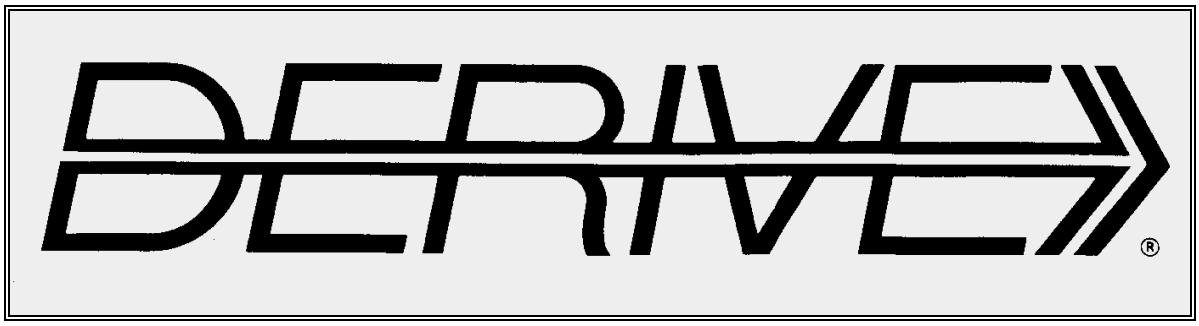
THE DERIVE - NEWSLETTER #70

ISSN 1990-7079

THE BULLETIN OF THE



USER GROUP

+ CAS-TI

Contents:

