations from the uniform distribution on the (0,1) interval). To simulate Poisson arrivals into a queueing system, for example, requires the generation of exponentially distributed inter-arrival times, $t_1, t_2, ..., t_k$ which can be obtained through

$$t_i = -\lambda \ln(r_i) \text{ for } i=1,2,...,k \quad (1)$$

where $r_1, r_2, ..., r_k$ are random numbers and λ is the parameter of the exponential distribution.

Note that the use of (1) guarantees appropriate randomness qualities of $t_1, t_2, ..., t_k$ only if $r_1, r_2, ..., r_k$ exhibit randomness qualities. Thus, the need to generate high-quality random numbers is crucial to simulation studies that model stochastic processes. Since it is impractical to use true random numbers such as those produced by a physical process (e.g., numbers from a geiger counter measuring background radiation), computer simulations use pseudo-random numbers generated by some algorithm. Following common convention, we will use the term random numbers to refer to pseudo-random numbers.

In this paper we give a detailed analysis of `rand()`, the random number generator that is embedded in Maple, and make some suggestions for obtaining independent streams of random numbers.

The `rand()` Generator

The `rand()` generator, developed by Michael Monagan for the Maple library, belongs to the class of linear congruential generators that are characterized by

$$X_{i+1} \equiv aX_i + c \text{ (modulo } m) \text{ for } i=1,2,... \quad (2)$$

where the modulus m, the multiplier a, the increment c and the seed X_0 are positive integers with $0 \le a, c, X_0 < m$. Specific choices of a, c and m can produce excellent generators (URN12 and URN30, pp. 115–118, [3]) as well as very poor ones (URN7 and URN10, pp. 114–115, [3]). `rand()` uses $m = 999999999989$, $a = 427419669081$, $c = 0$ and a defaulted X_0=1, stored as the global variable `_seed`. Detailed discussions of congruential random number generators are available in [3] and [4].

Quality of the `rand()` Generator

The quality of a random number generator is generally based on the factors discussed below.

PERIOD OF THE GENERATOR

By necessity all congruential random number generators have periods less than or equal to m. By a result due to R. D. Carmichael (see Knuth [4], p. 19), when $c = 0$ the maximal period is achieved if X_0 is relatively

[1] Supported by the Denison University Research Foundation.

[2] Department of Mathematical Sciences, Denison University, Granville, Ohio 43023. Email: karian@denison.edu and goyal_r@denison.edu

prime to m and a is a primitive element modulo m. Furthermore, for prime modulus, m, Knuth (p. 20, [4]) gives the following criterion for a maximal period of m-1:

$$a^{(m-1)/q} \not\equiv 1 \text{ (modulo } m) \text{ for all } q \in Q \tag{3}$$

where Q is the set of prime divisors of m-1. Since `rand()` uses a prime modulus, its period could be as large as m-1 = 10^{12}-12. The Maple procedure `MaxPeriod(a, m)`, available through the stats subdirectory of the share library, is based on (3) and can be used to verify that this is indeed the case.

Many recently proposed congruential generators take advantage of the 32-bit architecture of modern computers and use $m=2^{32}$ or $m=2^{31}$-1 (a prime). The $m=2^{31}$-1 generators, with maximal period of 2^{31}-2 $\approx 2 \times 10^9$, have been exhaustively investigated by Fishman and Moore [2]. The period of `rand()` is about 500 times that of the $m=2^{31}$-1 generators.

STATISTICAL RANDOMNESS

Although the period is an important consideration, statistical randomness qualities, which are independent of the period, are even more important. Consider, for example, the case $a=c=1$ which, for $X_0 = 1$, generates the sequence 1, 2, ... Our judgement of the quality of a generator is based on its performance on a battery of 19 statistical tests implemented in the TESTRAND[3] package which is described in detail in [1]. We briefly describe the most basic of the TESTRAND tests, the chi-square on chi-square test for uniformity.

Let the interval (0, 1) be partitioned into k equal-sized sub-intervals. Given a set of n random numbers, let e_i and o_i, respectively denote the expected and the observed frequencies of the random numbers in the i-th sub-interval. Then,

$$D = \sum_{i=1}^{k} \frac{(o_i - e_i)^2}{e_i} \tag{4}$$

has an approximate chi-square distribution with k-1 degrees of freedom. Clearly, if the set of n numbers is truly random, then D must be small. The randomness hypothesis is rejected if D exceeds the 100(1-α)th percentile, i.e., $D > \chi^2_\alpha (k-1)$, where α is the significance level of the test. In practice, if D exceeds the 95th percentile the generator is rejected. This is the usual chi-square test for uniformity. If the above test is performed on many batches of numbers, even a good generator would be rejected about 5% of the time. This is undesirable because it excludes sequences with large variations typical of true randomness.

The chi-square on chi-square (or CSCS) test solves this problem by considering a large number of D's. Let $D_1, D_2, ..., D_{1000}$ be the χ^2 statistics for 1000 batches of 10000 numbers each. Then each D_i, $i = 1, ..., 1000$ has an approximate chi-square distribution with k-1 degrees of freedom. The set of D_i's can be tested with the uniformity of distribution test with, say, 100 sub-intervals, whose boundaries are the 1%, 2%, ..., 99% points of the chi-square distribution with k-1 degrees of freedom. The CSCS statistic is then calculated as in (4). The CSCS test is stronger than the usual chi-square test and we recommend its use in all situations.

The CSCS statistics that result from the 19 TESTRAND tests are compared to the 95th and 99th percentiles of the chi-square distribution with 99 degrees of freedom; these values are 123.225 and 134.642, respectively. Following the standards established in [3], a generator would be disqualified if any one of its 19 CSCS values exceeds the 99th percentile, or two or more of them exceed the 95th percentile. As can be seen from Table 1, all 19 CSCS statistics for the `rand()` generator are below the 95th percentile mark, indicating that `rand()` passes every statistical test in the TESTRAND package. Only 3 of the 35 generators

[3] A random number generation and testing package available from The Ohio State University and Denison University.

considered in [3] performed this well; this puts `rand()` in the company of the best generators that are in common use.

Table 1: TESTRAND Results for `rand()`.

TEST	CSCS	TEST	CSCS
UDISTB	86.8	COUPON D=5	104.2
COUPON D=5, FR(100*R)	119.6	COUPON D=10	105.4
GAP BELOW MEAN	91.2	GAP ABOVE MEAN	114.0
GAP (0.333, 0.667)	80.0	PERMUTATION 3's	80.2
PERMUTATION 4's	108.4	PERMUTATION 5's	118.6
POKER (HAND 4, PART. 4)	87.0	POKER (HAND 5, PART. 6)	101.2
POKER (HAND 5, PART. 4)	93.2	RUNS UP R	119.8
RUNS UP FR(10*R)	122.6	RUNS UP FR(100*R)	92.2
SERIAL PAIRS 3×3	86.8	SERIAL PAIRS 10×10	108.2
SERIAL PAIRS 20×20	111.0		

PERFORMANCE ON THE SPECTRAL TEST

Let $u_1, u_2, \ldots$ be a sequence of random numbers produced by a *linear congruential* random number generator. Then the points $(u_1,u_2,\ldots,u_t)$, $(u_2,u_3,\ldots,u_{t+1})$, $(u_3,u_4,\ldots,u_{t+2})$, ... must lie on a finite number of equally spaced parallel hyperplanes in t-space. The spectral test is based on ν_t, the number of such hyperplanes, or equivalently, on $d_t = 1/\nu_t$, the inter-planar distance. For a given multiplier, a, the more hyperplanes there are, the better distributed the t-tuples are in the t-dimensional unit cube. For this reason, we look for maximal ν_t or minimal d_t for t=2,3,.... The best possible d_t is (see [3], pp. 107–108)

$$d_{t,\,min} = \frac{1}{((t!m)^{1/t} - 1)}. \tag{5}$$

Thus, $r_t = d_t / d_{t,\,min}$ which is always greater than 1, can be used to measure the relative quality of the multiplier. Of course, the closer r_t is to 1 the better.

We implemented a spectral test algorithm that determines r_t. `spectral()`, the Maple procedure for this algorithm (Algorithm S, pp. 98–100, [4]), is available from the stats subdirectory of the share library. `rand()`, when tested with `spectral()` produced

$$r_2=1.75,\quad r_3=2.20,\quad r_4=2.87,\quad r_5=2.89,\quad r_6=3.66.$$

On the basis of comparisons with other congruential generators (see pp. 106–112, [3]), we consider values of $r_t < 4$ to be very good.

To appreciate the significance of r_t, consider the case of t=3. For m=999999999989, the maximum number of hyperplanes possible is

$$\frac{1}{d_{t,\,min}} = (999999999989 \times 3!)^{1/3} - 1 = 18170. \tag{6}$$

Since r_3=2.20 when a=427419669081, we are assured that at least $1/(r_3 d_{3,min}) = 8259$ planes are needed to capture the 3-tuples that result from `rand()`.

In a related but different approach, Knuth (see p. 102, [4]) gives a test criterion based on

$$\mu_t = \frac{\pi^{t/2} v_t^t}{(t/2)!m}, \tag{7}$$

which is the volume of the ellipsoid

$$(x_1 m - x_2 a - \cdots - x_t a^{t-1})^2 + x_2^2 + \cdots + x_t^2 \le v_t^2. \tag{8}$$

Knuth uses this volume as a measure of the likelihood of certain points with integer coordinates being in the ellipsoid (8). The specific test criterion stated by Knuth (pp. 102, [4]) is

> We might say that the multiplier a passes the spectral test if μ_t is 0.1 or more for $2 \le t \le 6$, and it "passes with flying colors" if $\mu_t \ge 1$ for all these t.

Our use of `spectral()` on `rand()`'s multiplier yielded

$$\mu_2 = 2.05, \quad \mu_3 = 2.37, \quad \mu_4 = 1.75, \quad \mu_5 = 3.10, \quad \mu_6 = 1.51,$$

making `rand()` a definite "high flyer."

Summary of Results and Some Comments

The `rand()` generator has a large period, good statistical qualities, and does very well on the spectral test. We recommend its use in almost any study that requires random numbers. There are, of course, exceptions; linear congruential generators would not be appropriate in cryptology since the parameters of the generator can be easily determined from a sufficiently long sequence of numbers.

It is worth noting that Monagan chose a=427419669081 solely on its performance on the spectral test. In fact, Knuth specifically states (p. 89, [4])

> ... not only do all good generators pass [the spectral] test, all generators now known to be bad actually *fail* it. Thus it is by far the most powerful test known, and it deserves special attention.

However, we are aware of a number of congruential generators (e.g., $m=2^{31}-1$, a=1078318381) that do very well on the spectral test but fail several statistical tests in TESTRAND.

We also draw the user's attention to a misinterpretation that can result from `rand()`'s help message which states that `rand()` produces 12-digit positive integers. This would imply that to obtain random numbers on (0,1) one needs to divide `rand()`'s output by 10^{12}. However, `rand()` actually produces numbers between 1 and 999999999989, and dividing by 10^{12} would cause a slight bias.

Constructing Additional Linear Congruential Generators

In complex models, where independent and parallel processes need to be simulated, it is often desirable to have multiple independent streams of random numbers. Additional generators of good quality can be obtained as follows:

1. Choose a large prime as modulus, possibly the one used by `rand()`.
2. Choose a multiplier a, which gives maximum period. This can be done by picking an integer $b > \sqrt{m}$ and using `a:=numtheory[primroot](b,m)`. A particular a could also be tested by using the `MaxPeriod()` function that returns `true` if the chosen multiplier gives a maximal period and `false` otherwise.
3. Perform the spectral test on the multiplier-modulus combination chosen. If it fails the test then repeat

from step 2.

4. Perform tests for statistical quality, such as the TESTRAND tests. If the generator does not perform well on these tests, repeat from step 2.

We performed the above for m=999999999989, the modulus of `rand()`, and after rejecting two multipliers found the multiplier a=745580037424 that gave rise to a generator of quality comparable to `rand()`. The Maple interaction that lead to the construction of this generator is given below.

```
> with(share):
> readshare(spectral, stats):
> m := 999999999989:
> a := 745580037422:
> maximal := false:
> while not maximal do
>     a := a+1:
>     maximal := MaxPeriod(m, a)
>   od:
> [m, a];
```

$$[999999999989, 745580037424]$$

```
> spectral(m, a, 6):
```

$$t = 2, v = 565058.251600, v^2 = 319290827665.000000, d = 0.000002,$$
$$\mu = 1.003082, r = 2.502773$$

$$t = 3, v = 8917.721514, v^2 = 79525757.000000, d = 0.000112,$$
$$\mu = 2.970642, r = 2.037539$$

$$t = 4, v = 899.735517, v^2 = 809524.000000, d = 0.001111, \mu = 3.233920, r = 2.458905$$

$$t = 5, v = 200.232365, v^2 = 40093.000000, d = 0.004994, \mu = 1.694220, r = 3.263156$$

$$t = 6, v = 81.847419, v^2 = 6699.000000, d = 0.012218, \mu = 1.553561, r = 3.645558$$

After two iterations of the `while` loop, `MaxPeriod()` returned `true` for a=745580037424 and m=999999999989. The spectral test for this set of m and a for dimensions 1 through 6 produced

$$r_2 = 2.50, \quad r_3 = 2.04, \quad r_4 = 2.46, \quad r_5 = 3.26, \quad r_6 = 3.65$$

and

$$\mu_2 = 1.00, \quad \mu_3 = 2.97, \quad \mu_4 = 3.23, \quad \mu_5 = 1.69, \quad \mu_6 = 1.55.$$

Like `rand()`, this generator also passed all of the statistical tests of the TESTRAND package.

Acknowledgements

We wish to thank Michael Monagan for information on `rand()` and a number of helpful suggestions for improving this paper. Thanks are also due to Stan Devitt for his assistance.

References

[1] E.J. Dudewicz and T.G. Ralley: *Handbook of Random Number Generation and Testing with TESTRAND Computer Code*, American Sciences Press, Inc., Columbus, Ohio, (1981).

[2] G.S. Fishman and L.R. Moore: An Exhaustive Analysis of Multiplicative Congruential Random Number Generators with Modulus 2^{31}-1, *SIAM J. Sci. Stat. Comput.*, Vol. 7, No. 1, (1986).

[3] Z.A. Karian and E.J. Dudewicz: *Modern Statistical, Systems, and GPSS Simulation: The First Course*, Computer Science Press, New York, (1991).

[4] D.E. Knuth: *The Art of Computer Programming: Seminumerical Algorithms*, Vol. 2, Addison-Wesley, Reading, (1969).

Gaïa: A Package for the Random Generation of Combinatorial Structures

Paul Zimmermann[1]

Gaïa is a computer algebra package that helps counting and drawing random combinatorial structures of various sorts. It is an implementation of the calculus developed by Ph. Flajolet, B. Van Cutsem and the author in [5]. Given a combinatorial specification and an integer n, it draws a random object uniformly amongst all size n structures. It applies to all decomposable structures, either labelled or unlabelled, including trees of various kinds, surjections, set partitions, permutations, functional graphs of many sorts.

Some applications of random generation are: (*i*) analyzing the average case complexity of algorithms by making simulations to guess or to check analytic results, (*ii*) checking the correctness of programs by feeding them with random inputs, (*iii*) getting ideas about some parameter of a class of objects, for example the height of trees or the number of connected components of graphs, (*iv*) simply drawing a random object.

Uniform random generation is difficult because there is generally no closed formula for the number A_n of data structures of size n, and secondly most methods require an explicit bijection with integers modulo A_n, but such a bijection is known only in a few cases (for example permutations and integer partitions, see the `combinat` package).

The main idea underlying the Gaïa system is first to transform the specification of a combinatorial class into a *standard* specification restricted to atoms and union, product, *pointing* constructors; then the standard specification is translated into counting and drawing procedures using some well-defined *templates*. This ensures a *really uniform* random generation in $O(n \log n)$ arithmetic operations in the worst case, after a $O(n^2)$ preprocessing to compute the counting sequences up to size n.

This article explains how to define a class of decomposable combinatorial structures with Gaïa, how to count the number of structures of a given size, how to generate a random structure and how to use it. Details about the algorithms used will be found in [5] and [6].

A simple example

Once you have properly installed Gaïa as a Maple package (see the section 'Installing the package' below), it is very easy to generate a random object, for example a random binary tree:

```
% maple
> with(gaia):
  binary_tree := { B = Union(Z, Prod(B,B)) }:
  draw(binary_tree,unlabelled,B,7);
```

Prod(Prod(Z, Prod(Z, Prod(Prod(Prod(Z, Z), Z), Z))), Z)

```
> draw(binary_tree,labelled,B,5);
```

Prod(Prod(Prod(Z[2], Prod(Z[5], Z[1])), Z[4]), Z[3])

[1] Inria Lorraine, Nancy, France, Paul.Zimmermann@loria.fr. This work was partly supported by the ESPRIT Basic Research Action No. 7141 (ALCOM II).

The command `with(gaia)` loads the package, then one defines the grammar for binary trees, one draws an unlabelled tree of size 7 and a labelled one of size 5. The first two arguments of the `draw` command define a combinatorial specification, that is a grammar and a labelling type (see the section 'Defining a combinatorial specification' below). The third argument indicates the type of object to be generated (the specification may define several types) and the last one the desired size.

The function `count` is similar to `draw`, except it gives the number of objects of a given size:

```
> count(binary_tree,labelled,B,33);
```

48221992399111497884345907291989267777631289344000000

Defining a combinatorial specification

A class of decomposable combinatorial structures either contains only one object, or is built from simpler classes by means of *constructors*. The elementary classes are Epsilon, which denotes an object of size zero, and Z, which denotes an object of size one. The available constructors are:

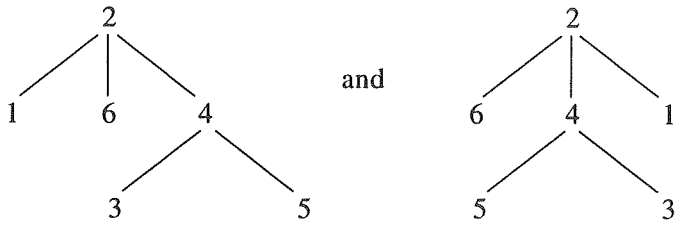
Atom	object of size 1 (Z is a predefined atom)
Union(A, B, ...)	disjoint union of the classes A, B, ...
Prod(A, B, ...)	product of the classes A, B, ...
Set(A)	all sets whose elements are in A
Sequence(A)	all sequences with elements of A
Cycle(A)	all directed cycles with elements of A.

For the constructors Set, Sequence and Cycle, it is possible to add some restrictions on the cardinality: for example, Set(A, card ≥ 1) means all non empty sets whose elements are in A, Sequence(A, card ≤ 3) means all sequences of at most three elements of A, and Cycle(A, card $= 5$) means all cycles of five elements from A.

A specification is a grammar and a labelling type, which is either 'labelled' or 'unlabelled'. In the labelled universe, each atom has a unique label, which is an integer from 1 to n, where n is the size of the whole object. In other words, the labels define a total order on all n atoms. In the unlabelled universe, there is no label. The grammar itself is a set of productions of the form $A = \langle \text{rhs} \rangle$, where A is the name of the class being defined, and $\langle \text{rhs} \rangle$ is an expression involving elementary classes, constructors and other classes. Below are some grammars and the corresponding combinatorial objects they define in the labelled universe.

{A = Prod(Z,Set(A))}	non plane trees
{B = Union (Z,Prod(B,B))}	plane binary trees
{C = Prod (Z,Sequence(C))}	plane general trees
{D = Set(Cycle(Z))}	permutations
{E = Set(Cycle(A)), A=Prod(Z,Set(A))}	functional graphs
{F = Set(Set(Z,card ≥ 1))}	set partitions
{G = Union (Z,Prod(Z,Set(G,card $= 3$)))}	non plane ternary trees
{H = Union (Z,Set(H,card ≥ 2))}	hierarchies
{L = Set(Set(Set(Z,card ≥ 1),card ≥ 1))}	3-balanced hierarchies
{M = Sequence(Set(Z,card ≥ 1))}	surjections

A non plane tree (type A) is a root node (Z) to which are attached some subtrees that may take any position around the root, thus forming a set; the set may be empty, and this gives a terminal node, that is a leaf. For example,



represent the same labelled non plane tree. In plane binary trees (type B), the number of subtrees is restricted to be two or zero, and they are ordered. Thus we get the grammar B = Union(Z,Prod(Z,B,B)), or simply B = Union(Z,Prod(B,B)) if we do not count internal nodes. A plane general tree (type C) is similar to a non plane tree except the subtrees are ordered (now the two pictures above represent two different plane trees), thus we just replace the Set by a Sequence construction in the grammar of A.

For permutations (type D), we could represent a permutation on $\{1 \ldots n\}$ by the sequence of its images $\sigma_1 \ldots \sigma_n$, for example the sequence 6,2,5,1,3,4 would represent the permutation σ_1=6, σ_2=2, σ_3=5, σ_4=1, σ_5=3, σ_6=4. This would give the grammar D = Sequence(Z). But usually it is more convenient to work on the cycle decomposition, for example (1 6 4) (2) (3 5) for the above permutation, which is defined by D = Set(Cycle(Z)). This last grammar is in some sense "more precise", the construction Set(Cycle(•)) being equivalent to Sequence(•) for labelled objects.

Functional graphs (type E) are graphs of functions on $\{1 \ldots n\}$. Such a function f has two kinds of points: cyclic points i such that some iterate of f on i goes back to i, such as 4, 8, 10, 11, 14 on Figure 1, and other points, which are non-cyclic. Starting from any point, and iterating the function, we attain necessary a cyclic point in a finite number of iterations (this is the trick used in Pollard's algorithm to find a factor of an integer). The set of points that go to the same cyclic point is a non plane tree (type A).

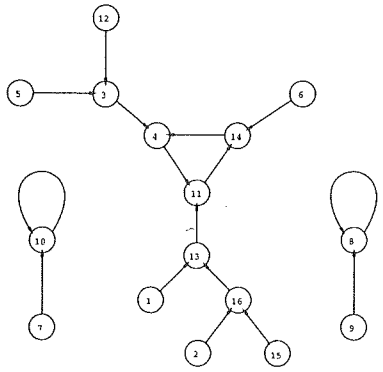


Figure 1: The graph of $x \rightarrow x^2 + 12 \bmod 17$

A partition of a set is exactly a set of non-empty sets, the latter being defined by Set(Z,card ≥ 1), thus we get the grammar of F. Non plane ternary trees (type G) are defined like non plane trees, except the number of subtrees is either 0 or 3 in the above grammar, we simplified Prod(Z,Set(G,card = 0)) into Z. A hierarchy (type H) is similar to a non plane tree too, but unary nodes are forbidden, thus the number of subtrees is either zero or greater or equal to two. Three-balanced hierarchies (type L) are balanced non plane trees (all leaves are at the same level) of height exactly 3. Finally, a surjection (type M) from $\{1 \ldots n\}$ to a totally ordered set is equivalent to a sequence of non empty sets (the integers with image the smallest element are in the first set, those with image the second smallest one are in the second set, and so on).

Other combinatorial objects are defined by the following grammars in the unlabelled universe.

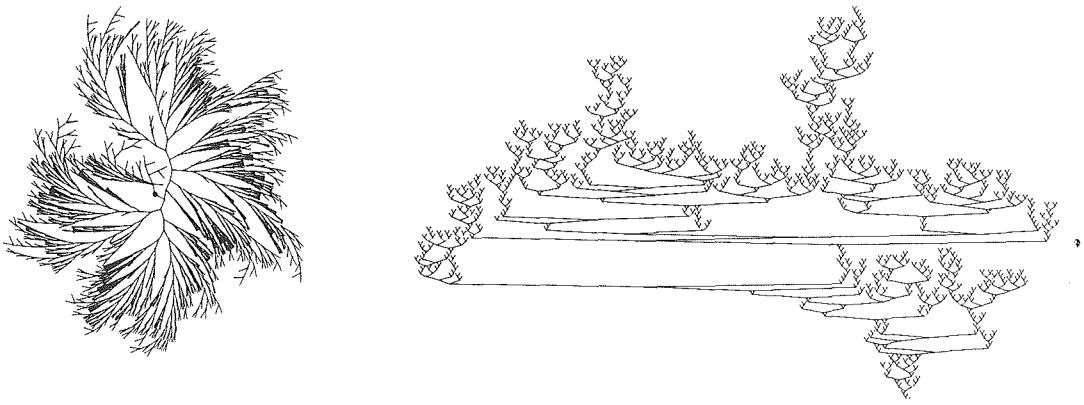
$\{A = \text{Set}(\text{Sequence}(Z, \text{card} \geq 1))\}$	integer partitions
$\{B = \text{Sequence}(\text{Union}(Y,Z)),\ Y = \text{Atom}\}$	binary sequences
$\{C = \text{Cycle}(\text{Set}(Z, \text{card} \geq 1))\}$	necklaces
$\{D = \text{Prod}(Z, \text{Set}(D))\}$	rooted unlabelled trees
$\{E = \text{Set}(\text{Cycle}(D)),\ D = \text{Prod}(Z, \text{Set}(D))\}$	random mappings patterns
$\{F = \text{Union}(Z, \text{Set}(F, \text{card} = 2))\}$	non plane binary trees
$\{G = \text{Union}(Z, \text{Set}(G, \text{card} = 3))\}$	non plane ternary trees
$\{H = \text{Union}(Z, \text{Set}(H, \text{card} \geq 2))\}$	unlabelled hierarchies
$\{M = \text{Sequence}(\text{Set}(Z, \text{card} \geq 1))\}$	integer compositions

It should be noticed that the same grammar may define different kinds of objects. As an example, Sequence(Set(Z, card ≥ 1)) defines surjections in the labelled universe, but integer compositions in the unlabelled universe.

Here again, the specifications are explained as follows. An integer partition, for example 17=12+3+1+1, is equivalent to a set of boxes of integer length, with repetitions allowed: { □□□□□□□□□□□□, □□□, □, □ } . Such a box is simply a non empty sequence of atoms: Sequence(Z, card ≥ 1). A necklace (type *C*) is a cycle of non empty sets of beads. By the way, let us remark that a set of beads Set(Z, card ≥ 1) is equivalent to a sequence of beads Sequence(Z, card ≥ 1) in the unlabelled universe.

Rooted non plane trees *D* have exactly the same grammar than in the labelled case. Similarly, random mappings patterns (type *E*) are the "skeletons" of functional graphs. Trees and hierarchies (types *F*, *G* and *H*) are defined like in the labelled case.

Figure 2 shows two objects of size 1000 generated using Gaïa: the first one is the binary search tree corresponding to a random permutation of size 1000 (type *D* in the labelled case), the second one is a plane binary tree. The left drawing was produced using a special-purpose Maple routine, and the right one was obtained using the algorithm described in [11] (Gaïa only produces a Maple expression, it does not include any graphical instruction). These examples show some values of interest that could be examined on combinatorial objects: the height of different kinds of trees, the number of sets in a random set partition, or the number of terms in a random integer partition, the distribution of degrees in general trees, the number of cycles in a permutation, etc.



A binary search tree of size 1000.
`{D=Set(Cycle(Z))},labelled`

A binary plane tree of size 1000.
`{B=Union(Z,Prod(B,B))},labelled`

Figure 2: Two random objects generated by Gaïa.

Using and printing objects generated by Gaïa

All objects produced by Gaïa are valid Maple expressions. They are either names (possibly labelled) representing atoms, or inert functions for all constructors. Thus you can access the components of an object with the usual Maple functions `op`, `nops`. For example, the following function computes the size of an object:

```
> size := proc(e)
    if type(e,epsilon) then 0
    elif type(e,name) then 1
    else convert(map(procname,e),'+')
    fi
  end:
```

We can check it rapidly:

```
> size(draw(binary_tree,unlabelled,B,20));
```

$$20$$

If you want your objects to be printed another way than the default, you can easily do it by redefining the functions `gaia/print/xxx` where `xxx` is a constructor. Take for example Cayley trees, which are printed by default as follows:

```
> Cayley := {A = Prod(Z,Set(A))},labelled:
  draw(Cayley,A,4);
```

Prod(Z[2], Set(Prod(Z[1], EmptySet), Prod(Z[4], Set(Prod(Z[3], EmptySet)))))

If you want to use Maple curly-bracket notation instead, just redefine `gaia/print/Set` for general sets and `gaia/print/EmptySet` for empty sets:

```
> 'gaia/print/Set' := () -> {args}:
  'gaia/print/EmptySet' := () -> {}:
  draw(Cayley,A,4);
```

Prod(Z[1], {Prod(Z[3], {{}, Prod(Z[2], {})}), {{}, Prod(Z[4], {})}})

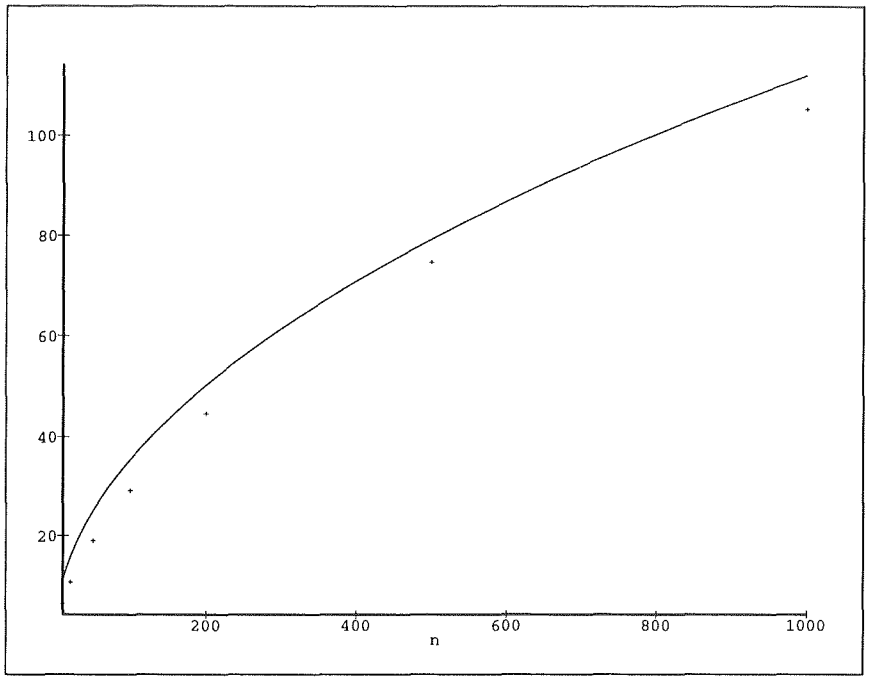
Notice that the `gaia/print/xxx` functions do not only modify the way objects are *printed* like the `print/xxx` functions of Maple, but really modify the internal structure of the objects (and consequently user-defined functions like `size` above may have to be redefined accordingly). This behaviour enables one to work further with random objects.

For example, suppose we want to analyze the height of unlabelled binary trees. We first write a `height` function:

```
> height := proc(b)
    if type(b,name) then 0 else 1+max(height(op(1,b)),height(op(2,b))) fi
  end:
```

and we are ready to experiment and compare to the actual result of $2\sqrt{\pi n} + O(n^{1/4+\varepsilon})$ from [4, Theorem B p. 200]. We plot for different sizes the average height over 100 random binary trees:

```
> s:=NULL:
  for n in [10,20,50,100,200,500,1000] do
    l:=seq(height(draw(binary_tree,unlabelled,B,n)),i=1..100);
    s:=s,[n,stats[average](l)]
  od:
  exper:=plot([s],n=10..1000,style=POINT):
  theor:=plot(2*sqrt(Pi*n),n=10..1000):
  plots[display]({exper,theor});
```



Similarly, one could analyze the path length of binary trees, the number of cycles in a random permutation, the number of connected components of a random functional graph, the number of elements in a set partition or an integer partition, the average node degree in a random hierarchy, and so on.

Some advanced examples

A lot of combinatorial structures encountered in the literature are decomposable, that is expressible by a specification in Gaïa. For example we saw in the section 'Defining a combinatorial specification' that a functional graph on $\{1 \ldots n\}$ is a set of cycles, each cycle being made of non plane trees; a functional digraph on $\{1 \ldots n\}$ is similar, except the cycles must have at least two elements. We can easily check the figures given in [9, p. 70]:

```
> sys:={F=Set(Cycle(D)),D=Prod(Z,Set(D)),FD=Set(Cycle(D,card>=2))},unlabelled:
  seq(count(sys,FD,n),n=1..11);
```

0, 1, 2, 6, 13, 40, 100, 291, 797, 2273, 6389

```
> seq(count(sys,F,n),n=1..11);
```

1, 3, 7, 19, 47, 130, 343, 951, 2615, 7318, 20491

Another beautiful example was suggested by Volker Strehl. We consider bicolored functional graphs on $\{1 \ldots n\}$, where each point has a color, either blue or red, and has at most one ancestor of each color. The corresponding specification is the following, with `Ab` (resp. `Ar`) denoting trees with a blue (resp. red) root, and `E` denoting bicolored functional graphs.

```
> sys := {Ab = Union(b,Prod(b,A),Prod(b,Ab,Ar)),
    Ar = Union(r,Prod(r,A),Prod(r,Ab,Ar)),
    A = Union(Ab,Ar),
    A2 = Union(Prod(r,Ab),Prod(b,Ar)),
    C = Cycle(Union(A2,b,r)),
    E = Set(C),
    b = Atom,
    r = Atom}, labelled:
  seq(count(sys,E,n),n=0..9);
```

1, 2, 12, 120, 1680, 30240, 665280, 17297280, 518918400, 17643225600

The numbers found are exactly $(2n)!/n!$ up to n=9. It is left as an exercise to the reader to check if this is true for every n. This is a typical example of research in combinatorics: defining with Gaïa a particular kind of objects, computing the first numbers, looking for an explicit formula or for similar sequences in Sloane's book [12], and perhaps deriving a bijection with other combinatorial objects.

The list of combinatorial constructors given above is not complete. In fact, the system itself uses two other constructors, `Theta` and `Int`. The construction `Theta(A)` produces objects of type `A` with one atom having a special mark, and `Int(A)` simply erases the mark in the objects of type `A`. Thus the constructor `Int` is only valid for marked objects. These two constructors are used in the *standard form* of combinatorial specifications (see [5] for more details). As an example, the standard form of the labelled specification `A=Set(B)` is:

```
> standardform({A = Set(B)},labelled);
```

{T1 = Prod[Set](T0, A), T2 = Int(T1), A = Union(EmptySet, T2), T0 = Theta(B)}

which means that (an object of type) A is either the empty set or T_2, T_2 being an object of type T_1 without the mark, T_1 being the product of T_0 and A, and T_0 being a marked object of type B.

In the unlabelled case, the standard specification uses a third constructor, the generalized diagonal `Delta` defined in [6]:

```
> standardform(A = Set(B),unlabelled);
```

{T1 = Delta[Set](T0), T2 = Prod[Set](T1, A), T0 = Theta(B), T3 = Int(T2),
A = Union(EmptySet, T3)}

These three constructions `Theta`, `Int` and `Delta` allow to define a wider class of structures. The following specifies for example unrooted non plane trees (the reader is not necessarily supposed to understand the specification, which is based on the notion of *similar node* defined in [9]).

```
> sys:={T=Prod(Z,Set(T)),t=Int(Union(T,Prod(T,Delta[Set(2)](Theta(T)))),
        Delta[2](Theta(T)))))},unlabelled:
  sum('count(sys,t,n)*x^n',n=1..10);
```

$$x + x^2 + x^3 + 2x^4 + 3x^5 + 6x^6 + 11x^7 + 23x^8 + 47x^9 + 106x^{10}$$

A lot of examples in the book of Harary and Palmer can be checked in the same manner, like in those of Comtet [3], Goulden and Jackson [7] and Bollobás [2].

Installing the package

For those who have an access to Internet, the Gaïa package is available by anonymous ftp from the machine `ftp.inria.fr`: [2]

```
% ftp ftp.inria.fr
Name (ftp.inria.fr:zimmerma): anonymous
Password: <your e-mail address>
ftp> cd INRIA/Projects/algo/gaia
ftp> bin
ftp> get gaia1.1.tar.Z
ftp> quit
% uncompress gaia1.1.tar.Z
% tar xvf gaia1.1.tar
```

[2] The following instructions assume you are working within a Unix-like environment.

This will create the following files: `gaia.mpl`, `gfun.mpl`, `gaia.test` and `README.tex`. Then you must create a Maple ".m" file from the files `gaia.mpl` and `gfun.mpl`. To do this, type:

```
% maple -s -q < gaia.mpl
% maple -s -q < gfun.mpl
```

You have now two files `gaia.m` and `gfun.m`. To be able to load the Gaïa package easily from Maple, add in your `.mapleinit` file (in your home directory) the line:

```
libname := '/users/eureca/zimmerma/Gaia',libname:
```

(`/users/eureca/zimmerma/Gaia` is the directory where the file `gaia.m` lies). Once you have created the file `gaia.m` and updated your `.mapleinit` file, just check that all works properly:

```
% maple -q < gaia.test
Total time=   215.133
```

Further developments.

Due to the exponential growth of the coefficients, the more expensive operations are those that deal with those huge numbers (the number of unlabelled binary trees of size 1000 has 597 digits). For this kind of computation, Maple is not as efficient as some specialized libraries like GMP [8], BigNum [10] or Pari [1]. An interface with these multiprecision libraries is in preparation. It works as follows: in Maple, you type:

```
> compile(binary_tree,unlabelled,gmp,'foo.c');
```

and this creates a C program `foo.c` that generates random unlabelled binary trees, using the multiprecision library GMP. The generation of random objects is about ten times faster with the C interface. The trees on page 41 were generated in about 10 seconds each using this C interface. Please contact the author for more information on this.

Once a random object was generated, Gaïa is not able to generate the *next* one, like the function `nextpart` of the `combinat` package. This ability would be very useful, because it would enable one to list all objects of a given size. Unfortunately, as already said in the introduction, this would require an explicit bijection between objects of size n and integers modulo A_n. This seems to be awkward with the methods of [5, 6].

Acknowledgements

The author thanks Bruno Salvy for his comments on a previous version of this article, and the referees for their careful reading and interesting remarks.

References

[1] C. Batut, D. Bernardi, H. Cohen and M. Olivier: *User's Guide to PARI-GP*, (Dec. 1991). Available by anonymous ftp from `megrez.ceremab.u-bordeaux.fr` or `math.ucla.edu`.

[2] B. Bollobás: *Random Graphs*, Academic Press, (1985).

[3] L. Comtet: *Advanced Combinatorics*, Reidel, Dordrecht, (1974).

[4] P. Flajolet and A. Odlyzko: The average height of binary trees and other simple trees. *J. Comput. Syst. Sci. 25* (1982), 171–213.

[5] P. Flajolet, P. Zimmermann and B. V. Cutsem: A calculus for the random generation of combinatorial structures. *Theoretical Comput. Sci.*, 29 pages, to appear. Also available as Inria Research Report number 1830.

[6] P. Flajolet, P. Zimmermann and B. V. Cutsem: *A calculus of random generation: Unlabelled structures*, (in preparation).

[7] I. P. Goulden and D. M. Jackson: *Combinatorial Enumeration*, John Wiley, New York, (1983).

[8] T. Granlund: *GNU MP: The GNU Multiple Precision Arithmetic Library*, 1.2 ed., (Dec. 1991). Available by anonymous ftp from `sics.se`.

[9] F. Harary and E. M. Palmer: *Graphical Enumeration*, Academic Press, (1973).

[10] J.-C. Hervé, B. Serpette and J. Vuillemin: *BigNum: A Portable and Efficient Package for Arbitrary-Precision Arithmetic*, Tech. Rep. 2, Digital Paris Research Laboratory, (May 1989).

[11] E. M. Reingold and J. S. Tilford: Tidier drawings of trees. *IEEE Trans. Softw. Eng. SE-7*, **2**: pp.223-228, (Mar. 1981).

[12] N. J. A. Sloane: *A Handbook of Integer Sequences*, Academic Press, (1973).

Kinematic and Power-Flow Analysis of Epicyclic Gear Drives

Nicola Pio Belfiore[1], *Ettore Pennestri*[2], *and Rosario Sinatra*[3]

Kinematic and power-flow analysis of epicyclic gear drives are basic tasks during the design of speed reducers. Classical textbooks on mechanical engineering (e.g. [2]) contain sections on this topic. The authors of this article developed methodologies and software for the analysis of gear drives [3], [4] and [5]. The aim was to provide the designer with simple and ready-to-use tools [6], [7], [8], and [9].

A first Pascal code for the kinematic and static force analysis was presented in 1989 [6]. Then, a general approach to power-flow analysis was developed [3] and also a second Pascal code completed [7].

However, only numerical solutions could be obtained. Traditionally high level languages as FORTRAN, Pascal and C have been mainly used in mechanical engineering, as *real number processors*. In other words, both the input and the output sets were supposed to be real numbers.

The implementation of analysis methodologies in Maple language makes it possible to automatically derive the solutions in an algebraic symbolic form. In our analysis the input phase consists of supplying the adjacency matrix of the graph corresponding to the kinematic structure under analysis. The Maple-generated outputs are the expressions for kinematic and power-flow analysis. For this reason the developed program allows a systematic comparison between different drive topologies [10].

The formulas deduced show how each gear ratio is related to overall speed reduction and power distribution inside the loops of the epicyclic gear train. It has been shown [5] that the amount of circulating power has a great influence on the mechanical efficiency of the speed reducer. Hence, the Maple program herein presented can conveniently support the engineer during the design of energy efficient gear drives.

A brief review of the theoretical bases

The adopted methodology is based on the correspondence between labeled graphs and epicyclic gear trains.

Graph theory traditionally begins in 1736, when Euler introduces a graph related to the Königsberg Bridges problem [1]. In this problem, an optimal path across a series of bridges on the Pregel river must be found in such a way that all the regions, two islands and two banks, are visited and each bridge is crossed only once. Now, each region may correspond to a vertex, while each bridge to an edge connecting two vertices. Euler found that the solution was impossible for that case and stated the condition for the existence of a solution. Graph theory has been considered as a very flexible and powerful tool in many fields as, for example, Physics, Economics, Chemistry, Electrical Engineering, Network theory, Optimization, and even Psychology. Among the various applications we find the very well known ones of Kirchhoff to electrical networks and of Cayley to the enumeration of chemical isomers.

A graph is an abstract object defined as a non-empty set of nodes and pairs of nodes. For the present application vertices correspond to links and edges to kinematic pairs. Each edge is labeled with different

[1] Department of Mechanics and Aeronautics, University of Rome "La Sapienza", Italy, belfiore@itcaspur.caspur.it
[2] Department of Mechanical Engineering, University of Rome "Tor Vergata", Italy, pennestri@vaxmec.mec.utovrm.it
[3] Institute of Machines, University of Catania, Italy, im@idm.ing.unict.it

characters depending on the nature of the kinematic pairs. For example, all the edges corresponding to geared pairs are labeled with the character g; the edges corresponding to revolute pairs are hierarchically labeled with the letters a, b, c, and so on, depending on their distance from the root vertex (i.e. frame link). As for the above mentioned hierarchical distribution of the *turning pair* labels, it must be pointed out that the removal of all the *geared edges* converts the graph into a spanning tree. In fact, by adding a single geared edge to that tree one obtains a graph with only one circuit. The set of all the circuits related to the spanning tree by the corresponding geared edge is therefore called set of *fundamental circuits*.