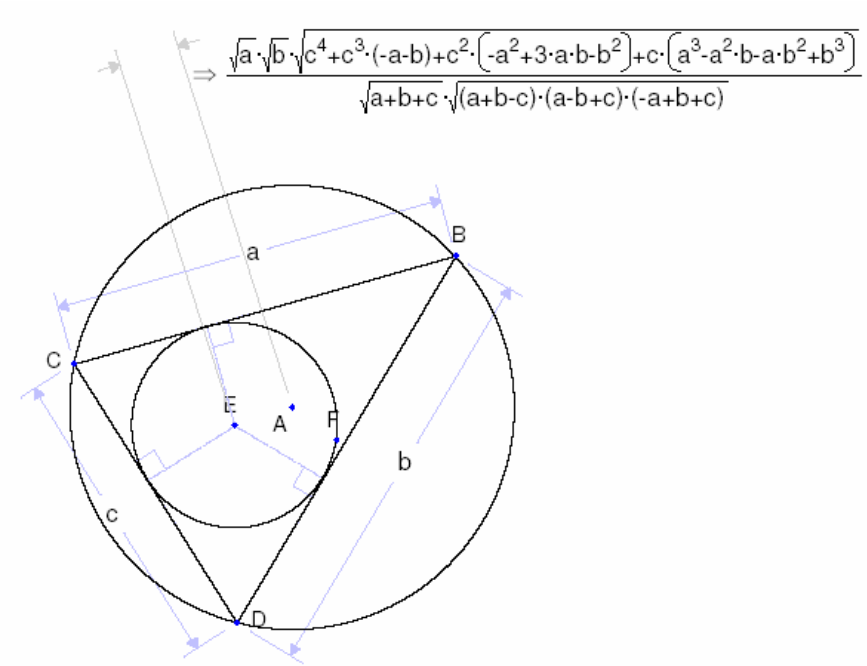
June 2008

D-N-L#70	**Information**	D-N-L#70

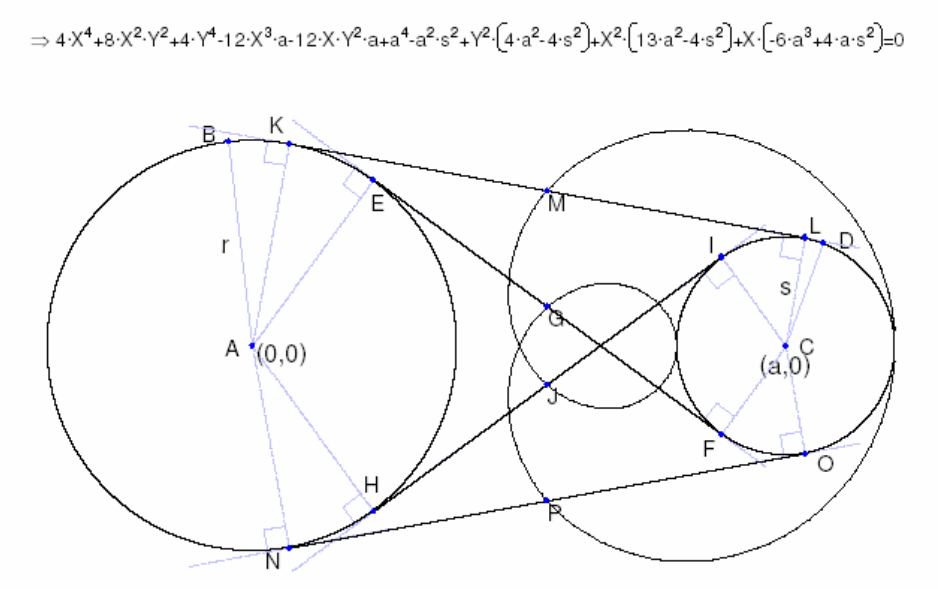
I attended the USACAS 2008 Conference in Chicago-Northfield which was held 28/29 June. Among many interesting lectures was one very special presentation. Philip Todd from Tigard, Oregon showed his Symbolic Geometry program ***Geometry Expressions***.

This is an exciting piece of software which combines dynamical geometry program with symbolic calculations in a very impressive way.

Here are two screen shots. The first one shows the general formula of the distance of incenter and circumcenter of a triangle given by its three sides a, b and c.



The second example shows the implicit form of the locus of centers of common tangents to two circles.



Find more information and download the manual (pdf) and two collections of worked examples (pdf) at

www.geometryexpressions.com

Dear DUG Members,

Let me start with an excuse for being too late with DNL#70. Due to some urgent family obligations and my participation at USACAS 2008 in Chicago I was unable to finish this issue by end of June.

At the other hand I am glad ot offer an issue with 48 pages content. When starting composing this DNL I intended to have a smaller issue but as you can see it is again full of – hopefully – interesting stuff.

At USACAS 2008 I attended several interesting lectures. Among others I met Philip Todd and Steve Arnold. Philip demonstrated a fascinating piece of software – *Geometry Expressions*. I´like to invite you to visit his website (see Information page). Steve had a well attended excellent session on the didactical use of TI-NSpire. Have a look on his website, too and download lots of NSpire materials (page 3).

Steve is one of the keynote speakers at TIME 2008. All keynote speakers and the titles of their lectures are given in the box below.

You can find the abstracts of all accepted submissions for lectures and workshops for the DERIVE & TI-strand and a list of all accepted lectures for the ACDCA-strand for TIME 2008.

We have three main contributions in this DNL: Peter Schofield deals with periodic decimal numbers. His article reminded me how I performed calculations when I was a pupil. Benno Grabinger demonstrates – once more – how mathematics can be found everwhere around us. This time is a toothpaste tube in the focus of his interest. Finally Johann Wiesenbauer offers his Titbits 35, which are in his opinion one of his best Titbits. Among very sophisticated routines connected with the RSA-algorithm he offers a function which enables the output of calculation time for any program and/or function. (We had this in one of the earlier DNLs. In DNL#55 Albert Rich answered a respective request from our Swiss friend René Hugelshofer).

I wonder if anybody can treat the simultaneous equation given on page 37 in a satisfying way with DERIVE.

With my best wishes for a fine summer
I remain

It can happen that DNL#71 will be late, too. We will attend TIME 2008 (22 – 26 Sept.) and then have a 3 weeks roundtrip in South Africa.

TIME 2008 – The Keynotes

David Arnold : *Meaningful Algebra with CAS*

Bernhard Kutzler : *Technology and the Yin and Yang of Mathematics Education*

Nurit Zehavi : *Didactical practices of computer algebra in mathematics education*

David Jeffrey: *Debugging Computer Algebra; Debugging Mathematics. A Two Way Street*

Download all *DNL-DERIVE*- and TI-files from
http://www.austromath.at/dug/

The *DERIVE-NEWSLETTER* is the Bulletin of the *DERIVE* & CAS-*TI User Group*. It is published at least four times a year with a contents of 40 pages minimum. The goals of the *DNL* are to enable the exchange of experiences made with *DERIVE*, *TI*-CAS and other CAS as well to create a group to discuss the possibilities of new methodical and didactical manners in teaching mathematics.