The correspondence between labeled graph and mechanisms is particularly convenient for the analysis of planetary gear trains. An elementary epicyclic train corresponds to a fundamental circuit and the relative gear carrier link can be automatically recognized as the *transfer vertex* of that circuit. The *transfer vertex* is the only vertex of the circuit which is incident to *turning pair* edges having different labels (see Figure 1). This makes possible the automatic generation of a system of linear equations for the kinematic analysis of a complex gear drive. In particular, such a system of equations is generated by writing down the Willis' formula for each gear pair (i.e. for each elementary train) [2]. Conversely, the power flow analysis is performed with the aid of a flow diagram (see Figure 2). In such a diagram, blocks (i.e. boxes), nodes (i.e. dots) and flows (i.e. plane lines) are represented. Blocks correspond to elementary epicyclic trains, nodes to links, and flows to powers flowing between the extremes of the lines (i.e. blocks or nodes). Each block has three gates: first mating gear (i.e. first end of the geared edge), second mating gear (i.e. second end of the geared edge) and gear carrier (i.e. the transfer vertex).

Since we have already solved the problem of finding the transfer vertices, we can build a graph whose vertices and edges respectively correspond to blocks (or nodes), and flows. The algebraic system of linear analysis equations can be set up by writing down a power balance equation for each node and two equations which relate the flowing powers to the gear ratios and the angular velocities. If one gate of the block is connected to the frame link then only one equation needs to be considered. References [3] and [4] report on such equations.

This second system of linear equations can also be solved in a symbolic form respect to the unknown power flowing through each line [3].

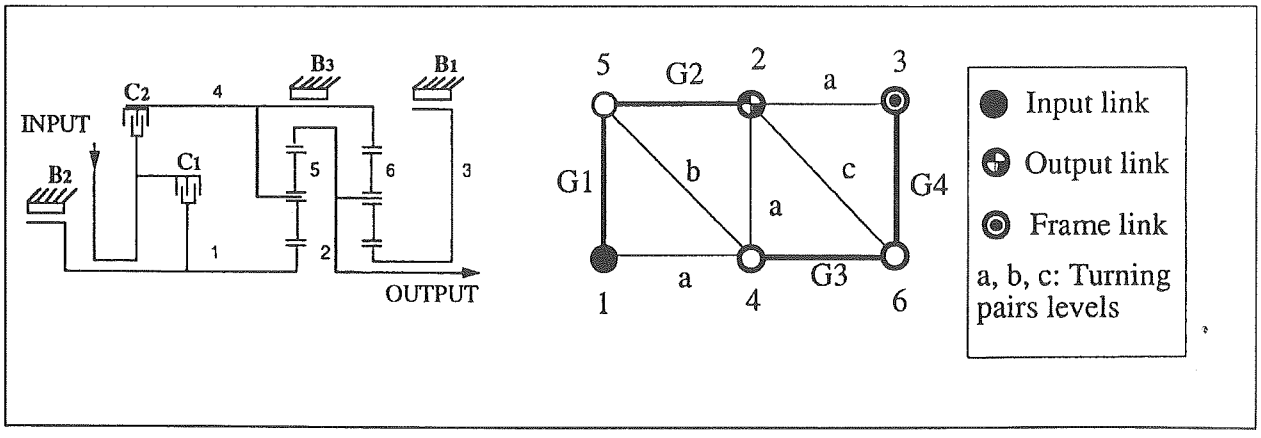


Figure 1: Epicyclic geat drive (left) and corresponding graph (right) (1st speed).

Description of the program

The proposed methodology has been implemented in Maple and the following step-by-step procedure developed.

1. Input phase.
2. Detection of the transfer vertices.
3. Detection of the set of unknown angular velocities.
4. Creation of the system of equations for the kinematic analysis.
5. Symbolic solution of the equations generated at step 4.
6. Build up of the the graph describing the power-flow (e.g. that depicted on the right of Figure 2).
7. Detection of the set of unknown powers.
8. Creation of the system of equations for the power-flow analysis.
9. Symbolic solution of the system generated at step 8.

Step 1: Input phase

The information requested consists of the number of vertices `n` and the labeled adjacency matrix `A` for the epicyclic train under estimation. Also to be specified are the frame (i.e. `frame`), input (i.e. `input`), and output (i.e. `output`) vertices. With reference to Figure 1, the following input segment was used for the analysis of the first speed of the *Hydra-Matic THM 440-T4* [11]. Symbols C1, C2 represent clutches, while B1, B2, and B3 are the brakes. In particular, when clutch C1 and brake B1 are actuated the gear train is in the first speed. The variable `gcont` denotes the total number of geared pairs. `VTA` is a `gcont`× 3 matrix of integers. Each row is composed of three integers: the first two denote the vertices corresponding to the mating gear links, while the third is the transfer vertex of the elementary epicyclic train relative to the fundamental circuit under consideration.

```
> n := 6:
  A := array(1..n,1..n,
    [ [ o , o , o , a , g , o ],
      [ o , o , a , a , g , c ],
      [ o , a , o , o , o , g ],
      [ a , a , o , o , b , g ],
      [ g , g , o , b , o , o ],
      [ o , c , g , g , o , o ] ]):
  tre_span := array(1..n,1..n):
  VTA := array(1..n,1..3):
  frame := 3:
  input := 1:
  output := 2:
```

Step 2: Detection of the transfer vertices

A single procedure has been written in order to find the transfer vertex for a given gear pair `i - j` and a given adjacency matrix `tre_span`. The procedure is named `vtransf` and is called as many times as the number of gear pairs.

```
> vtransf := proc(i,j,tre_span):
```

The first stage of the procedure deletes all the geared-pair edges. The character `o` stands for the null label (i.e. no connection for the two vertices). At the end of the segment matrix `tre_span` becomes the adjacency matrix of a spanning tree of the input graph.

```
for iloc from 1 to n do
  for jloc from 1 to n do
    if tre_span[iloc,jloc] = g then
      tre_span[iloc,jloc] := o
    fi
  od
od:
```

The second stage deletes all the leaves but vertices `i` and `j` from the tree `tre_span` obtained at the previous stage. By iterating this procedure the unique path connecting `i` and `j` (i.e. the fundamental circuit relative to the gear pair `i - j`) is computed and stored in the matrix `tre_span`.

```
ripeti := true:
while ripeti do
  ripeti := false:
  for iloc from 1 to n do
    grado := 0:
    for jloc from 1 to n do
      if tre_span[iloc,jloc] <> o then
        grado := grado + 1
      fi
    od:
    if grado = 1 then
      if (iloc <> i) and (iloc <> j) then
        jloc := 0:
        notrovato := true:
        while (jloc < n + 1) and notrovato do
          jloc := jloc + 1:
          if tre_span[iloc,jloc] <> o then
            notrovato := false
          fi
        od:
        tre_span[iloc,jloc] := o:
        tre_span[jloc,iloc] := o:
        ripeti := true
      fi
    fi
  od
od:
```

Now, the transfer vertex is the only vertex of the path which is incident to two edges having different labels.

```
for iloc from 1 to n do
  primavolta := true:
  grado := 0:
  for jloc from 1 to n do
    if tre_span[iloc,jloc] <> o then
      grado := grado + 1:
      if primavolta then
        primo := tre_span[iloc,jloc]:
        primavolta := false
      else
        secondo := tre_span[iloc,jloc]
      fi
    fi
  od:
  if (grado = 2) and (primo <> secondo) then
    vertex := iloc
  fi:
od:
vertex
end:
```

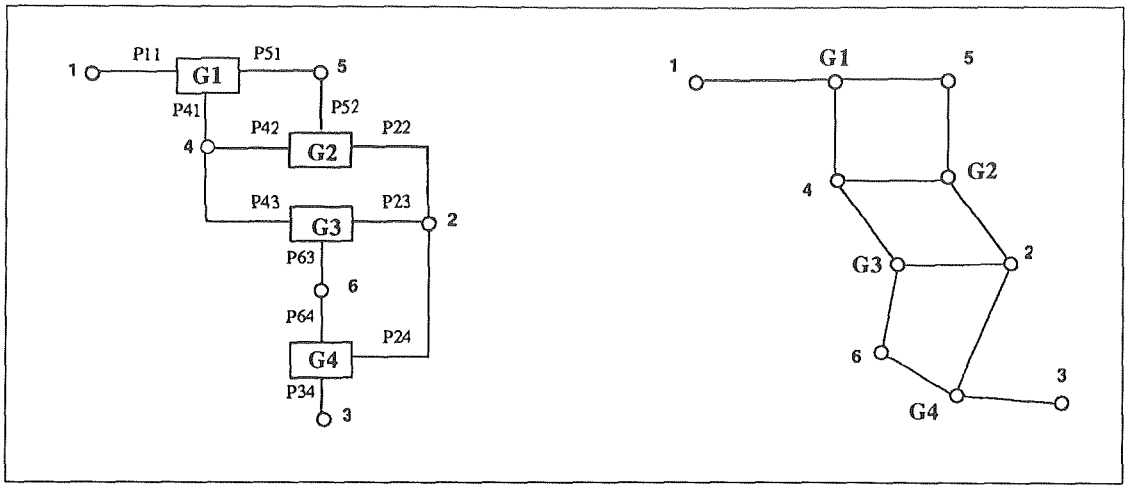


Figure 2: Power-flow diagram (left) and corresponding graph (right).

It must be pointed out that the algorithm described is simplified with respect to the general-purpose algorithms for graphs. In fact, the class of graphs under analysis is supposed to hold all the properties described in the previous section. General users may find programming convenient with the network package offered by the release 2 of Maple. The cited package is another powerful tool which allows creation and modification of graphs in a straightforward manner. Commands like `new()`, `cycle()`, and `complete()` immediately create new graphs while `addedge()` and `addvertex()` easily modify them. Our program was born with the first release of Maple V. However, the procedure `vtransf()` herein proposed seems novel even for the network package cited above.

Step 3: Detection of the set of unknown angular velocities

The angular velocities are denoted by the symbols `OM1`, `OM2`, ... The angular velocity of the frame link is set equal to 0. The detection of the set of unknowns is straightforward. In fact, all the angular velocities but the ones of the frame link (called `frame`) and the input link (called `input`) are unknown.

```
> varset := {}:
  for i from 1 to n do varset := varset union {OM.i} od:
  varset := varset minus {OM.frame}:
  varset := varset minus {OM.input}:
  print(varset);
```

Steps 4 and 5: Creation and solution of the system of equations for the kinematic analysis

The matrix `VTA` is generated by the following code:

```
> gcont := 0:
  for i from 1 to (n - 1) do
    for j from (i+1) to n do
      if A[i,j] = g then
        gcont := gcont + 1:
        for i1 from 1 to n do
          for j1 from 1 to n do
            tre_span[i1,j1] := A[i1,j1]
          od
        od:
        vt := vtransf(i,j,tre_span):
        VTA[gcont,1] := i: VTA[gcont,2] := j: VTA[gcont,3] := vt:
      fi
    od
  od:
  for i3 from 1 to gcont do print(VTA[i3,1],VTA[i3,2],VTA[i3,3]) od;
```

For our example, one obtains:

$$n := 6$$

$$A := \begin{bmatrix} o & o & o & a & g & o \\ o & o & a & a & g & c \\ o & a & o & o & o & g \\ a & a & o & o & b & g \\ g & g & o & b & o & o \\ o & c & g & g & o & o \end{bmatrix}$$

$$frame := 3$$

$$input := 1$$

$$output := 2$$

$$\{OM2, OM4, OM5, OM6\}$$

1, 5, 4

2, 5, 4

3, 6, 2

4, 6, 2

Everything is now ready for the generation and solution of kinematic analysis equations. `Rji` denotes the transmission ratio relative to gear pair `j - i`.

```
> for m from 1 to gcont do
    i := VTA[m,1]: j := VTA[m,2]: k := VTA[m,3]:
    eqn.m := OM.i - R.j.i * OM.j + (R.j.i - 1) * OM.k = 0
  od:
  OM.frame := 0:
  solutions := solve({eqn.(1..gcont)}, varset);
```

$$solutions := \{ OM2 = \frac{R63\,R52\,OM1}{\%1}, OM4 = \frac{(-R63 + R64)\,R52\,OM1}{\%1},$$

$$OM5 = \frac{OM1\,(-R64 - R63\,R52 + R64\,R52)}{\%1}, OM6 = -\frac{(R63 - 1)\,R52\,OM1}{\%1} \}$$

$$\%1 := -R63\,R52 - R64\,R51 + R64\,R52$$

Step 6: Build-up of the adjacency matrix for the graph describing the power-flow

During the implementation of this phase a further simplification of the original algorithm has been adopted. In fact, only a minor of the adjacency matrix of the graph corresponding to the diagram is needed. We briefly recall that such a graph, shown at the right of Figure 2, is conceptually different from the input graph of the epicyclic train of Figure 1. The simplification is possible because the graph of Figure 2 is a bipartite graph. In fact, its vertex set can be partitioned into two subsets containing, respectively, vertices corresponding to links and blocks. Each edge of the bipartite graph joins vertices belonging to different subsets. Hence one can arrange a boolean matrix `C` having `n` (i.e. number of nodes or links) rows and `gcont` (i.e. number of blocks or fundamental circuits) columns. Element `C[i,j]` is true if and only if node `i` is

connected to block j. In other words, C is a boolean look-up table which contains all the informations for the construction of the graph relative to the power flow diagram.

```
> C := array(1..n,1..gcont):
  for i from 1 to n do
    for j from 1 to gcont do
      C[i,j] := false
    od:
  od:
  for m from 1 to gcont do
    C[VTA[m,1],m] := true: C[VTA[m,2],m] := true: C[VTA[m,3],m] := true:
  od:
```

Step 7: Detection of the set of unknown powers

The information stored in matrix C can be used to build the set of unknown powers. Each line of the diagram depicting the power distribution is associated with a symbol P.i.m, that is the character P with two indices, ordinately denoting the node i and the block m connected by the line.

```
> powerset := {}:
  for i from 1 to n do
    if i <> frame then
      for m from 1 to gcont do
        if C[i,m] then
          powerset := powerset union {P.i.m}
        fi
      od
    fi
  od:
  powerset := powerset union {Pout}:
```

Steps 8 and 9: Creation and solution of the system of equations for the power-flow analysis

The set of equations relative to the block was created according to the theoretical bases discussed previously [3 and 4].

If none of the adjacent nodes is the frame link then, for each block, two equations are written; otherwise only one equation is added. The angular velocities are denoted with O1, O2, ... instead of OM1, OM2, ... to avoid automatic substitutions of the values obtained during the kinematic analysis.

Construction of the group of equations relative to boxes:

```
> equcont := 0:
  for m from 1 to gcont do
    i := VTA[m,1]; j := VTA[m,2]; k := VTA[m,3];
    equazji := P.j.m = - R.j.i * P.i.m * ( O.j / O.i):
    equazkj := P.k.m = ( (1 - R.j.i) * O.k * P.j.m) / (R.j.i * O.j):
    equazki := P.k.m = ( (R.j.i - 1) * O.k * P.i.m) / ( O.i):
    if C[frame,m] then equcont := equcont + 1:
      if frame = i then equf.equcont := equazkj
      elif frame = j then equf.equcont := equazki
      elif frame = k then equf.equcont := equazji
      fi
    else
      equcont := equcont + 1: equf.equcont := equazji:
      equcont := equcont + 1: equf.equcont := equazki
    fi
  od:
```

For each node a simple power balance equation is written for all the power flowing from and to that node. Input and output power must be also considered during the construction of these equations.

Construction of the group of equations relative to nodes:

```
> for m from 1 to n do
    if m <> frame then
      equcont := equcont + 1:
      termine := 0:
      if m = input then
        termine := termine + Pin
      fi:
      if m = output then
        termine := termine - Pout
      fi:
      for mm from 1 to gcont do
        if C[m,mm] then
          termine := termine + P.m.mm
        fi
      od:
      equf.equcont := termine = 0
    fi
  od:
  for m from 1 to equcont do print(equf.m) od:
```

Finally the system of equations can be solved by using one Maple instruction. The output areas herein reported show the set of unknowns (called `powerset`), the system of equations, and the solutions obtained for the case shown in Figures 1 and 2.

```
> powers := solve({equf.(1..equcont)}, powerset);
```

$$P51 = -\frac{R51\,P11\,O5}{O1}$$

$$P41 = \frac{(R51-1)\,O4\,P11}{O1}$$

$$P52 = -\frac{R52\,P22\,O5}{O2}$$

$$P42 = \frac{(R52-1)\,O4\,P22}{O2}$$

$$P23 = \frac{(1-R63)\,O2\,P63}{R63\,O6}$$

$$P64 = -\frac{R64\,P44\,O6}{O4}$$

$$P24 = \frac{(R64-1)\,O2\,P44}{O4}$$

$$Pin + P11 = 0$$

$$-Pout + P22 + P23 + P24 = 0$$

$$P41 + P42 + P44 = 0$$

$$P51 + P52 = 0$$

$$P63 + P64 = 0$$

$$\text{powers} := \{ P23 = \frac{O2\,R64\,Pin\,(-R52-R63\,R51+R63\,R52+R51)}{O1\,R52\,R63},\ P44 = -\frac{O4\,Pin\,(-R51+R52)}{O1\,R52},$$

$$P42 = \frac{O4\,Pin\,R51R52-1}{O1\,R52},\ P24 = -\frac{O2R64-1Pin-R51+R52}{O1\,R52},\ P22 = \frac{Pin\,R51O2}{O1\,R52},$$

$$P52 = -\frac{R51\,Pin\,O5}{O1},\ P64 = \frac{R64\,Pin\,(-R51+R52)\,O6}{O1\,R52},\ P11 = -Pin,$$

$$Pout = -\frac{Pin\,O2\,(-R63\,R52-R64\,R51+R64\,R52)}{O1\,R52\,R63},\ P41 = -\frac{O4\,Pin\,(R51-1)}{O1},$$

$$P63 = -\frac{R64\,Pin\,(-R51+R52)\,O6}{O1\,R52},\ P51 = \frac{R51\,Pin\,O5}{O1}\ \}$$

The full listing of the Maple program developed is available upon request.

Acknowledgement

The first two authors wish acknowledge financial support from Consiglio Nazionale delle Ricerche-Progetto Finalizzato Robotica, Grant n.93.00930.PF67.

References

[1] F. Harary: *Graph Theory*, Addison-Wesley Publ. Co., Reading, MA, (1972).

[2] Joseph E. Shigley and John J. Uicker: *Theory of Machines and Mechanisms*, McGraw-Hill Book Co., New York, NY, pp. 305 - 321, (1980).

[3] Ettore Pennestrì: On The Automatic Design Analysis of Gear-Trains, *Doctoral Dissertation*, Columbia University NY, University Microfilms, Ann Arbor MI, Order No. 9127952).

[4] Ettore Pennestrì and Ferdinand Freudenstein: A Systematic Approach to Power-Flow and Static-Force Analysis in Epicyclic Spur-Gear Trains, *ASME Journal of Mechanical Design*, **115**, pp. 639 - 644 (1993).

[5] Ettore Pennestrì and Ferdinand Freudenstein: The Mechanical Efficiency of Epicyclic Gear Trains, *ASME Journal of Mechanical Design*, **115**, pp. 645 - 651, (1993).

[6] Nicola Pio Belfiore and Ettore Pennestrì: Kinematic and Static Force Analysis of Epicyclic Gear Trains, *Proc. 1st Nat. Conf. on Applied Mechanisms and Robotics*, Cincinnati, Ohio, Paper No. 89AMR-6B, (1989).

[7] Nicola Pio Belfiore, Ettore Pennestrì, and Antonio Belmontesi: An Integrated System for Power-Flow Analysis and Evaluation of Mechanical Efficiency of Epicyclic Gear Trains, (in Italian), *Proc. XI Nat. Cong. AIMETA, Sez. Meccanica delle Macchine*, Trento, Italy, pp. 47 - 52, (1992).

[8] Ettore Pennestrì, Rosario Sinatra, and Nicola Pio Belfiore: Multiple-Speed Planetary Gear Systems with Kinematic and Power-Flow Analysis, *Internal Report, University of Rome Tor Vergata*, Rome, Italy, (1993).

[9] Ettore Pennestrì, Rosario Sinatra, and Nicola Pio Belfiore: A Catalogue of Automative Transmissions with kinematic and Power-Flow Analysis, *Proc. 3rd Nat. Conf. on Applied Mechanisms and Robotics, Paper No. AMR-93-057*, Cincinnati, Ohio, (1993).

[10] Anders Hedman: Transmission Analysis: Automatic Derivation of Relationships, *Proc. ASME Int. Power Transmission and Gearing Conf.*, DE-Vol. 43-1, Scottsdale, AZ, pp. 259 - 266, (1992).

[11] Lung-Wen Tsai, E.R. Maki, T. Liu, and N.G. Kapil: The Categorization Of Planetary Gear Trains For Automatic Transmission According To Kinematic Topology, *SAE Technical Paper Series, FISITA 88*, Washington DC, (1988).

On Solving the McConnell Equations in Biochemistry

Johannes Grotendorst, Paul Jansen, and Siegfried M. Schoberth

In recent years, in vivo nuclear magnetic resonance (NMR) spectroscopy has allowed us to measure rate constants of transport and diffusion across living cell membranes [1-2]. One of the authors (S.M.S.) is interested in a special NMR technique — inversion transfer — to study bacterial systems [3]. The theoretical basis for the analysis of inversion transfer experiments is a system of differential equations first formulated by McConnell [4]. These equations describe the rate of change of nuclear spin magnetization of a single nuclear species which is transferred back and forth between two different magnetic environments (A, B) by kinetic processes. In this article we show how the McConnell equations, a linear inhomogeneous system of differential equations with constant coefficients, can be solved elegantly by using symbolic matrix algebra only, in particular by using matrix exponentials. This method utilizes the special structure of the ODE system and therefore is faster and more direct than simply using dsolve, the solver for differential equations in Maple. It is also an independent method compared to the solution techniques found in the literature [5-7]. In addition, methods of mixed symbolic-numeric type for the determination of the formal parameters involved in the analytical solutions are described.

This application is presented as a Maple worksheet, a new feature of the graphical user interface in Maple V Release 2, for creating mathematically live documents. It combines Maple input, output, text and graphics in one easily accessible document. The goal of this worksheet is to illustrate the valuable assistance of symbolic computation in modeling and solving a special mathematical problem in Biochemistry.

The McConnell equations as formulated by Led and Gesmar [5] to study chemical exchange rates are given as follows

```
> eq1 := diff(MA(t),t) = -k[A1]*MA(t) + k[-1]*MB(t) + k[A];
  eq2 := diff(MB(t),t) = k[1]*MA(t) - k[B1]*MB(t) + k[B];
```

$$eq1 := \frac{\partial}{\partial t}\mathrm{MA}(t) = -k_{\mathrm{A1}}\mathrm{MA}(t) + k_{-1}\mathrm{MB}(t) + k_A$$

$$eq2 := \frac{\partial}{\partial t}\mathrm{MB}(t) = k_1\mathrm{MA}(t) - k_{\mathrm{B1}}\mathrm{MB}(t) + k_B$$

Here, MA(t) and MB(t) describe the time-dependent peak heights of the magnetic resonance signals in A and B, respectively. These signals are measured in NMR experiments. The parameters k[1] and k[-1], interpreted as indexed names in Maple, characterize the first-order rate constants for the forward and reverse reactions. The meaning of the other parameters will become clear in the sequel. From linear algebra we know that linear systems of differential equations with constant coefficients can be solved with the help of matrix exponentials [8]. To do it in Maple we first have to define the appropriate matrices.

```
> with(linalg): #load the linear algebra package
  A := matrix([[-k[A1], k[-1]], [k[1], -k[B1]]]);
  C := vector([k[A], k[B]]);
  Y0 := vector([MA0, MB0]);

Warning: new definition for   norm
Warning: new definition for   trace
```

$$A := \begin{bmatrix} -k_{A1} & k_{-1} \\ k_1 & -k_{B1} \end{bmatrix}$$

$$C := \begin{bmatrix} k_A & k_B \end{bmatrix}$$

$$Y0 := \begin{bmatrix} MA0 & MB0 \end{bmatrix}$$

Applying the Maple function for the evaluation of matrix exponentials leads to the following solution of the inhomogeneous ODE system:

```
> F := multiply(exponential(A,t), Y0):
  f := s -> multiply(exponential(A, t-s), C);
  P := [Int(f(s)[1], s=0..t), Int(f(s)[2], s=0..t)];
```

$$f := s \rightarrow \text{multiply}(\text{exponential}(A, t - s), C)$$

$$P := \left[\int_0^t -\frac{1}{2} \left(-k_A \%3 \sqrt{\%1} - k_A k_{A1} \%2 + k_A k_{A1} \%3 + k_A k_{B1} \%2 - k_A k_{B1} \%3 \right.\right.$$

$$\left. - k_A \sqrt{\%1} \%2 + 2 k_{-1} k_B \%2 - 2 k_{-1} k_B \%3 \right) / \sqrt{\%1}\, ds,$$

$$\int_0^t \frac{1}{2} \left(-2 k_1 k_A \%2 + 2 k_1 k_A \%3 + k_B \%3 \sqrt{\%1} + k_B k_{B1} \%2 - k_B k_{B1} \%3 \right.$$

$$\left.\left. - k_B k_{A1} \%2 + k_B k_{A1} \%3 + k_B \sqrt{\%1} \%2 \right) / \sqrt{\%1}\, ds \right]$$

$$\%1 := k_{A1}^2 - 2 k_{A1} k_{B1} + k_{B1}^2 + 4 k_{-1} k_1$$

$$\%2 := e^{\left(-1/2 \left(k_{A1} + k_{B1} + \sqrt{\%1} \right) (t - s) \right)}$$

$$\%3 := e^{\left(-1/2 \left(k_{A1} + k_{B1} - \sqrt{\%1} \right) (t - s) \right)}$$

Here, the % label signifies common subexpressions in the output, a way Maple makes output more compact. The matrix exponential of a d x d square matrix A is defined by:

$$\exp(A) = I + A + \frac{1}{2!}A^2 + \frac{1}{3!}A^3 + \ldots + \frac{1}{n!}A^n + \ldots$$

where I is the identity matrix and the matrix series on the right hand side can be considered as a d^2 (scalar) series, one for each of the elements of exp(A) [8]. Now, the solution of the inhomogeneous ODE system is obtained by adding the solution F of the corresponding homogeneous ODE system and the particular solution P.

```
> S1:= add(F, map(value,P)):
```

It should be noted that the solution of the equivalent one dimensional ODE is found in a formally identical manner. The correctness of the one dimensional solution is easily verified.

```
> exp(a*t)*y0 + int(exp(a*(t-s))*c, s=0..t):
  evalb(expand(diff(",t) = a*" + c));
```

true

If matrix A is invertible then the solution of the inhomogeneous ODE system can be computed with matrix algebra only. We have:

```
> AC := multiply(inverse(A), C):
  S2 := add(multiply(exponential(A,t), add(Y0, AC)), -AC):
```

Again, this direct solution can be motivated by the one dimensional case:

```
> dsolve({diff(y(t),t)=a*y(t)+c,y(0)=y0}, y(t));
```

$$y(t) = -\frac{c}{a} + e^{(at)}\left(\frac{c}{a} + \mathrm{y0}\right)$$

Comparing the solutions S1 and S2 yields:

```
> map(simplify, add(S1,-S2));
```

$$[0\ 0]$$

Next, we introduce physical boundary conditions to obtain a special solution. Let MeA and MeB denote the unperturbed equilibrium magnetizations of A and B respectively, i.e. the limits of MA(t) and MB(t) for t -> +infinity. These limits are given by the components of the constant vector -AC in solution S2, provided that both eigenvalues of A are negative.

```
> MeA = -AC[1], MeB = -AC[2]; #(*)
```

$$\mathrm{MeA} = \frac{k_A k_{B1} + k_{-1} k_B}{k_{A1} k_{B1} - k_{-1} k_1},\ \mathrm{MeB} = \frac{k_A k_1 + k_B k_{A1}}{k_{A1} k_{B1} - k_{-1} k_1}$$

Solving for k[A] and k[B] yields:

```
> solve({"}, {k[A], k[B]});
  assign("):
```

$$\{k_A = -k_{-1}\mathrm{MeB} + k_{A1}\mathrm{MeA},\ k_B = \mathrm{MeB}\, k_{B1} - k_1 \mathrm{MeA}\}$$

Inserting these expressions into the system of ODEs and collecting coefficients results in the following system of equations:

```
> eq1 := collect (eq1, [k[A1], k[-1]]);
  eq2 := collect (eq2, [k[B1], k[1]]);
```

$$eq1 := \frac{\partial}{\partial t}\mathrm{MA}(t) = (-\mathrm{MA}(t) + \mathrm{MeA})\, k_{A1} + (\mathrm{MB}(t) - \mathrm{MeB})\, k_{-1}$$

$$eq2 := \frac{\partial}{\partial t}\mathrm{MB}(t) = (-\mathrm{MB}(t) + \mathrm{MeB})\, k_{B1} + (\mathrm{MA}(t) - \mathrm{MeA})\, k_1$$

Now, the inhomogeneous ODE system is transformed to a homogeneous system by $\mathrm{MA}(t) \rightarrow \mathrm{MA1}(t) + \mathrm{MeA}$ and $\mathrm{MB}(t) \rightarrow \mathrm{MB1}(t) + \mathrm{MeB}$:

```
> neweq1:=eval(subs(MA(t)=MA1(t)+MeA, MB(t)=MB1(t)+MeB, eq1)); #(**)
  neweq2:=eval(subs(MA(t)=MA1(t)+MeA, MB(t)=MB1(t)+MeB, eq2));
```

$$neweq1 := \frac{\partial}{\partial t}\mathrm{MA1}(t) = -\mathrm{MA1}(t) k_{A1} + \mathrm{MB1}(t) k_{-1}$$

$$neweq2 := \frac{\partial}{\partial t}\mathrm{MB1}(t) = -\mathrm{MB1}(t) k_{B1} + \mathrm{MA1}(t) k_1$$

For the solution of this system we use matrix exponentials again.

```
> time1   := time():
  Y1      := vector([MA10, MB10]):
  fundsys := multiply(exponential(A,t), Y1):
  # adding the boundary values
  linsol  := add(fundsys, vector([MeA, MeB]));
  time1   := time() - time1;
```

$$linsol := \left[-\frac{1}{2}(-MA10\,\%3\sqrt{\%1} - MA10\,k_{A1}\%2 + MA10\,k_{A1}\%3 + MA10\,k_{B1}\%2 - MA10\,k_{B1}\%3 - MA10\sqrt{\%1}\,\%2 + 2k_{-1}MB10\,\%2 - 2k_{-1}MB10\,\%3)/\sqrt{\%1}\,ds + MeA \right.$$

$$\left. \frac{1}{2}(-2k_1MA10\,\%2 + 2k_1MA10\,\%3 + MB10\,\%3\sqrt{\%1} + MB10\,k_{B1}\%2 - MB10\,k_{B1}\%3 - MB10\,k_{A1}\%2 + MB10\,k_{A1}\%3 + MB10\sqrt{\%1}\,\%2)/\sqrt{\%1}\,ds + MeB \right]$$

$$\%1 := k_{A1}^2 - 2k_{A1}k_{B1} + k_{B1}^2 + 4k_{-1}k_1$$

$$\%2 := e^{\left(-1/2\left(k_{A1}+k_{B1}+\sqrt{\%1}\right)t\right)}$$

$$\%3 := e^{\left(-1/2\left(k_{A1}+k_{B1}-\sqrt{\%1}\right)t\right)}$$

$$time1 := .310$$

Thus, under condition (*) the solution of the original inhomogeneous ODE system is simplified to a sum where one term is the solution of the homogeneous ODE system (**) and the other term, usually the particular solution, is constant. Applying the collect function recursively to each component of the vector linsol provides a compact analytical solution of the following general form:

MA(t) = C1 * exp(lam1 * t) + C2 * exp(lam2 * t) + MeA
MB(t) = C3 * exp(lam1 * t) + C4 * exp(lam2 * t) + MeB

```
> linsol := map(collect, linsol, [%2, %3, MA10, MB10]);
```

$$linsol := \left[\left(-\frac{1}{2}\frac{(-k_{A1} - \sqrt{\%1} + k_{B1})\,MA10}{\sqrt{\%1}} - \frac{k_{-1}MB10}{\sqrt{\%1}} \right) e^{\left(-1/2\left(k_{A1}+k_{B1}+\sqrt{\%1}\right)t\right)} \right.$$

$$+ \left(-\frac{1}{2}\frac{(-k_{B1} + k_{A1} - \sqrt{\%1})\,MA10}{\sqrt{\%1}} - \frac{k_{-1}MB10}{\sqrt{\%1}} \right) e^{\left(-1/2\left(k_{A1}+k_{B1}-\sqrt{\%1}\right)t\right)} + MeA$$

$$\left(-\frac{k_1MA10}{\sqrt{\%1}} + \frac{1}{2}\frac{(k_{B1} - k_{A1} + \sqrt{\%1})\,MB10}{\sqrt{\%1}} \right) e^{\left(-1/2\left(k_{A1}+k_{B1}+\sqrt{\%1}\right)t\right)}$$

$$\left. + \left(\frac{k_1MA10}{\sqrt{\%1}} + \frac{1}{2}\frac{(k_{A1} - k_{B1} + \sqrt{\%1})\,MB10}{\sqrt{\%1}} \right) e^{\left(-1/2\left(k_{A1}+k_{B1}-\sqrt{\%1}\right)t\right)} + MeB \right]$$

$$\%1 := k_{A1}^2 - 2k_{A1}k_{B1} + k_{B1}^2 + 4k_{-1}k_1$$

It is easily verified that the parameters lam1 and lam2 are the eigenvalues of the coefficient matrix of the homogeneous ODE system. We have:

```
> eigenvals(A);
```

$$-\frac{1}{2}k_{A1}-\frac{1}{2}k_{B1}+\frac{1}{2}\sqrt{k_{A1}^2-2k_{A1}k_{B1}+k_{B1}^2+4k_{-1}k_1},$$
$$-\frac{1}{2}k_{A1}-\frac{1}{2}k_{B1}-\frac{1}{2}\sqrt{k_{A1}^2-2k_{A1}k_{B1}+k_{B1}^2+4k_{-1}k_1}$$

Next, we solve the homogeneous ODE system (**) by applying the Maple function dsolve, the solver for differential equations in Maple.

```
> time2 := time():
  start := MA1(0) = MA10, MB1(0) = MB10:
  fcns := {MA1(t), MB1(t)}:
  dsol := dsolve({neweq1, neweq2, start}, fcns):
  assign(dsol):
  #adding the boundary values
  dsol := vector([MA1(t) + MeA, MB1(t) + MeB]):
  time2 := time() - time2;
  speedup := time2/time1;
```

$$time2 := 23.810$$

$$speedup := 76.80645161$$

Thus, dsolve needs much more time to find the solution compared to the method using the matrix exponentials. This is explained by the more general algorithms used in dsolve which work for nonlinear equations and may not be efficient for homogeneous linear systems. The time (in seconds) was measured on a RS/6000-32H workstation. Comparing the result of dsolve with that found by using matrix exponentials yields:

```
> map(simplify, add(dsol, -linsol));
```

$$[0\ 0]$$

If we define the parameters:

```
> MA10 := MA0 - MeA: MB10 := MB0 - MeB:
  k[A1] := 1/T[A1] + k[1]: k[B1] := 1/T[B1] + k[-1]:
```

we get analytical solutions for MA(t) and MB(t) which depend on the eight parameters MA0, MB0, MeA, MeB, T[A1], T[B1], k[1], k[-1]. Here, 1/T[A1] and 1/T[B1] denote the relaxation rates of the spins in the two sites, whereas MA0 and MB0 denote the initial values of MA(t) and MB(t) at time 0. Usually, the parameters involved in MA(t) and MB(t) are determined by a nonlinear least-squares analysis, i.e. by fitting the model parameters to experimental data for MA(t) and MB(t) obtained at different values of time [5-7]. The symbolic evaluation of the Jacobian matrix, required for the numerical fit program, is readily done with the help of the Maple procedure jacobian. For the translation into optimized FORTRAN code and the generation of a driver program for the fitting routine we used Macrofort [9], a Maple package for FORTRAN code generation. The numerical parameter-fitting itself is accomplished by the ACM algorithm NL2SOL [10] which is based on the Levenberg-Marquardt algorithm and which needs analytic Jacobian matrices as input. The Macrofort program is capable to construct complete and ready to compile FORTRAN code for a given set of functions and parameters.

Now, for plotting the functions MA(t) and MB(t) we substitute special fitted values obtained from NMR data to investigate transport processes in a special biological system [3].

```
> k[1] := 6.6: k[-1] := 10.5:
  MeA := 153.5: MeB := 78.8:
  MA0 := 130.4: MB0 := 27.08:
  T[A1] := 1.4: T[B1] := 1.8:
```

The parameter values lead to the following negative eigenvalues of matrix A:

```
> eigenvals(A);
```

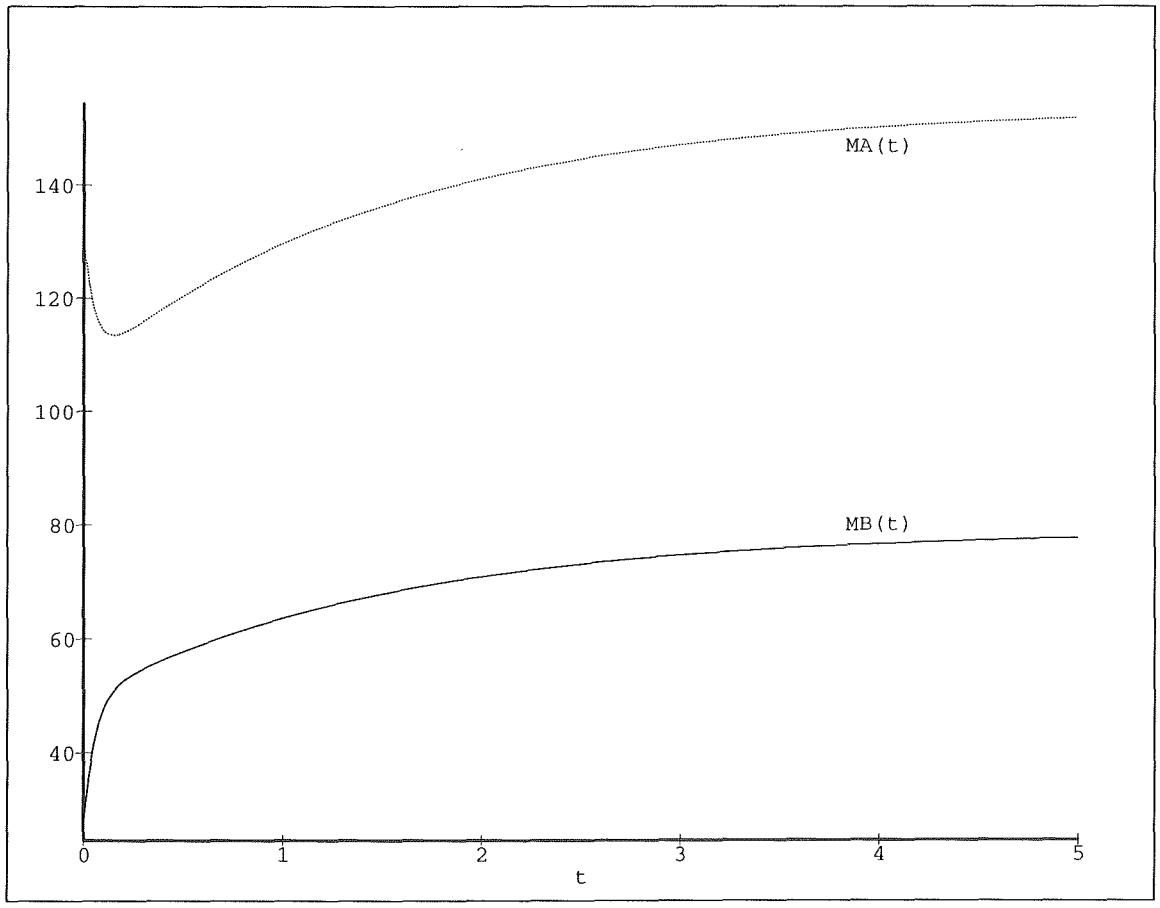
$$-.652671519, \quad -17.71716975$$

The given boundary conditions of MA(t) and MB(t) for t -> +infinity are reproduced by:

```
> map(limit, linsol, t=+infinity);
```

$$[153.5000000 \ 78.80000000]$$

```
> with(plots):
  pl := plot({seq(linsol[i], i=1..2)}, t=0..5):
  t1 := textplot([4, 79, 'MB(t)'], align=ABOVE):
  t2 := textplot([4, 147, 'MA(t)']):
  display(pl, t1, t2);
```



Acknowledgements

The authors would like to thank Edgar Wermuth for his help to find the explict solution in terms of matrix exponentials. Valuable comments of the referee for improving the readability of the article are acknowledged.

References

[1] K. Kirk: *NMR Biomed,* **3:** pp. 1-16, (1990).

[2] P. W. Kuchel: *NMR Biomed*, **3:** pp. 102-119, (1990).

[3] S. M. Schoberth, A. A. de Graaf, B. E. Chapman, P. W. Kuchel, and H. Sahm: *XV International Conference on Magnetic Resonance in Biological Systems, Jerusalem, Israel, August 16-21*, (1993).

[4] H. M. McConnell: *J. Chem. Phys.*, **28:** pp. 430-431, (1958).

[5] J. J. Led and H. Gesmar: *J. Magn. Res.*, **49:** pp. 444-464, (1982).

[6] P. W. Kuchel and B. E. Chapman: *J. Theor. Biol.*, **105:** pp. 569-589, (1983).

[7] G. Robinson, B. E. Chapman, and P. W. Kuchel: *Eur. J. Biochem.*, **143:** pp. 643-649, (1984).

[8] P. Hartman: *Ordinary Differential Equations*, John Wiley & Sons, Inc., (1964).

[9] C. Gomez: *Macrofort: a FORTRAN code generator in Maple*, Maple Share Library.

[10] J. E. Dennis, D. M. Gay, and R. E. Welsch: *ACM Trans. Math. Software*, Vol. 7, No. 3, (1981).

Modeling Snow Transport, An Application of Maple

Christian W. U. Hoffmann[1]

The Problem

When modeling physical problems by building a scaled version of the real thing, one has to deal with the 'scaling problem'. This arises because the different quantities in the model are multiples of the original quantities, and the factors used come from physical considerations. In principle, all these factors are powers of the scaling factor λ = lengths in the original divided by lengths in the model. If, for example, we want to model rainfall, scaling down linear dimensions is not enough, the viscosity of the air must be scaled, too, otherwise raindrops will behave very differently. The scaling factors will usually be derived by physical considerations [1]. For most models, however, there will remain 'misfits', as not all the desired quantities can be made to scale as desired. It is then the art of practical modeling to seek a compromise and extract the desired information.

In our example, an application of snow mechanics [2], the transport of snow by wind immediately above the snow cover is an important process for avalanche genesis. This transport is simulated at a smaller scale with real snow. Here three similarities are considered:

1. geometrical, i.e. proportions of lengths are preserved. (height of fence, diameter of snow grains, characteristic length, and others.)
2. kinematical, i.e. proportions of velocities are preserved. (wind speed, shear stress speed, speed of sound, sinking speed of snow grains, etc)
3. dynamical, i.e. proportions of forces are preserved. (gravitational force, flow resistance of snow grains)

These similarities cannot be satisfied simultaneously. But by scaling other factors, like densities, pressure, Reynolds' number, threshold shear velocity, cohesion, and others, one can find a way to build realistic models.