Editor: Mag. Josef Böhm
D´Lust 1, A-3042 Würmla
Austria
Phone/FAX: 43-(0)2275/8207
e-mail: nojo.boehm@pgv.at

Contributions:
Please send all contributions to the Editor. Non-English speakers are encouraged to write their contributions in English to reinforce the international touch of the *DNL*. It must be said, though, that non-English articles will be warmly welcomed nonetheless. Your contributions will be edited but not assessed. By submitting articles the author gives his consent for reprinting it in the *DNL*. The more contributions you will send, the more lively and richer in contents the *DERIVE* & CAS-*TI Newsletter* will be.

Next issue: September 2008
Deadline 15 August 2008

Preview: Contributions waiting to be published

Some simulations of Random Experiments, J. Böhm, AUT, Lorenz Kopp, GER
Wonderful World of Pedal Curves, J. Böhm
Tools for 3D-Problems, P. Lüke-Rosendahl, GER
Financial Mathematics 4, M. R. Phillips
Hill-Encription, J. Böhm
Farey Sequences on the *TI,* M. Lesmes-Acosta, COL
Simulating a Graphing Calculator in *DERIVE*, J. Böhm
Henon & Co, J. Böhm
Do you know this? Cabri & CAS on PC and Handheld, W. Wegscheider, AUT
An Interesting Problem with a Triangle, Steiner Point, P. Lüke-Rosendahl, GER
Overcoming Branch & Bound by Simulation, J. Böhm, AUT
Diophantine Polynomials, D. E. McDougall, Canada
Graphics World, Currency Change, P. Charland, CAN
Cubics, Quartics – interesting features, T. Koller & J. Böhm
Logos of Companies as an Inspiration for Math Teaching
Exciting Surfaces in the FAZ / Pierre Charland´s Graphics Gallery
BooleanPlots.mth, P. Schofield, UK
Old traditional examples for a CAS – what´s new? J. Böhm, AUT
Truth Tables on the TI, M. R. Phillips
Advanced Regression Routines for the TIs, M. R. Phillips
Where oh Where is IT? (GPS with CAS), C. & P. Leinbach, USA
Embroidery Patterns, H. Ludwig, GER
Mandelbrot and Newton with *DERIVE*, Roman Hašek, CZ
Snail-shells, Piotr Trebisz, GER
A Conics-Explorer, J. Böhm, AUT
Exercise Long Division with DERIVE
Practise Working with times

and others

Impressum:
Medieninhaber: *DERIVE* User Group, A-3042 Würmla, D´Lust 1, AUSTRIA
Richtung: Fachzeitschrift
Herausgeber: Mag.Josef Böhm

William Pickles, Petersfield, UK will@willpwr.demon.co.uk

Josef
I just dug out my HP95LX running DERIVE* and can't seem to get the display running with small characters -- is there a mail list where I might be able to get some help getting this right ?
Regards
William
* I put this combination together myself about 10 years ago.

Sorry, but I don´t have any idea. I´ll put your mail into our User Forum. Hopefully one of our members will have some advice for you, Josef.

Wolfgang Pröpper, Nürnberg, Germany

Dear Josef,
finally I found some time to read DNL#69. I found a mistake: Hubert´s TI-Nspire files `wurf.tns` and `wurf2.tns` (page 4) are missing in mth69.zip.
Regards
Wolfgang

Another sorry. I apologize and include both mentioned files into mth70.zip. Thanks for the note, Josef.

Reanimating a TI-89

Renate Wronski, a colleague from Styria wrote that her TI-89 didn´t show any sign of life after having a break of some months because of her sabbatical. As she didn´t receive any answer from TI I offered to send her one of my TI-89s. But then she wrote back:

Renate Wronski, Graz, Styria

I tried to reach the TI-service by phone and had success. The following procedure brought my TI-89 to life again:

> Remove one of the four batteries. Keep the APPS-key pressed while setting in this battery again. The device will 'wake up'. Then remove one battery again and and set it in without pressing any key. You should see now the black bar on the screen which is the sign for loading the OS. The you can recognize the well known TI-89 desktop.