We define the following quantities, where variables in reality will be indexed by P (for prototype), those in the model by M:

- λ = geometrical scale factor = $\text{lengths}_P / \text{lengths}_M$ = 30
- ρ = density of air snow mixture
- q_s = transport rate in the 'saltation' layer (here the wind picks up snow)
- u = velocity
- u^* = shear velocity
- u_t^* = threshold shear velocity

[1] Federal Institute for Forest, Snow, and Landscape Research, CH-8903, Birmensdorf, Switzerland. hoffmann@wsl.ethz.ch

In [2] it is shown that $u_{t_P}^* = \sqrt{\lambda}\, u_{t_M}^*$, $u_P(x_P) = \sqrt{\lambda} u_M(x_M)$, $\rho_P(x_P) = \rho_M(x_M)$, so that with $dx_P = \lambda dx_M$ we can derive from the physical definition of

$$q_s = \int u(x)\rho(x)dx$$

the scaling equation

$$q_{s_P} = \lambda^{3/2} q_{s_M} \tag{1}$$

But from empirical equation describing snow transport, we have a condition conflicting with this, namely $q_{s_P} = \lambda q_{s_M}$. Thus we know that we cannot satisfy (1) and this condition simultaneously (for general u_M). But since u occurs in the defining integral, we can study the behavior of

$$q_{s_P} = z\lambda^{3/2} q_{s_M} \tag{2}$$

in the vicinity of z=1, say between 0.6 and 1.4, to find for given λ and u_M^* , where the transport rate q_{s_P} will be simulated to within 40%.

By 'looking closely' at (2) we find that it can best be analyzed by defining the following normalizing constants α and β and the variables x, y (, and z from above):

- $\alpha = \sqrt{\lambda}\, u_M^*$, $\beta = u_M^*$
- $x = u_{t_P}^* / \alpha > 0$, $y = u_{t_M}^* / \beta > 0$

With these abbreviations, (2) can be transformed into the zeros of the function

$$f(x, y, z) = x(x^2 - 1) - \sqrt{\lambda} \cdot zy(y^2 - 1) \tag{3}$$

Solution

We examine the graph of the function (3) $f(x, y, z) = 0$ for different values of z. Our interest lies in inspecting the regions between the curves $z=z_0$,

$$z_0 \in \{0.6, 0.8, 1.0, 1.2, 1.4\} \tag{4}$$

in the rectangle $0 < x < 1$, $0 < y < 1$, since other values do not have a physical meaning. From equation (3) we get by differentiation

$$(3x^2 - 1)\, dx - \sqrt{\lambda} z \cdot (3y^2 - 1)\, dy = 0 \tag{5}$$

and, also from (3) by solving for z

$$z = \frac{x(x^2 - 1)}{\sqrt{\lambda} y (y^2 - 1)} \tag{6}$$

Let us do some elementary 'curve discussion'. Let (x, y, z) be a point in three-dimensional space. By elementary considerations we get from (3) and (6) the following loci satisfying (3):

$$\begin{array}{lll} (0, 0, z) & (0, y, 0) & (x, 0, \infty) \\ (1, 0, z) & (1, y, 0) & (x, 1, \infty) \\ (0, 1, z) & (1, 1, z) & \end{array} \tag{7}$$

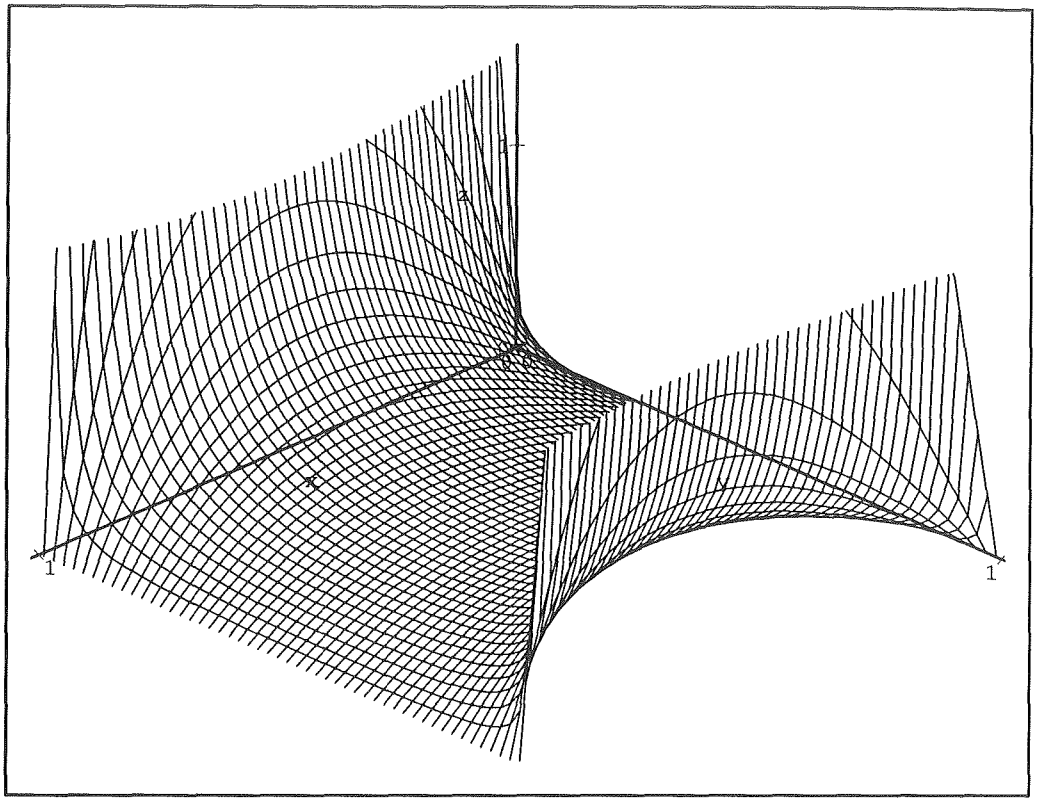


Figure 1: Oblique view of $z(x,y)$, with λ=10.

The following piece of MapleV2 code generates Figure 1, the graph of z in the desired rectangle.

```
>lambda:=10: f(x,y,z):=x*(x^2-1)-sqrt(lambda)*y*z*(y^2-1):
>ze:=solve(f(x,y,z)=0,z):
>with(plots): str:=convert(lambda,string):
>                  # 0.01, 0.99: to avoid infinities of z at y=0, 1
>plot3d(ze,x=0..1,y=0.01..0.99,axes=NORMAL,view=0..1.5,grid=[50,50],
>       orientation=[40,40],tickmarks=[2,2,2],labels=['x','y','z'],
>       title=`lambda=`.str);
```

Restriction of z to the values of (4) gives four special points (z will be dropped in the following):

$$(0, 0),\ (1, 0),\ (0, 1),\ (1, 1) \tag{8}$$

From (5) we get the loci of special values for the derivative:

$$\begin{aligned} \text{vertical tangents:} \quad & dx = 0: \quad (x, 1/\sqrt{3}) \\ \text{horizontal tangents:} \quad & dx = 0: \quad (1/\sqrt{3}, y) \\ \text{saddle points:} \quad & dx = dy = 0: \quad (1/\sqrt{3}, 1/\sqrt{3}) \end{aligned}$$

Horizontal (*vertical*) tangents mean that there is a constant (*mis*)scaling of q_{s_P} over a range of $u^*_{t_P}$ ($u^*_{t_M}$). In the vicinity of the saddle point the constant (*mis*)scaling is found for a broader range of both $u^*_{t_P}$ and $u^*_{t_M}$.

The condition for the saddle point to lie on the graph of $z=z_0$, and the condition that there exists a straight solution through (0, 0), prove to be identical, namely

$$z_0 = 1/\sqrt{\lambda} \tag{9}$$

This straight line automatically runs through (1, 1).

With the abbreviation $C = 1/(\sqrt{\lambda}z)$ the slopes dy/dx at the special points (8) are

$$\frac{dy}{dx}(0, 0) = C$$

$$\frac{dy}{dx}(0, 1) = -C/2$$

$$\frac{dy}{dx}(1, 0) = -2C$$

$$\frac{dy}{dx}(1, 1) = C$$

A qualitative trace of (6) $z(x, y) = z_0$ can now be sketched: For $C < 1$ the graph will connect the pairs of points (0, 0), (1, 0), and (0, 1), (1, 1); for $C > 1$ the pairs of points (0, 0), (0, 1), and (1, 0), (1, 1) are connected; for $C=1$ we have the saddle point at the place where the straight line through (0, 0), (1, 1), and the curved line through (0, 1), (1, 0) meet.

The first course to trace the exact curve $z=z_0$ is to take the z values required in (4), specify some values for y, and solve (3) for x. This procedure proves to be awkward because (3) may have two, one, or zero solutions for the value of y considered. Bookkeeping might be nontrivial.

The second course is to integrate (5) parametrically, starting from the special points. From (5) we get

$$dx = -\sqrt{\lambda}z\,(3y^2 - 1)\,dt, \qquad dy = -(3x^2 - 1)\,dt \tag{10}$$

We integrate for $t > 0$ numerically, starting from (0, 0) and (1, 1). One of the two curves will pass through (0, 1), the other through (1, 0), depending on the value of C, see above. The minus signs were introduced into (10) so that the integration algorithm will start off into the correct direction. The Maple program given below implements this procedure.

In the special case (9) the integration will break down in the vicinity of the saddle point because of $dx \to 0$, $dy \to 0$ according to (10). Here we use the straight line between (0, 0) and (1, 1) explicitly. The second, curved arm of the cross will be generated by starting at (1, 0) and (0, 1), in both cases using dx and dy, but without the minus signs. This again will make sure that the lines will run towards the saddle point. Some distance before the saddle point we stop and connect the ends with a short broken straight line through the saddle point.

Technique

Here is the MapleV program used for generating Figure 2.

```
>showreg := proc (lambda, wid:array, step:array, PLT)
>#  wid, step regulate integration algorithm rkf45 in G
>#  lambda=z[3] can be used to show the existence of a saddle point
>  local DX,DY,DfY,fcns,G,ii,kk,P,PS,spExist,sq3,SS,strL,sys,t,ttl,x,y,z;
>  G := array(1..2, 1..6): P := array(1..2, 1..6):
>  z := array(1..5, [0.6, 0.8, 1.0, 1.2, 1.4]):
>  fcns := {x(t), y(t)}:
>  _RELERR := Float(1,-2): _ABSERR :=Float(1,-3):
>  strL:=convert(lambda,string): sq3 := evalf(sqrt(1/3)):
>  PS := plot([0,0, 1,0, 1,1, 0,1, sq3,sq3], style=POINT): #spec. points
>  SS := {PS}:
>  DX := -sqrt(lambda)*(3*y(t)^2-1): DY := -(3*x(t)^2-1):
>  DfY := diff(y(t),t) = DY:
>  for ii from 1 to 5 do
>    sys:= diff(x(t),t)= z[ii]*DX, DfY:
>    G[1,ii] := dsolve({sys,x(0)=0,y(0)=0},fcns,numeric):
>    G[2,ii] := dsolve({sys,x(0)=1,y(0)=1},fcns,numeric):
>  od:
>  if abs(z[3]^2*lambda-1) < 0.001 then spExist:=1 else spExist:=0 fi:
>  if spExist=1 then
>    sys:= diff(x(t),t)=-DX, diff(y(t),t)=-DY:
>    G[1,6]:=dsolve({sys,x(0)=0,y(0)=1},fcns,numeric):
>    G[2,6]:=dsolve({sys,x(0)=1,y(0)=0},fcns,numeric):
>    P[1,3] := plot([0,0,1,1]):
>    PS:=plot([seq(subs(G[1,6](wid[6]*t),[x(t),y(t)]),t=step[1,6]-1),[sq3,sq3],
>             seq(subs(G[2,6](wid[6]*t),[x(t),y(t)]),t=step[2,6]-1)]):
>    SS:= SS union {PS,P[1,3]}:
>  fi:
>  for kk from 1 to 2 do
>    for ii from 1 to 5+spExist do
>      if (spExist=0) or (ii<>3) then # leave out P13, P23 if spExist=1
>        P[kk,ii]:=plot( map(subs, map(G[kk,ii], [seq(wid[ii]*t,
>                        t=0..step[kk,ii])]), [x(t),y(t)] )):
>        SS := SS union {P[kk,ii]}:
>      fi:
>    od:
>  od:
>  ttl:=`lambda=`.strL.`, z=1.4..0.6, from y=0 and y=1 inward`;
>  PLT:=plots[display](SS,view=[0..1,0..1],title=ttl):
>end:
```

Here is a sample call. To generate this plot (shown in Figure 2) may take 10 minutes on a 25 MHz 386 PC-class machine:

```
>showreg (10, array(1..6, [0.002, 0.003, 0.0025, 0.0025, 0.002, 0.0005]),
>  array(1..2,1..6,[[290,275,140,180,120,1],[200,320,60,70,65,1]]),`Plot1`):
>Plot1;
```

Here is another sample. This plot of a saddle point is not shown:

```
>showreg (1, array(1..6, [0.004, 0.004, 0.006, 0.004, 0.003, 0.005]),
>  array(1..2,1..6,[[290,320,140,320,320,220],[200,320,75,200,220,220]]),`Plot2`):
>Plot2;
```

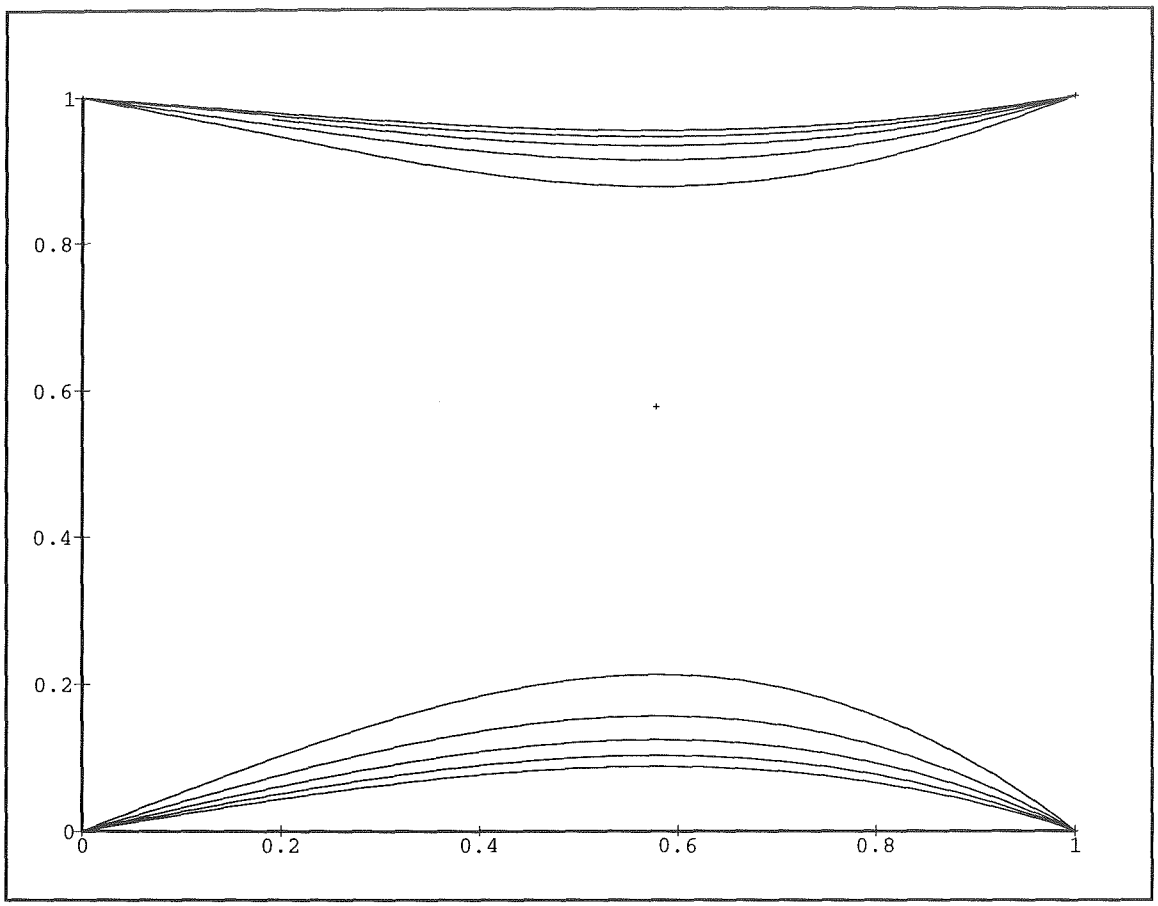


Figure 2: Lines $z_0 \in \{1.4,1.2,1.0,0.8,0.6\}$, λ=10, from y=0 and y=1 inward.

Acknowledgements

I thank Rob Corless and Michael Monagan for help via email to navigate through the complexities of MapleV, and an unnamed referee for various helpful suggestions.

References

[1] G. I. Barenblatt: *Dimensional Analysis*, Gordon and Breach Science Publishers, New York, (1987).

[2] T. Voegeli: *Dissertation 10457*, ETH Zurich, (1993).

Maple in Education - Part II

Tony Scott, Michael Monagan[1], Richard Pavelle[2] and Darren Redfern[3]

In the Fall of 1993, many Maple users attended the "Technology in Mathematics Teaching" (TMT) conference at the University of Birmingham and participated in Maple related presentations from Jenny Watson of CLECOM Ltd, and Simon Eveson of the University of York. This included interesting panels of teachers discussing the impact of the computer algebra system (CAS) on the undergraduate curriculum.

The first thing that was pointed out is how the CAS could totally change the approach by which a student would solve a particular problem.

An illustrative example is the problem of partial fraction decomposition. We meet this when teaching integration of rational functions. Splitting a rational function into partial fractions and then integrating each term of the partial fraction decomposition is a simple idea. The problem is the mechanics and algebra required to actually do a partial fraction decomposition. Even a relatively simple problem, such as this one

$$\frac{2x^4 - 4x^3 + 3x^2 + 1}{x^3 - 2x^2 + x} = 2x + x^{-1} + 2\,(x-1)^{-2}$$

would be a real challenge to most students to get right. Consequently partial fractions are often not covered very well. It is not possible to ask students to tackle a realistic problem. However such computations are easy for a CAS! This suggests that much of the algebraic manipulation "drudgery" can be deferred to the machine allowing teachers more time to spend on concepts. However, these possibilities are tempered if one is not familiar with the capabilities of the CAS. This creates a need for problem-solving sessions which make optimal use of the available resources within the system.

In issue no. 7 of this Newsletter [1], we presented an article devoted to the use of Maple as an educational tool for solving problems in Science. To further illustrate the capabilities of the Maple system as a teaching tool, we present another four such educational examples in Physics, Chemistry and Engineering. The solutions presented here are in the form of Maple worksheets and range in difficulty from senior high-school years to sophomore level problems.

The first example is a straightforward exercise which solves the linear system of equations generated by Kirchoff's laws for the well-known "Wheatstone Bridge" of RC circuit theory. This was an extremely useful circuit developed in 1843 by Charles Wheatstone and was widely used to determine values of unknown resistances. What Maple gives us here is formulae (for the solutions) in the parameters R_1 to R_5 which denote the resistances in the circuit.

The next two examples originate from D. McLaughin's article entitled "Symbolic Computation in Chemical Education" [2]. The second example is a straightforward problem in statistical mechanics, namely the derivation of the Stefan-Boltzmann Law from the Planck Radiation formula. This is handled by Maple's capabilities for solving definite integrals [3]. The third exercise consists of getting both the eigenvalues and eigenvectors for the electronic structure of molecules in organic (quantum) chemistry, according to the simple approximation known as Huckel Molecular Orbitals (HMO) theory. Getting the eigenvalues is a feasible exercise for many symbolic systems but getting the correct eigenvalues is a more difficult task. Nonetheless,

[1] Institute für Wissenschaftliches Rechnen, ETH-Zentrum, CH 8092 Zürich, Switzerland.

[2] MIT Lincoln Laboratory, Lexington, MA, USA 02173 and Computer Algebra Associates, 23 Berkshire Drive, Winchester, MA, USA 01890.

[3] Practical Approach, P.O. Box 1007, Stratford, Ontario, Canada N5A 6W4.

this particular case presents no difficulties thanks to the eigenvector implementation discussed in issue 8 of this Newsletter [4]. Thus, the first three sessions represent very accessible problems one can find in any undergraduate curriculum, all of which are readily handled by Maple.

The fourth example involves a somewhat more challenging problem, namely solving for the motion of a chain sliding off a table [6]. Specifically, the problem consists of a uniform chain of finite length L and linear density μ which lies in a heap right on the edge of a smooth table and then starts sliding over the edge. The question is: *what is the acceleration of the chain during the time it is sliding over the edge?* Note that the mass of the chain which is pulled down by the gravitational force increases as the chain is sliding, so the mass here is varying in time. This involves using the general form of Newton's first law and leads to a second-order non-linear differential equation which could be quite daunting unless the student makes the right assumption about the solution. However, Maple can help guide the student to the solution as shown in the worksheet.

Note that these educational sessions as well as those presented in issue no. 7 are available through the share library as discussed previously in the *News and Announcements* section.

Acknowledgements

Special thanks to Grant Keady for fruitful discussions and Jenny Watson for her invaluable assistance during the TMT conference.

References

[1] T.C. Scott, R. Pavelle and D. Redfern: Maple in Education, *Maple Technical Newsletter* issue 7, Birkhauser, (Spring 1992).

[2] D. McLaughin: Symbolic Computation in Chemical Education. *Applications of Computer Algebra*, Ed. by R. Pavelle, Kluwer Academic Publishers, Hingham, MA, (1985).