This is a way to reset your calculator. I hope that there will be no more troubles in the future.

Best regards and many thanks for your offer,
Renate Wronski

For our TI-Nspire Users:

At the occasion of USACAS 2008 Conference in Chicago-Northfield I had the occasion to meet ***Stephen Arnold*** from Australia who is one of the TIME 2008 keynote speakers. Steve has a great website with a bundle of TI-NspireCAS papers:

http://compasstech.com.au.

Piterr, Poland playmakerpit@poczta.onet.pl

Hello
how calculate this differential equation:

$$4y'' \cdot \sqrt{y} - 1 = 0.$$

Thanks, Piterr

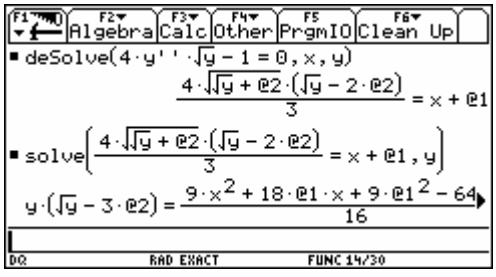
Dear Piterr,
This is the result of the DERIVE attempt - according to the Online Help
(see under Second order Ordinary Differential Equations how to interpret the general solution):

> DERIVE Online Help:
>
> **AUTONOMOUS_CONSERVATIVE(q, x, y, x0, y0, v0)** simplifies to an implicit algebraic solution of an equation of the form y''=q(y) having initial conditions y=y0 and y'=v0 at x=x0. Equations of this form are autonomous because the variable x does not occur and conservative because y' does not occur. If instead of the initial condition y'=v0 you have a second boundary condition y=y2 at x=x2, substitute x2 for x and y2 for y in the solution and solve for v0, which you can then eliminate in favor of x2 and y2.

$$\#1: \quad \text{AUTONOMOUS\_CONSERVATIVE}\left(\frac{1}{4\cdot\sqrt{y}}\right)$$

$$\#2: \quad x = \frac{4\cdot(2\cdot v0^2 - 3\cdot\sqrt{y0})\cdot|v0|\cdot \text{IF}(v0 = 0,\ 1,\ \text{SIGN}(v0))}{3} + \frac{4\cdot\sqrt{(\sqrt{y} + v0^2 - \sqrt{y0})}\cdot(\sqrt{y} - 2\cdot(v0^2 - \sqrt{y0}))\cdot \text{IF}(v0 = 0,\ 1,\ \text{SIGN}(v0))}{3} + x0$$

The TI-Voyage 200 has a built-in DE-Solver, which returns the general solution together with two constants @1 and @2.



Answer from Piterr:

Thanks
Function AUTONOMOUS_CONSERVATIVE(q, x, y, x0, y0, v0) must have begin condition. My example don't have begin condition.

Hm… Could I use this function ? I review Derive help and I can't match function for this example.

Abstracts of the ACCEPTED proposals for TIME 2008 – DERIVE & TI-CAS - Conference

1 Revisiting surprising results with CAS calculators

(Lecture 25 min)

Gilles Picard and Michel Beaudin, Ecole de technologie superieure, Montreal, Canada

We teach a variety of math topics (review of College Algebra, Calculus, Differential Equations, applied probability and statistics) in a Technical Engineering School. The Voyage 200 (or TI 89 Titanium) is mandatory for all new full-time students. We make use of this calculator on a regular basis, for exploring with students, in the classroom, all the classical curriculum in mathematics.

In July 2006, in Dresden, using the Voyage 200 handheld calculator, equipped with the operating system OS 3.10, we showed some surprising results given by the device. We showed examples where the CAS system gave unsatisfactory or strange results or where it couldn´t perform some commands. These problems were often related to the way the CAS system would simplify (or not) some expressions in some intermediate step of the calculations. We made some suggestions in order to fix the encountered bugs. We are happy to see that in Nspire CAS, OS 1.3, some of these bugs have been fixed and, in this talk, we will use Nspire CAS (both handheld and software) to show it and to see which bugs still remain. After revisiting these old examples, we will show some new problems encountered with our students while using the Voyage 200 CAS engine. These surprising results are still not resolved as of Version 1.3.2437 (2008-01-08) of the Nspire CAS calculator.