[3] T.C. Scott, Y.B. Band, and K.O. Geddes: Recipes for Solving Broad Classes of Definite Integrals and Applications, *Maple Technical Newsletter* issue 10, Birkhauser, (Fall 1993).

[4] M.B. Monagan: Exact Eigenvalue and Eigenvector Computation in Maple, *Maple Technical Newsletter* issue 8, Birkhauser, (Fall 1992).

[5] J.J. Brophy: *Basic Electronics for Scientists*, McGraw-Hill, 3rd ed., New-York, pp. 17-19, (1977).

[6] R.A. Becker: *Introduction to Theoretical Mechanics*, McGraw-Hill, New York, pp. 184-190, (1954).

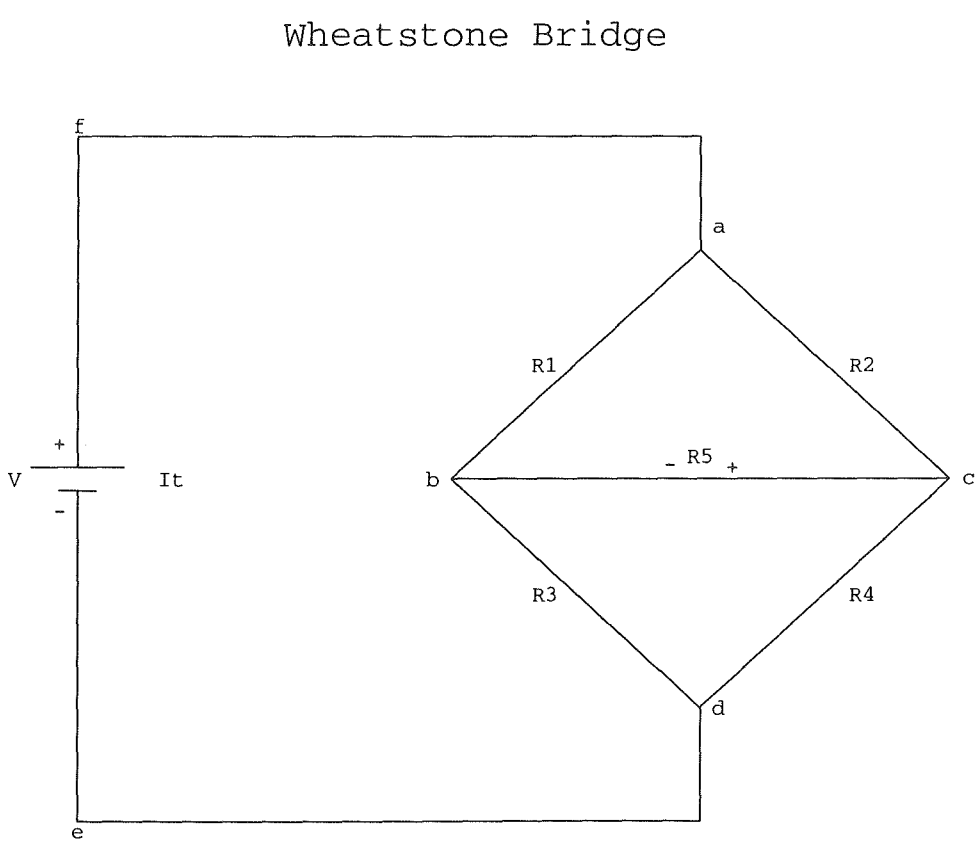
Worksheet 1: Wheatstone Bridge

Physics: RC Circuit Theory

by Tony Scott and Michael Monagan (see reference [5])

Consider the following circuit known as the Wheatstone bridge input as a Maple PLOT data structure.

```
>bridge := CURVES(
     [[0,2.9],[0,0],[10,0],[10,1],[6,3],[10,5],[10,6],[0,6],[0,3.1]],
     [[6,3],[14,3]],[[10,1],[14,3],[10,5]] ):
>IRtexts := TEXT( [7.5,2], 'R3' ), TEXT( [7.5,4], 'R1' ), TEXT( [12.6,4], 'R2' ),
     TEXT( [12.6,2], 'R4 ' ), TEXT( [10,3.2], 'R5' ):
>labels := TEXT([10.3, 1], d), TEXT([10.3,5.2], a), TEXT([5.7,3], b),
     TEXT([14.3,3], c), TEXT([0,6.1],f),TEXT([0,-0.1],e):
>battery := CURVES( [[-0.75,3.1], [0.75,3.1]], [[-0.3,2.9], [0.3,2.9]] ),
     TEXT( [1.5,3], 'It'), TEXT( [-0.3,3.3], '+'), TEXT( [-1,3], V),
     TEXT( [-0.3,2.7], '-'), TEXT([9.5,3.1],'-'), TEXT([10.5,3.1],'+'):
>circuit := PLOT( bridge, IRtexts, battery, labels, AXESSTYLE(NONE),
     TITLE( 'Wheatstone Bridge') ):
>circuit; # display the Wheatstone bridge circuit
```



Apply Kirchhoff's rule for conservation of current to branch points a,b, and d, where the currents I1 to I5 denote respectively the currents across the resistors R1 to R5:

```
>eqI1:=It-I1-I2=0;
```

$$eqI1 := It - I1 - I2 = 0$$

```
> eqI2:=I1-I3+I5=0;

                              eqI2 := I1 - I3 + I5=0

> eqI3:=I3+I4-It=0;

                              eqI3 := I3 + I4 - It=0
```

Apply Kirchhoff's rule for conservation of voltage (energy) to loops abdefa, acba and bcdb. Note carefully the indicated polarities of the various IR drops as they are encountered in each traversing loop.

```
> eqV1:=-I1*R1-I3*R3+V=0;

                            eqV1 := - I1R1 - I3R3 + V=0

> eqV2:=-I2*R2-I5*R5+I1*R1=0;

                           eqV2 := - I2R2 - I5R5 + I1R1=0

> eqV3:= I5*R5-I4*R4+I3*R3=0;

                           eqV3 := I5R5 - I4R4 + I3 R3=0
```

Solve for 6 equations in 6 unknown currents.

```
> sols := solve(eqI1,eqI2,eqI3,eqV1,eqV2,eqV3,I1,I2,I3,I4,I5,It):
```

Pick off the solution for I5

```
> I5 := subs( sols, I5 );

      I5 := V(R1R4 - R2R3) / (R5R4R3 + R5R2 R1 + R5R2R3 + R5R4 R1 + R2R4R3
                          + R2R1R3 + R1R4 R3 + R2R4R1)
```

I5 is directly proportional to V. Now find out what R4 must be to make the current I5 zero in terms of R1, R2, and R3

```
> solve( I5, R4 );
```

$$\frac{R2\ R3}{R1}$$

which is the "classic result". As pointed out on p.19 of Brophy's book: in the common version of a Wheatstone bridge, resistances R1 and R2 are connected to a switch to give decade values of the ratio R2/R1, and R3 is a continuously variable calibrated resistor. Once the bridge is adjusted such that the current I5 is zero, then the unknown resistance has to be R3 R2 / R1 as shown by Maple.

Worksheet 2: Planck Radiation Formula

Statistical Mechanics

by Tony Scott, Michael Monagan and Richard Pavelle (see reference [2])

T = temperature k = Boltzmann's constant h = Planck's constant

Telling Maple that these quantities are positive:

```
> assume(h>0): assume(k>0): assume(T>0):
```

Getting the Stefan-Boltzmann Law from the Planck Radiation Distribution function:

```
> rho := v -> 8*Pi*h*v^3/(exp(h*v/(k*T)) - 1);
```

$$\rho := v \rightarrow 8\frac{\pi h v^3}{e^{\frac{hv}{kT}} - 1}$$

```
> Energy:=Int(rho(nu),nu=0..infinity)= int(rho(nu),nu=0..infinity);
```

$$Energy := \int_0^{\infty} 8\frac{\pi h\sim \nu^3}{e^{\frac{h\sim \nu}{k\sim T\sim}} - 1} = \frac{8}{15}\frac{\pi^5 k\sim^4 T\sim^4}{h\sim^3}$$

This is the Stefan-Boltzmann Law (of Blackbody radiation). The energy is proportional to T^4 and the constant of proportionality is:

```
> sigma:=coeff(op(2,"),T,4);
```

$$\sigma := \frac{8}{15}\frac{\pi^5 k\sim^4}{h\sim^3}$$

This is known as Stefan's constant.

Worksheet 3: Huckel Molecular Orbitals (HMO)

Chemistry: Organic (Quantum) Chemistry

by Darren Redfern, Richard Pavelle and Tony Scott (see reference [2])

Huckel Molecular Theory provides a simple approximation to the electronic structure of molecules, i.e.:

1) The eigenvalues approximate the energy spectrum.

2) The eigenvectors approximate the states.

```
>with(linalg):
```

The HMO for Butadiene can be represented with the following matrix.

```
>matButadiene:=array([[a,b,0,0],[b,a,b,0],[0,b,a,b],[0,0,b,a]]);
```

$$matButadiene := \begin{bmatrix} a & b & 0 & 0 \\ b & a & b & 0 \\ 0 & b & a & b \\ 0 & 0 & b & a \end{bmatrix}$$

Calculate the eigenvalues, approximating Butadiene's energy spectrum.

```
>eigenvals(matButadiene);
```

$$a-\frac{1}{2}b+\frac{1}{2}\sqrt{5}\,b,\ a-\frac{1}{2}b-\frac{1}{2}\sqrt{5}\,b,\ a+\frac{1}{2}b+\frac{1}{2}\sqrt{5}\,b,\ a+\frac{1}{2}b-\frac{1}{2}\sqrt{5}\,b$$

Calculate the eigenvectors, approximating Butadiene's energy states.

```
>eigenvects(matButadiene, radical);
```

$$\left[a-\frac{1}{2}b+\frac{1}{2}\sqrt{5}\,b,\ 1,\ \left\{\left[1\ \ -\frac{\frac{1}{2}b-\frac{1}{2}\sqrt{5}\,b}{b}\ \ \frac{\frac{1}{2}b-\frac{1}{2}\sqrt{5}\,b}{b}\ \ -1\right]\right\}\right],$$

$$\left[a-\frac{1}{2}b-\frac{1}{2}\sqrt{5}\,b,\ 1,\ \left\{\left[1\ \ -\frac{\frac{1}{2}b+\frac{1}{2}\sqrt{5}\,b}{b}\ \ \frac{\frac{1}{2}b+\frac{1}{2}\sqrt{5}\,b}{b}\ \ -1\right]\right\}\right],$$

$$\left[a+\frac{1}{2}b+\frac{1}{2}\sqrt{5}\,b,\ 1,\ \left\{\left[1\ \ -\frac{-\frac{1}{2}b-\frac{1}{2}\sqrt{5}\,b}{b}\ \ -\frac{-\frac{1}{2}b-\frac{1}{2}\sqrt{5}\,b}{b}\ \ 1\right]\right\}\right],$$

$$\left[a+\frac{1}{2}b-\frac{1}{2}\sqrt{5}\,b,\ 1,\ \left\{\left[1\ \ -\frac{-\frac{1}{2}b+\frac{1}{2}\sqrt{5}\,b}{b}\ \ -\frac{-\frac{1}{2}b+\frac{1}{2}\sqrt{5}\,b}{b}\ \ 1\right]\right\}\right]$$

As you can see, there is one eigenvector corresponding to each eigenvalue. (The actual eigenvectors are the lists contained within the sets.) The eigenvectors are identical except for change(s) of sign(s).

Moving on to the next example, the HMO for Naphthalene is represented thus.

```
>matNaphthalene:=array([
>     [a,0,0,0,0,0,0,b,b,0],
>     [0,a,b,0,0,0,0,0,b,0],
>     [0,b,a,b,0,0,0,0,0,0],
>     [0,0,b,a,b,0,0,0,0,0],
>     [0,0,0,b,a,0,0,0,0,b],
>     [0,0,0,0,0,a,b,0,0,b],
>     [0,0,0,0,0,b,a,b,0,0],
>     [b,0,0,0,0,0,b,a,0,0],
>     [b,b,0,0,0,0,0,0,a,b],
>     [0,0,0,0,b,b,0,0,b,a]]);
```

$$matNaphthalene := \begin{bmatrix} a & 0 & 0 & 0 & 0 & 0 & 0 & b & b & 0 \\ 0 & a & b & 0 & 0 & 0 & 0 & 0 & b & 0 \\ 0 & b & a & b & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & b & a & b & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & b & a & 0 & 0 & 0 & 0 & b \\ 0 & 0 & 0 & 0 & 0 & a & b & 0 & 0 & b \\ 0 & 0 & 0 & 0 & 0 & b & a & b & 0 & 0 \\ b & 0 & 0 & 0 & 0 & 0 & b & a & 0 & 0 \\ b & b & 0 & 0 & 0 & 0 & 0 & 0 & a & b \\ 0 & 0 & 0 & 0 & b & b & 0 & 0 & b & a \end{bmatrix}$$

Calculate the the eigenvalues, approximating Naphthalene's energy spectrum.

```
>eigenvals(matNaphthalene);
```

$$a-b,\ a+b,\ a-\frac{1}{2}b+\frac{1}{2}\sqrt{5}\,b,\ a-\frac{1}{2}b-\frac{1}{2}\sqrt{5}\,b,\ a+\frac{1}{2}b+\frac{1}{2}\sqrt{13}\,b\ a+\frac{1}{2}b-\frac{1}{2}\sqrt{13}\,b,$$

$$a-\frac{1}{2}b+\frac{1}{2}\sqrt{13}\,b\ a-\frac{1}{2}b-\frac{1}{2}\sqrt{13}\,b,\ a+\frac{1}{2}b+\frac{1}{2}\sqrt{5}\,b\ a+\frac{1}{2}b-\frac{1}{2}\sqrt{5}\,b$$

A specific eigenvector lambda can be computed by solving A x = lambda x or equivalently computing the nullspace of the characteristic matrix A - lambda I . For example

```
>nullspace(charmat( matNaphthalene, (a+b) ));
```

$$\{\begin{bmatrix} 0 & 0 & -1 & -1 & 0 & 0 & -1 & -1 & 1 & 1 \end{bmatrix}\}$$

Worksheet 4: Chain Sliding off the Edge of a Table

Physics: Classical Mechanics

by Darren Redfern and Tony Scott (see reference [6])

This particular problem consists of a uniform chain of finite length (say L) and linear density (say r), which lies in a heap right on the edge of a smooth table and then starts sliding over the edge. The question is:

What is the acceleration "a" of the chain during the time it is sliding over the edge?

The mass of the chain, which is pulled down by the gravitational force, increases as the chain is sliding, so the mass in our problem is varying in time. Hence, we can't use the old-fashioned F = ma (force equals mass times acceleration) here. We must use a more general version of Newton's second law, namely:

$$F = \frac{\partial}{\partial t} p(t), \; p(t) = m(t)v(t)$$

which implies that,

$$F = \left(\frac{\partial}{\partial t} m(t)\right) v(t) + \left(\frac{\partial}{\partial t} v(t)\right) m(t)$$

We take y=0 as our reference for the table top, and since the chain starts from a rest position t=0 we have:

$$v(t=0)=0, \; y(t=0)=0$$

Thus, the mass m(t) is r*y(t) (where the y-axis points downwards). So

$$\frac{\partial}{\partial t} m(t) = r\left(\frac{\partial}{\partial t} y(t)\right)$$

which is also equal to:

$$rv(t)$$

As well, we can say that F is equal to:

$$m(t)g = ry(t)g$$

where "g" is the gravitational constant. Therefore our equation of motion for F is:

$$ry(t)g = rv(t)^2 + a(t)ry(t)$$

where v(t) equals

$$\frac{\partial}{\partial t} y(t)$$

and a(t) equals

$$\frac{\partial^2}{\partial t^2} y(t) = \frac{\partial}{\partial t} v(t)$$

which is the acceleration. Dividing the above by the density "r", we get the DE we have to solve.

$$y(t)\, g = \left(\frac{\partial}{\partial t} y(t)\right)^2 + y(t)\left(\frac{\partial^2}{\partial t^2} y(t)\right)$$

Thus we obtain a second-order NON-linear differential equation which can be quite daunting for students, however, with Maple, we can get the solution as shown below:

```
>readlib(isolate):
>eq := y(t)*g = diff(y(t), t)^2 + y(t)*diff(y(t), t2):
>ans := dsolve(", y(t));
```

$$ans := t = \int_0^{y(t)} \frac{\sqrt{3}y1}{\sqrt{2\,y1^3 g+3\_C1}}dy1 - \_C2,\ t = \int_0^{y(t)} -\frac{\sqrt{3}y2}{\sqrt{2\,y2^3 g+3\_C1}}dy2 - \_C2$$

So, there is a solution of the form t=y(t). Select the solution corresponding to positive time. Also, at t=0, v=0 and y=0: so clearly _C2=0. Thus, our solution is obtained from:

```
>res := t = int( sqrt(3)*y2/sqrt(2*y2^3*g+3*_C1), y2=0..y(t) );
```

$$res := t = \int_0^{y(t)} \frac{\sqrt{3}y2}{\sqrt{2\,y2^3 g+3\_C1}}dy2$$

We still have one arbitrary constant left. Let's isolate the expression for the velocity:

```
>diff(", t): subs(diff(y(t), t)=v, y(t)=y, ");
```

$$1 = \frac{\sqrt{3}yv}{\sqrt{2y^3 g + 3\_C1}}$$

Isolate the velocity itself and see what happens for small "y":

```
>assign(isolate(", v)):
>v;
```

$$\frac{1}{3}\frac{\sqrt{2y^3 g + 3\_C1}\sqrt{3}}{y}$$

```
>series(v, y);
```

$$\sqrt{\_C1}y^{-1} + \frac{1}{3}\frac{g}{\sqrt{\_C1}}y^2 + O(y^5)$$

This suggests that _C1 must be zero; but since 1/_C1 also appears above, the case _C1=0 is a special case:

```
>subs(_C1=0, res);
```

$$t = \int_0^{y(t)} \frac{1}{2}\frac{\sqrt{2}\sqrt{3}}{\sqrt{y2}\sqrt{g}}dy2$$

Here is our solution:

```
>isolate(", y(t));
```

$$y(t) = \frac{1}{6}t^2 g$$

Here is the velocity:

```
>diff(", t);
```

$$\frac{\partial}{\partial t}y(t) = \frac{1}{3}tg$$

Here is the acceleration:

```
>diff(", t);
```

$$\frac{\partial^2}{\partial t^2} y(t) = \frac{1}{3} g$$

which is a constant in time, and which also is exactly 1/3 of "g". Since we had v(t=0) = y(t=0) = 0 here, this is the physical solution. Armed with hindsight, one could then see that the DE resembled the form: => $v^2 = vo^2 + 2*a*y$ (a constant acceleration problem), where vo=0. However, intuition often fails when you're dealing with a non-linear DE. Nonetheless, as we have seen, Maple can readily give us the correct solution.

A Separable Differential Equation: New Insights

Robert J. Lopez[1]

The separable ordinary differential equation considered in this paper was part of a routine assignment extracted from a traditional calculus text during a second semester calculus class based on Maple. Each student had a workstation running Maple available during class and on exams, thereby allowing the implementation of a curriculum that stretched the bounds of the traditional course. While making up the student grader's key for this problem it became clear that using a powerful tool like Maple for such a routine problem was inappropriate. The manipulative practice embedded in exercises of this type must yield to more imaginative explorations of a higher pedagogic value. This paper is an exposition of how this one problem can be extended, via Maple, to become a vehicle of greater mathematical insight.

The differential equation

```
> q := diff(y(t),t) = t^2/(1+3*y(t)^2);
```

$$q := \frac{\partial}{\partial t} y(t) = \frac{t^2}{1 + 3y(t)^2}$$

can be solved mentally by separation of variables and two simple integrations. In Maple, this approach is implemented with

```
> int(1+3*y^2,y)=int(t^2,t)+c;
```

$$y + y^3 = \frac{1}{3}t^3 + c$$

whereas using Maple's built-in solver requires

```
> q1 := dsolve(q,y(t));
```

$$q1 := y(t) + y(t)^3 - \frac{1}{3}t^3 = \_C1$$

In either event the solution is given implicitly as shown in q1. Typically, the exercise ends here. A little algebra to separate variables, and two integrations. Hardly the kind of intellectual activity to create a lasting attachment to mathematics.

But suppose we ask "What does the solution $y(t)$ actually look like?" Does Maple allow us to gain some insight into the behavior of $y(t)$? There are several approaches we might take to visualize the solutions to this differential equation. Some of these require exposure to concepts normally taught in the third semester calculus and we leave it to the reader to contemplate a calculus sequence where the appropriate tools are made available sooner than they typically are.

A first approach might be a completely numeric solution to the differential equation. If we load Maple's DEtools package:

```
> with(DEtools);
```

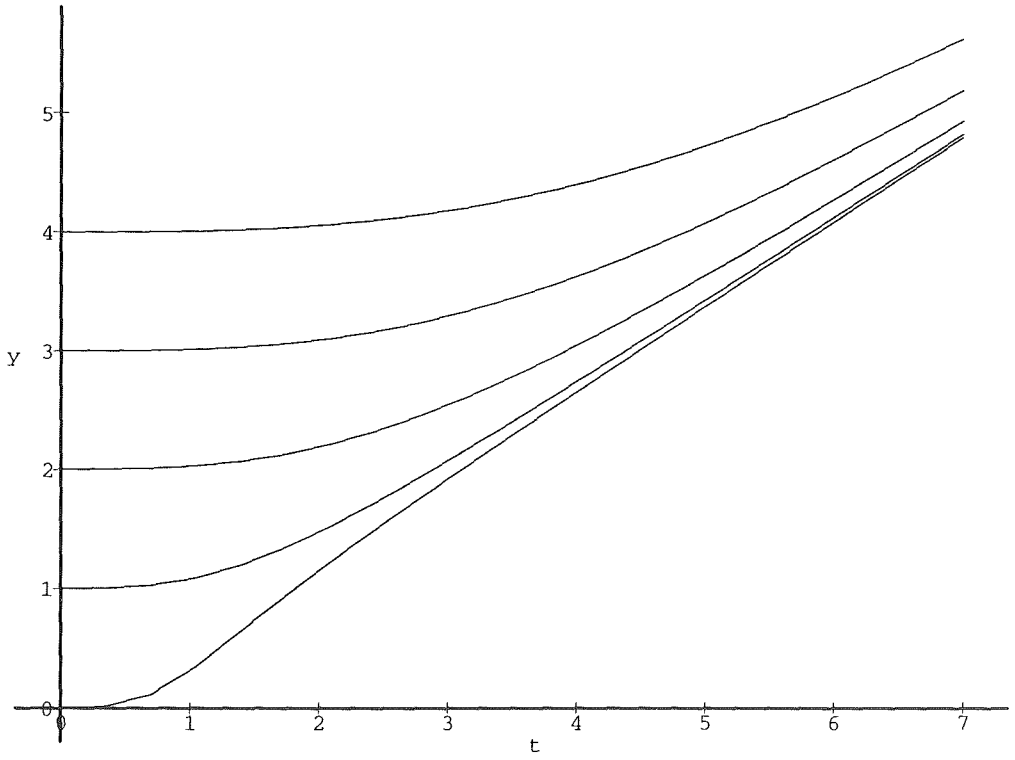
[*DEplot, DEplot1, DEplot2, Dchangevar, PDEplot, dfieldplot, phaseportrait*]

[1] Department of Mathematics Rose-Hulman Institute of Technology Terre Haute, IN 47803, r.lopez@rose-hulman.edu

we gain access to the DEplot command with which we can produce simultaneously a graph of several initial value problems for the given differential equation:

```
> DEplot(q,[t,y],0..7,{[0,0],[0,1],[0,2],[0,3],[0,4]},title='Figure 1');
```

Figure 1



The resulting graph suggests that each solution might be tending to an oblique asymptote, and in class it is possible to get the students to articulate this hypothesis. Moreover, they can even be induced to recall that verification of this guess will require showing that the limiting slope of $y(t)$ becomes constant. The time spent extracting these insights from the students is more rewarding for the instructor than an equal amount of time spent drilling in the manipulations for solving separable differential equations!

Before tackling the task of verifying the oblique asymptote we present the reader with some alternatives for obtaining the above figure. First, we can look at the relationship in q1 as an implicit definition of $y(t)$ and use Maple's implicitplot command to extract an equivalent graph. Begin by loading Maple's plots package to access both the implicitplot command and the display command with which we will superimpose the individual implicit plots created.

```
> with(plots):
```

The values of the constant _C1 corresponding to the initial conditions chosen in the DEplot command are 0, 2, 10, 30, and 68. If we put these values into a list c:

```
> c := [0,2,10,30,68];
```

$$c := [0, 2, 10, 30, 68]$$

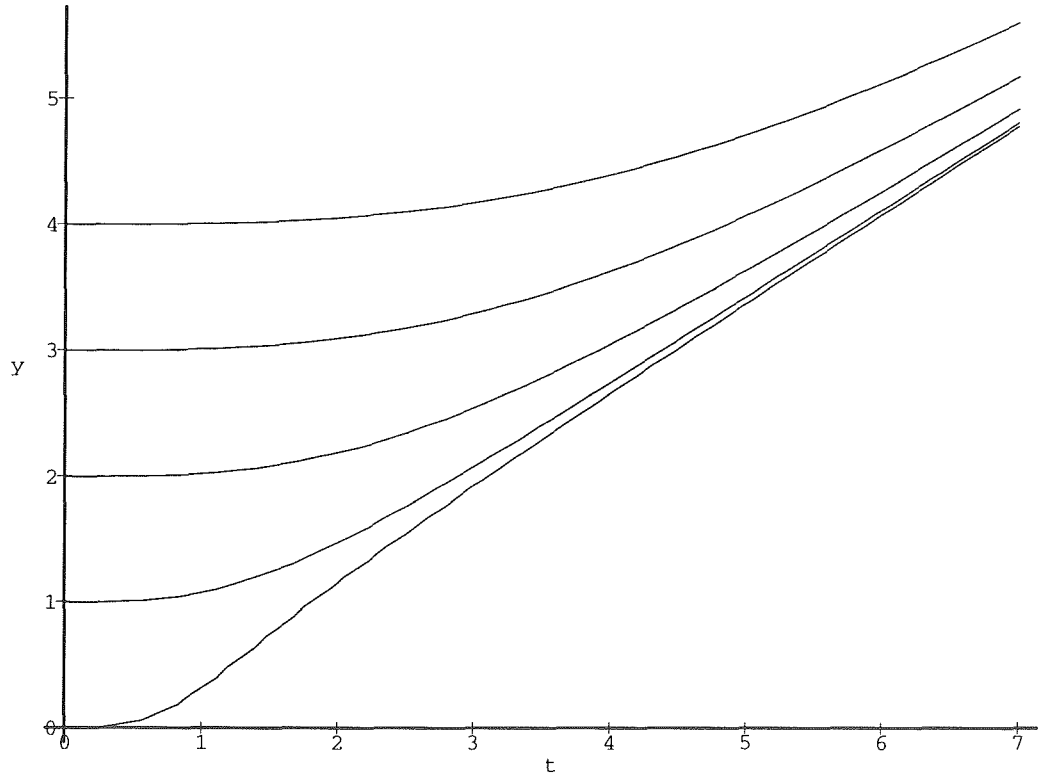
we can use a Maple loop to create five implicit plots.

```
> for k from 1 to 5 do
  f.k := implicitplot(subs(_C1=c[k],q1),t=0..7,y=0..6);
  od:
```

The colon on the "od" is essential for suppressing printing of the plot data structures generated, and the individual graphs are shown on one set of axes by:

```
> display([f.(1..5)],title='Figure 2.');
```

Figure 2



We leave the reader to experiment with an approach via contours. If we think of the implicit function in q1 as $z(t, y)$ defined by the left hand side of q1, then the contours of this surface are graphs of solutions such as seen in Figures 1 and 2. Thus, one can interactively manipulate a 3d plot to obtain contours on the surface $z(t, y)$ and rotate the surface so a contour map results. Alternatively, one can use Maple's contourplot command directly.

We consider now the challenge of obtaining an explicit representation for $y(t)$ and using it to verify the existence of the oblique asymptote. Thus,

```
> q2 := solve(q1,y(t));
```

$$q2 := \%1^{1/3} - \frac{1}{3}\frac{1}{\%1^{1/3}}, -\frac{1}{2}\%1^{1/3} + \frac{1}{6}\frac{1}{\%1^{1/3}} + \frac{1}{2}I\sqrt{3}\left(\%1^{1/3} + \frac{1}{3}\frac{1}{\%1^{1/3}}\right),$$

$$-\frac{1}{2}\%1^{1/3} + \frac{1}{6}\frac{1}{\%1^{1/3}} - \frac{1}{2}I\sqrt{3}\left(\%1^{1/3} + \frac{1}{3}\frac{1}{\%1^{1/3}}\right)$$

$$\%1 := \frac{1}{6}t^3 + \frac{1}{2}\_C1 + \frac{1}{18}\sqrt{4 + 3t^6 + 18t^3\_C1 + 27\_C1^2}\,\sqrt{3}$$

There are three solutions, separated by commas, and compressed by the label %1. The second and third solutions contain I = sqrt(-1) so that perhaps just the first solution is real. Let's grab this first solution via:

```
> Y := q2[1];
```

$$Y := \left(\frac{1}{6}t^3 + \frac{1}{2}\_C1 + \frac{1}{18}\sqrt{4 + 3t^6 + 18t^3\_C1 + 27\_C1^2}\sqrt{3}\right)^{1/3}$$

$$-\frac{1}{3}\frac{1}{\left(\frac{1}{6}t^3 + \frac{1}{2}\_C1 + \frac{1}{18}\sqrt{4 + 3t^6 + 18t^3\_C1 + 27\_C1^2}\sqrt{3}\right)^{1/3}}$$

Note the use of upper case Y as the name for the real solution. This avoids confusion if parts of this worksheet are to be re-evaluated later on. Also note how Maple dropped the abbreviation when it "sensed" that space was available.

This is the solution $y(t)$ given explicitly. (We could also construct Figure 1 from Y, which was how students did this exploration prior to the availability of Release 2.) There is an oblique asymptote if $y'(t)$ becomes constant for large t.

```
> L := limit(diff(Y,t),t=infinity);
```

$$L := \frac{1}{18}18^{2/3}6^{1/3}$$

Well, L appears to be constant but in need of simplification:

```
> simplify(L);
```

$$\frac{1}{3}3^{2/3}$$

We have verified the observation that the solutions to the given separable differential equation tend to an oblique asymptote. While beyond the scope of the traditional course in integral calculus, this same asymptotic behavior can also be seen from:

```
> asympt(Y,t);
```

$$\frac{1}{3}3^{2/3}t - \frac{1}{3}\frac{3^{1/3}}{t} + \frac{1}{3}\frac{3^{2/3}\_C1}{t^2} + \frac{1}{3}\frac{3^{1/3}\_C1}{t^4} + O\left(\frac{1}{t^5}\right)$$

Clearly, the linear behavior dominates as t becomes large, a result again supported by analytical evidence. Even more interesting, however, is the question: can the asymptotic behavior of $y(t)$ be predicted solely from the differential equation itself? We leave that challenge to the reader, along with an additional stipulation: if the answer to the question is "yes," can it be explained within the framework of second semester calculus?

Thermodynamics with Maple V: II Phase Equilibria in Binary Systems

Ross Taylor[1]

Vapor-liquid equilibrium data for two component or binary systems has been compiled by Gmehling and Onken [1]. One of the data sets for benzene - toluene at a pressure of 101325 Pa is summarized below:

```
> btdata:=array([[0,0,383.75],[0.1,0.2080,379.35],[0.2,0.3720,375.35],
    [0.3,0.5070,371.75],[0.4,0.6190,368.45],[0.5,0.7130,365.45],
    [0.6,0.7910,362.55],[0.7,0.8570,359.95],[0.8,0.9120,357.35],
    [0.9,0.9590,355.35],[1.0,1.0,353.25]]):
  print (btdata);
```

$$\begin{bmatrix} 0 & 0 & 383.75 \\ .1 & .2080 & 379.35 \\ .2 & .3720 & 375.35 \\ .3 & .5070 & 371.75 \\ .4 & .6190 & 368.45 \\ .5 & .7130 & 365.45 \\ .6 & .7910 & 362.55 \\ .7 & .8570 & 359.95 \\ .8 & .9120 & 357.35 \\ .9 & .9590 & 355.35 \\ 1.0 & 1.0 & 353.25 \end{bmatrix}$$

where column 1 of the above matrix is the mole fraction of benzene in the liquid, column 2 is the (measured) mole fraction of benzene in the vapor that is in equilibrium with the liquid and column 3 is the measured temperature in kelvin. Note that the mole fraction of benzene in the vapor is always higher than the mole fraction of benzene in the liquid.

The prediction of equilibrium between two phases is an important application of thermodynamics. This article shows how Maple can be used to perform some simple phase equilibrium calculations. In addition, we demonstrate how Maple can be used to create phase diagrams and interpret the data given above.

Theoretical Background

The mole fractions of the i-th component in two different phases in equilibrium with each other are related by the following expression:

```
> EQM[i]:=y[i]=K[i]*x[i]: EQM[i];
```

$$y_i = K_i x_i$$

[1] Department of Chemical Engineering, Clarkson University, Potsdam, New York 13699-5705

where the K_i are the K-values or equilibrium ratios. They are also referred to sometimes as K-constants although they are probably the least constant property in all of science and engineering. In fact, K-values are, in general, complicated functions of temperature, pressure and the composition of both phases.

It is beyond the scope of this article to discuss in detail the wide variety of models used by thermodynamicists for calculating K-values. We shall reserve such a discussion for a later article. We shall use some specific models for the K-values later when the time comes to carry out some actual calculations.

In view of the way they are defined, the mole fractions of both phases sum to 1.

```
> Sumx:=sum(x[i],i=1..c)=1: Sumy:=sum(y[i],i=1..c)=1: Sumx,Sumy;
```

$$\sum_{i=1}^{c} x_i = 1, \sum_{i=1}^{c} y_i = 1$$

where c is the number of components in the mixture.

There are 2c+2 variables in this set of equations: c mole fractions in the x-phase, c mole fractions in the y-phase, the system temperature, and the system pressure. There are only c+2 equations, however: c equilibrium equations and 2 mole fraction summation equations. The number of variables that must be specified before we can carry out any calculations is the difference between these two numbers, c.

Simple phase equilibrium calculations are based on finding the roots of an objective function derived by subtracting the mole fraction summation equations.

```
> Eqn:=Sumy-Sumx: Eqn;
```

$$\left(\sum_{i=1}^{c} y_i\right) - \left(\sum_{i=1}^{c} x_i\right) = 0$$

In what follows we will assume that the x-phase is a liquid and the y-phase is a vapor.

Bubble Point Calculations

The *bubble point* temperature is the temperature at which the first bubble of vapor forms in a liquid as it is heated. To determine the bubble point of a liquid mixture we specify c-1 mole fractions of the x-phase and the pressure (or temperature). The mole fraction of component number c may be calculated from the summation equation for the x-phase. The remaining equations may then be solved for the mole fractions of the y-phase and for the system temperature (or pressure). We eliminate the vapor phase mole fractions from the objective function above, noting that the mole fractions of the x-phase sum to 1, to obtain:

```
> BPeqn:=subs({EQM[i],Sumx},Eqn): BPeqn;
```

$$\left(\sum_{i=1}^{c} K_i x_i\right) - 1 = 0$$

We will now make a function of this expression for later use.

```
> fBP:=proc(K,x,components)
  local i;
  if type(components,'..') then
    sum(K[i]*x[i],i=components)-1;
  else
    convert([seq(K[i]*x[i],i=components)],'+')-1;
  fi;
  end:
```

Example 1 - Bubble Point of an Ideal System

Estimate the bubble point of a mixture containing 20 mole percent benzene and 80 mole percent toluene. Assume that the benzene - toluene system may be considered to behave as an ideal system; one where the K-values are functions only of temperature and pressure:

The K-values of benzene and toluene will be calculated using Raoult's law with the vapor pressures computed from the Antoine equation [2].

```
> Antoine := (T, A, B, C) -> exp(A - B / (T + C));
```

$$Antoine := (T, A, B, C) \rightarrow e^{\left(A - \frac{B}{T+C}\right)}$$

The Antoine coefficients for benzene and toluene are (recalculated from Gmehling and Onken, 1977)

```
> AntA[benzene]:=21.06807; AntB[benzene]:=2948.78; AntC[benzene]:=-44.563;
```

$$\mathrm{AntA}_{\mathrm{benzene}} := 21.06807$$

$$\mathrm{AntB}_{\mathrm{benzene}} := 2948.78$$

$$\mathrm{AntC}_{\mathrm{benzene}} := -44.563$$

```
> AntA[toluene]:=21.15917; AntB[toluene]:=3242.38; AntC[toluene]:=-47.181;
```

$$\mathrm{AntA}_{\mathrm{toluene}} := 21.15917$$

$$\mathrm{AntB}_{\mathrm{toluene}} := 3242.38$$

$$\mathrm{AntC}_{\mathrm{toluene}} := -47.181$$

which give the vapor pressures in pascals if the temperature is in kelvin.

The K-values may be expessed as follows:

```
> component:=[benzene,toluene]:
  for i in component do K[i] := Antoine(T,AntA[i],AntB[i],AntC[i]) / P: od;
```

$$K_{\mathrm{benzene}} := \frac{e^{\left(21.06807 - 2948.78\frac{1}{T-44.563}\right)}}{P}$$

$$K_{\mathrm{toluene}} := \frac{e^{\left(21.15917 - 3242.38\frac{1}{T-47.181}\right)}}{P}$$

We make functions of these expressions for help in later calculations.

```
> for i in component do Kvalue[i]:=unapply(K[i],T,P); od:
```

We continue by making a function from the bubble-point equation, incorporating the above expressions for the K-values:

```
> x:='x': BP2:=unapply(fBP(K,x,component),T,P,x);
```

$$BP2 := (T, P, x) \rightarrow \frac{e^{\left(21.06807 - 2948.78\frac{1}{T-44.563}\right)} x_{\mathrm{benzene}}}{P} + \frac{e^{\left(21.15917 - 3242.38\frac{1}{T-47.181}\right)} x_{\mathrm{toluene}}}{P} - 1$$

where the arguments to the Maple function are temperature (in kelvin), pressure (in pascals), and the mole fractions of both components. Note that we have retained the dependence on the mole fractions and pressure as well as temperature so that we can use the same function to find the bubble point temperature under other conditions. We will need to do this in a later example.

Maple's `fsolve` command is unable to find a zero to this function unless we tell it roughly where to look. To help us locate the zero we plot the function:

```
> x[component[1]]:=0.2: x[component[2]]:=1-x[component[1]]:
  plot(BP2(T,101325,x),T=300..400);
>
```

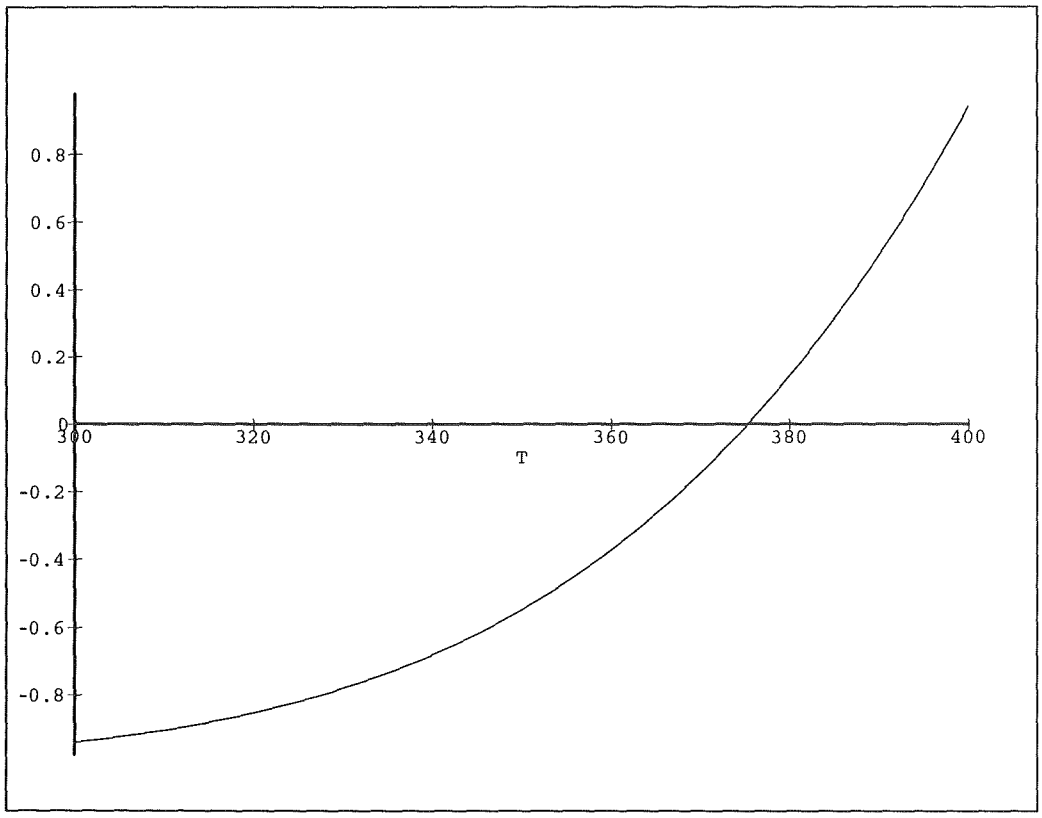


Figure 1

from which we see that there is a zero somewhere between 370 and 400 K. Now we can invoke the `fsolve` command and specify the range in which we know a solution exists.

```
> TBP:=fsolve(BP2(T,101325,x),T,350..400): T[bubble]=TBP;
```

$$T_{\text{bubble}} = 375.3696892$$

Thus, the estimated bubble point of this mixture is 375.37 K. This result compares quite favorably with the measured value of 375.35 K given in the table. The mole fractions of benzene and toluene in the vapor in equilibrium with the liquid at this temperature are:

```
> y:='y': for i in component do y[i]=Kvalue[i](TBP,101325)*x[i]; od;
```

$$y_{\text{benzene}} = .3747998330$$

$$y_{\text{toluene}} = .6252001673$$

The experimental value for the mole fraction of benzene from the table of data above is 0.3720.

Dew Point Calculations

The *dew point* of a vapor mixture is the point at which the first drop of liquid forms as it is cooled. The calculation of the dew point requires the specification of the mole fractions of the vapor (y) phase and the system pressure (or temperature).

```
> i:='i': DPeqn:=subs({x[i]=solve(EQM[i],x[i]),Sumy},Eqn): DPeqn;
```

$$1 - \left(\sum_{i=1}^{c} \frac{y_i}{K_i} \right) = 0$$

We will now make a procedure of this expression for later use.

```
> fDP:=proc(K,y,components)
  local i;
  if type(components,'..') then
    1-sum(y[i]/K[i],i=components);
  else
    1-convert([seq(y[i]/K[i],i=components)],'+');
  fi;
  end:
```

Example 2 - Dew Point Temperature Calculation

Estimate the dew point temperature of a mixture of benzene (61.9 mole percent) and toluene (38.1 mole percent) at a pressure of 101325 Pa.

We first make a function of the dew point equation using the Raoult's law K-values we used in the prior example.

```
> DP2:=unapply(fDP(K,y,component),T,P,y);
```

$$DP2 := (T, P, y) \rightarrow 1 - \frac{y_{\text{benzene}} P}{e^{\left(21.06807 - 2948.78 \frac{1}{T - 44.563}\right)}} - \frac{y_{\text{toluene}} P}{e^{\left(21.15917 - 3242.38 \frac{1}{T - 47.181}\right)}}$$

As before, we plot this function over a range of temperatures to see approximately where the solution lies (see Figure 2).

```
> y[component[1]]:=0.619: y[component[2]]:=1-y[component[1]]:
  plot(DP2(T,101325,y),T=350..450);
```

Now we can ask Maple to compute a more precise numerical approximation to the dew point temperature.

```
> TDP:=fsolve(DP2(T,101325,y),T,350..450): T[dew]=TDP;
```

$$T_{\text{dew}} = 368.5407080$$

The computed estimate of 368.54 K compares well with the measured value of 368.45 K as reported in the table of data in the Introduction. The mole fractions in the liquid in equilibrium with the vapor at this temperature are:

```
> x:='x': for i in component do x[i]=y[i]/Kvalue[i](TDP,101325); od;
```

$$x_{\text{benzene}} = .3985858047$$

$$x_{\text{toluene}} = .6014141969$$

The experimental value of the mole fraction of benzene from the table above is 0.4.

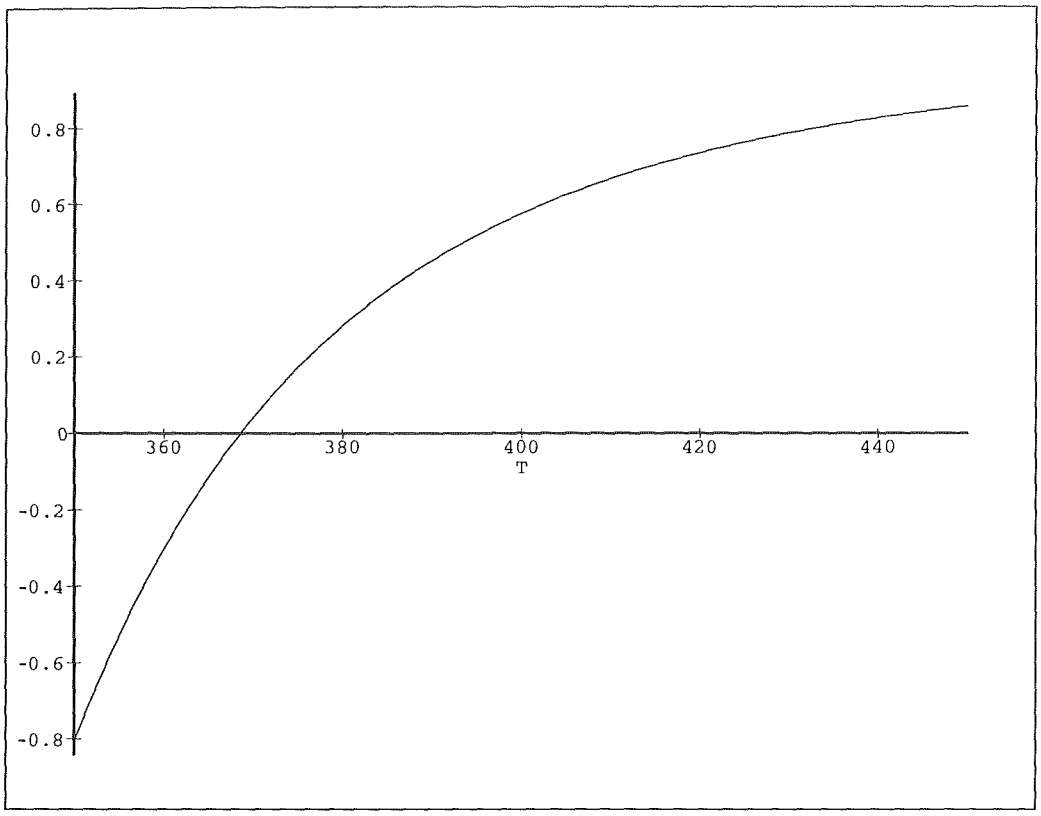


Figure 2

Example 3 - Dew Point Pressure

The bubble and dew point equations can also be solved for pressure if the temperature is specified. In this example we determine the pressure at which a mixture starts to condense at a fixed temperature.

```
> y:='y': fDP(K,y,component)=0;
```

$$1-\frac{y_{\text{benzene}}P}{e^{\left(21.06807-2948.78\frac{1}{T-44.563}\right)}}-\frac{y_{\text{toluene}}P}{e^{\left(21.15917-3242.38\frac{1}{T-47.181}\right)}}=0$$

This expression can be solved explicitly for pressure:

```
> Peqn:=P=solve(-",P): Peqn;
```

$$P=\frac{1}{\frac{y_{\text{benzene}}}{e^{\left(21.06807000-2948.780000\frac{1}{T-44.56300000}\right)}}+\frac{y_{\text{toluene}}}{e^{\left(21.15917000-3242.380000\frac{1}{T-47.18100000}\right)}}}$$

We make a function of this result:

```
> fP:=unapply(rhs(Peqn),T,y):
```

and evaluate the function at a temperature of 375 K for an equimolar mixture of benzene and toluene.

```
> y[component[1]]:=0.5: y[component[2]]:=1-y[component[1]]:
  P[dew]=fP(375,y);
```

$$P_{\text{dew}}=110562.9261$$

Thus, an equimolar mixture of these two components at 375 K will start to condense when the pressure reaches 110563 Pa.

Phase Diagrams

Phase diagrams are a useful method of displaying vapor liquid equilibrium data graphically. We shall compute the VLE curves for the benzene-toluene system that we have used in the above examples.

Computing the vapor - liquid equilibrium curves involves carrying out a series of bubble point or dew point calculations for mole fractions of one component from 0 to 1 (making sure that the mole fractions of both components sum to 1). We have chosen to use bubble-point calculations in the following examples. They are easier to perform than are dew point calculations (although there is no essential difference in the degree of difficulty when Raoult's law is used to calculate the K-values). These calculations are carried out in the following examples.

Example 4 - Phase Diagrams for Benzene - Toluene

In the following sequence of Maple commands we carry out a series of bubble point calculations covering the range of liquid mole fractions of benzene from 0 to 1. The bubble point temperature is computed first and the vapor composition follows from the equilibrium equation. The results are stored in an array for later processing.

```
> numpts:=31: pts:=array(1..numpts,1..3):
  for k from 1 to numpts do
    x[component[1]]:= evalf((k-1)/(numpts-1)): x[component[2]]:=1-x[component[1]]:
    TBP:=fsolve(BP2(T,101325,x),T,300..410):
    y[component[1]]:= Kvalue[component[1]](TBP,101325)*x[component[1]]:
    pts[k,1]:=x[component[1]]: pts[k,2]:=y[component[1]]: pts[k,3]:=TBP:
  od:
```

In order to display the measured and calculated equilibrium data we have created the procedure `Txyplot`. This procedure makes use of the function `makepoints` which takes as input an array (or matrix) and returns a set of points that can be displayed with the `plot` comand.

```
> makepoints := proc(A,j1,j2,i1,nrows)
  # Procedure to make a list of points for plotting purposes
  # A       = matrix or array of data
  # j1, j2  = Column indeces of 1st and 2nd columns extracted from A
  #           to make the x and y coordinates of the points.
  #           (optional, but j1 must be specified if j2 is given).
  # i1      = Index of first row of A for points extracted from A
  #           (optional but j1 and j2 must be given first).
  # nrows   = Number of rows of matrix A to be extracted
  #           (optional, and can only be given as the fifth argument).

    local col1, col2, row1, nrowsA, i, lastrow, lrow;
    col1 := 1; col2 := 2; row1 := 1; lastrow:=linalg[rowdim](A);
    if nargs > 1 then col1 := j1;
      if nargs > 2 then col2 := j2;
        if nargs > 3 then row1 := i1;
          if nargs > 4 then lrow := row1 + nrows - 1;
            if lrow > lastrow then lrow := lastrow; else lastrow := lrow;
    fi; fi; fi; fi; fi;
    [seq([A[i,col1],A[i,col2]], i = row1..lastrow)];
  end:
```

The procedure `Txyplot` is designed to take binary VLE data (both computed and experimental) that is stored in a Maple array (or matrix) with three columns: liquid mole fraction of component 1, vapor mole fraction of component 1 and temperature. This procedure is easily modified for systems with more than two components.

```
> Txyplot := proc(calcdata, expdata)
    local bubline, dewline, bubdata, dewdata, Txyplot1, Txyplot2;
    bubline := makepoints(calcdata,1,3): dewline := makepoints(calcdata,2,3):
    Txyplot1 := plot({bubline,dewline},0..1,axes=boxed,labels=['x, y','T']):
    if nargs > 1 then
      bubdata := makepoints(expdata,1,3): dewdata := makepoints(expdata,2,3):
      Txyplot2 := plot({bubdata,dewdata},0..1,style=point):
    fi;
    if nargs = 1 then Txyplot1; else plots[display]({Txyplot1,Txyplot2}); fi;
  end:
```

It is also useful to plot the vapor composition against the liquid composition on an x-y diagram. This is done with the procedure `yxplot` which is similar to `Txyplot`.

```
> yxplot := proc(calcdata, expdata)
    local yxline, diagline, yxdata, xyplot1, xyplot2;
    yxline := makepoints(pts,1,2): diagline := makepoints(pts,1,1):
    xyplot1 := plot({yxline,diagline},0..1,0..1,labels=['x','y'],axes=boxed):
    if nargs > 1 then
      yxdata := makepoints(expdata,1,2):
      xyplot2 := plot(yxdata,0..1,0..1,style=point):
    fi;
    if nargs = 1 then xyplot1; else plots[display]({xyplot1,xyplot2}); fi;
  end:
> Txyplot(pts,btdata);
```

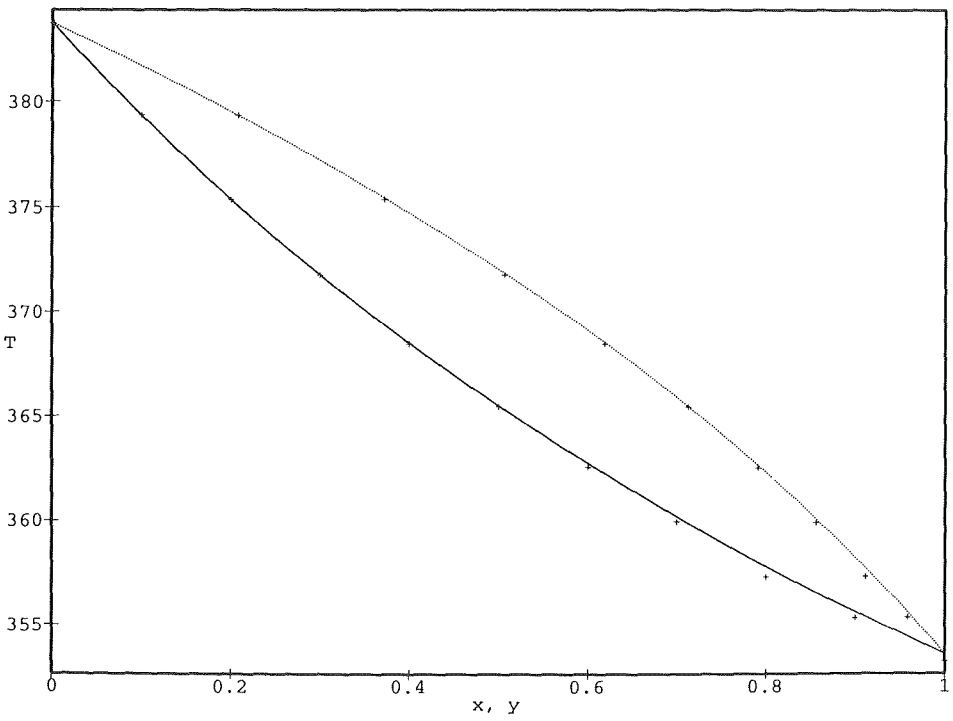


Figure 3

Figure 3 shows the calculated bubble-point line (the lower curve) and the calculated dew-point line (upper curve) as a function of the mole fraction of benzene (x-axis). The experimental data are shown as isolated points. You can see that the data are in good agreement with the calculated curves.

```
> yxplot(pts,btdata);
```

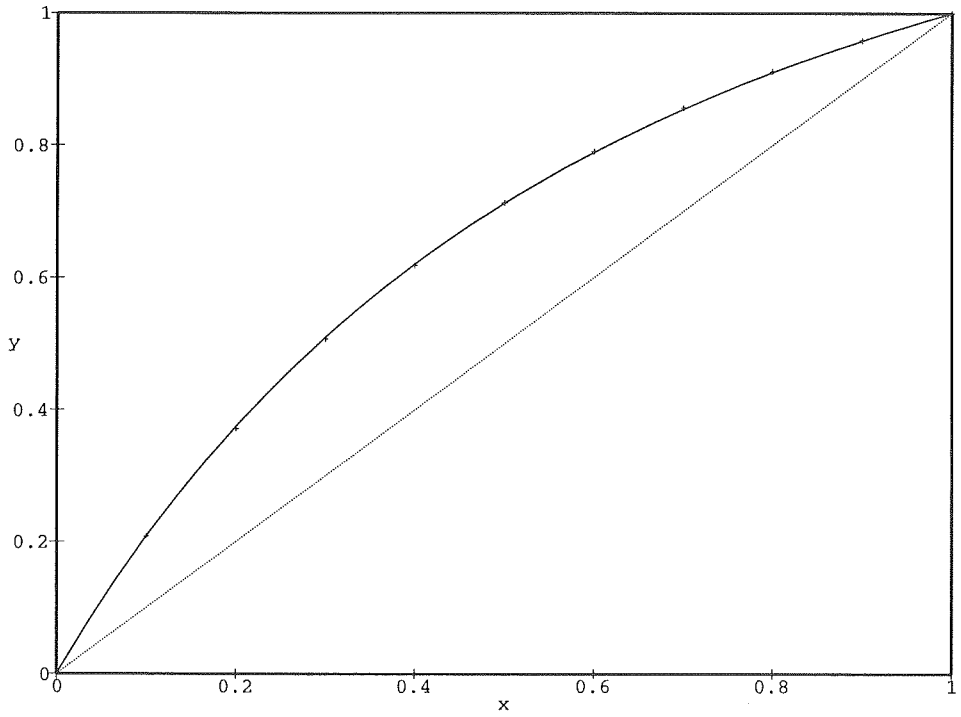


Figure 4

Note that the measured data points lie almost exactly on top of the predicted equilibrium line. The close agreement between the measured data and the calculated curves is an indication that the benzene - toluene system is indeed nearly ideal and that Raoult's law may safely be used to estimate the equilibrium ratios of this system.

Exercises

1. Repeat each of the first three examples for some of the other data points in the table of data in the introduction.
2. What is the bubble point temperature of the mixture in Example 1 at a pressure of 50000 Pa? At 200000 Pa?
3. In this article we have solved three different kinds of phase equilibrium problem: Bubble point temperature, dew point temperature, and dew point pressure. Use Maple to compute bubble point pressures.
4. Create a set of equilibrium data for the benzene - toluene system at constant temperature. Write a procedure `Pxyplot` to display *P-x-y* diagrams where pressure (rather than temperature) is plotted as a function of composition.

For further reading (as well as a source of many other problems) we recommend the book by Walas [3].

Postscript

In this article we have tried to show how Maple can be used to solve some simple phase equilibrium problems. We have, of course, only barely scratched the surface of what can be done in this area but lack of space prevents us from including additional examples.

A Maple worksheet version of this article `phase.ms` is available in the Maple share library. Raoult's law only works for a few simple systems such as the benzene-toluene mixture used in the preceding examples. For nonideal systems like, for example, 2-propanol -water, the K-values depend on composition and we must use different models to evaluate them. An additional worksheet called `phase2.ms` includes similar examples for a nonideal binary system.

Acknowledgements

The author would like to thank Alastair Rough for his assistance in the preparation of this article and Mike Monagan for advice on Maple coding.

References

[1] J. Gmehling, and U. Onken: *Vapor-Liquid Equilibrium Data Collection*, DECHEMA Chemistry Data Series, DECHEMA, (1977ff).

[2] R.C. Reid, J. Prausnitz, and B. Poling: *Properties of Gases and Liquids*, 4th Ed., McGraw-Hill, New York, (1987).

[3] S.M. Walas: *Phase Equilibria in Chemical Engineering*, Butterworth, (1985).

Guidelines for Authors

Any user of Maple who has implemented a useful, elegant, or illustrative solution to a particular problem is invited to submit a description of his or her solution to the *Maple Technical Newsletter*. We especially encourage authors whose articles have technical merit and are accessible and self-explanatory to the general user of Maple. Three classes of articles are published in the *Maple Technical Newsletter*:

(i) Articles on Maple functionality.

(ii) Education-level articles.

(iii) Articles on Research/Applications.

Applications in *all* areas are of interest to the newsletter but the intent (and category) of the article must be made very clear by the articles within their articles. The articles of category (i) are usually provided by the developers of the Maple system. Articles of category (ii) must be as accessible as possible. All articles should be roughly 5 to 10 pages long, as they would appear in double-spaced, single column format on 8.5 x 11 inch paper. It is preferable that the article be submitted in either the form of LaTeX code including hard copy output or a Maple worksheet file including hardcopy output. It is important that the results obtained by the Maple system be *reproducible* to the general user. In particular for articles of class (i), it is recommended that results obtained by the Maple system should be written in the form of an interactive session. To this end, the author need only submit Maple instruction sets in *verbatim* mode, such as, for example:

```
> a := int( exp( -x^2 ), x = 0..infinity );
> evalf(a);
> ...
```

and whereby the generated output is converted into LaTeX code. Both Maple *input* and *output* must be shown. To further ensure reproducibility of results, the author must use a standard version of Maple and not his or her own customized version. References should be included.

Guidelines and tools for preparing articles for the Maple Technical Newsletter, as well as LaTeX style files and sample articles can be obtained via anonymous ftp from the MTN directory of the share library. With submission of an article, a short one-paragraph biography from the author(s) including fields of interest and a test file containing the Maple input code contained within the submitted article, should be included. Exceptions to these guidelines must be approved by the editor.

Our intention is to provide the user with useful detailed "how-to" information of a computational nature. Any inquiries or submissions should be directed to the editor Tony Scott at:

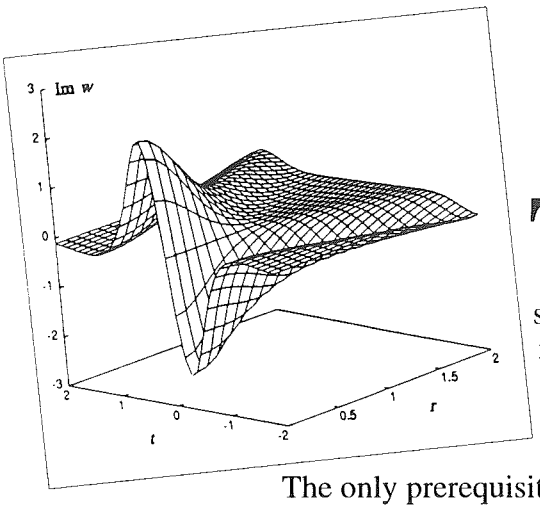
Mathematical Institute
University of Oxford
24-29 St-Giles'
Oxford OX1-3LB
tel: +44 (0)865 (2)73556
e-mail: tcscott@maths.oxford.ac.uk
or scott@cfa.harvard.edu

Acknowledgements

Tony Scott would like to thank Prof. Ian Grant FRS of the Mathematical Institute at Oxford for allowing him the resources and time to contribute to the present issue of the *Maple Technical Newsletter.* Special thanks to Michael Monagan for his additional contributions to the *Maple Technical Newsletter.*

Additional thanks to: Bruce Char, Dominik Gruntz, Bruno Salvy, Marc Rybowicz, Greg Fee, Alastair Rough, Ross Taylor, Robert Lopez, Pearl Dresser, Kate Atherley, David Clark, Anu Pathria, Kelly Roach, Clyde Martin, Kathleen White, Viswanath Ramakrishna, Jérôme Lang, Jenny Watson, Tim Jennings, and Tony Clewett for their assistance; to the authors, to the members of the editorial board who make the newsletter possible; and to everyone at Waterloo Maple Software and Birkhäuser Publishing who helped in the preparation of this newsletter.

A FRIENDLY GUIDE TO WAVELETS



Gerald Kaiser
University of Massachusetts at Lowell, MA

This volume is designed as a textbook for an introductory course on wavelet analysis and time-frequency analysis aimed at graduate students or advanced undergraduates in science and engineering. It can also be used as a self-study or reference book by practicing researchers in signal analysis and related areas. Since the expected audience is not presumed to have a high level of mathematical background, much of the needed analytical machinery is developed from the beginning. The only prerequisites for the first eight chapters are matrix theory, Fourier series, and Fourier integral transforms. Each of these chapters ends with a set of straightforward exercises designed to drive home the concepts just covered, and the many graphics should further facilitate absorption. Chapters 10 and 11 consist of original research and are written in a more advanced style. In Chapter 10 it is shown that the structure of Maxwell's equations implies the existence of a wavelet analysis specifically adopted to electromagnetic radiation. The associated "electromagnetic wavelets" are pulses parameterized by their point of origin and their scale, and can be made arbitrarily short by choosing fine scales. Furthermore, it is shown that every electromagnetic wave can be composed of such localized wavelets. This is applied in Chapter 11 to give a new formulation of radar based on electromagnetic wavelets. Since this theory is fully relativistic, its description of the Doppler effect is exact. In particular, it is three-dimensional, and does not make the usual assumption that the outgoing signal has a narrow bandwidth. Thus it should be useful in the construction of ultra-wideband radar systems.

CONTENTS:

August 1994 Approx. 250 pp. Hardcover $34.50 ISBN 0-8176-3711-7

ISSN 1061-5733
Birkhäuser
Boston Basel Berlin