2 Functions, Programs and Libraries with TI-Nspire

(Workshop 90 min)

Josef Boehm, ACDCA, A 3042 Wuermla, Austria

TI-NspireCAS version 1.3 offers a very comfortable program editor. In this workshop we will focus on the difference between functions and programs by introductory examples. The second issue will be demonstrating how to create and to work with libraries which can be used very similar to utility packages in other computer algebra systems. TI-Nspire units will be provided.

3 Exploring Zeros of Complex Functions Graphically

(Lecture 25 min)

Giora Mann and Nurit Zehavi, Weizmann Institute of Science, Israel

Implicit plotting – one of the tools available in CAS – makes the exploration of zeros of any complex function a very simple procedure, as we shall demonstrate. Historically, the constraint on solving complex equations was that the graph of a complex function is 4D. Our idea was that both loci can be plotted implicitly, and the zeros are the intersection points of the two curves. However, not all the points that look like intersection points are actually intersection points. As a measure of control we introduce contour maps of $|f(z)|^2$ to determine whether a 'suspected' intersection point is indeed an intersection point, which is a zero of the function.

Students who are familiar with analysis of 2-variable functions can use their knowledge for finding the minimum points of $|f(z)|^2$ as the intersection points of $Re(|f(z)|^2)$ and $Im(|f(z)|^2)$. There are two drawbacks in this approach: (a) the students need to understand partial derivatives and (b) zero partial derivatives is only a necessary condition; it means that we get sometimes as many saddle points as minimum points.
The idea of zero imaginary and real parts of a function at a certain point, making this point a zero of the function, is a basic idea of complex analysis, which makes it independent of 2-variable analysis. This procedure is general and can be integrated into an introductory course of complex analysis, requiring only basic knowledge of 2-variable real functions and complex numbers.

4 Real-life applications of ODEs for undergraduates

(Lecture 25 min)

YuHe Yuan, Steve Joubert, Ying Gai., Dept. of Mathematics and Statistics, Tshwane University of Technology (TUT), Pretoria, South Africa

This study introduces real-life mathematical theories and models of international relationships suitable for undergraduate ordinary differential equations, by investigating conflicts between different nations or alliances. Based on the work of Richardson, systems of differential equations are constructed. The solutions and the stability of systems of ODEs are observed, with the aid of mathematical softwares such as Derive, Mathematica and Scientific Workplace. One of the most interesting tasks is to analyse the coefficients in the constructed models. In our opinion, the model first constructed by Richardson is an excellent application of ODEs (ordinary differential equations) and is useful for practice for learning ODEs. One would expect this kind of model to be added to the material in textbooks as a typical example.

5 Can CAS be trusted?

(Lecture 25 min)

Stephan V. Joubert and Temple H. Fay, Department of Mathematics and Statistics; TUT, Pretoria, South Africa

Most computer algebra systems (CAS) have built-in ordinary differential equation (ODE) solvers, but the accuracy of the solutions produced is not always obvious. Various ways of estimating the accuracy of ODE solvers are discussed here, extending work presented at the "Remarkable Delta 2003" conference in Queenstown, New Zealand and the "TIME 2004 Conference" in Montreal, Canada. Our methods are easy enough for undergraduates to implement because the needed mathematics is accessible to them. Many students (and their teachers) have an in-depth knowledge of how to check the accuracy of numerical routines, but many trust them blindly. On the other hand, testing the accuracy of a routine takes more time than just running the routine to produce a solution and this is another reason for taking a solution at face value. Such blind trust could have negative connotations if carried through to industry and elsewhere after the student graduates. We cite an example of how experienced mathematical scientists (academics) have fallen into the trap of assuming numerical solutions to be correct. There already exist a number of routines to test the accuracy of ODE solvers, some of them time intensive, and some not. The routines introduced here add to this collection of routines and one of them substantially reduces the calculation time of an existing routine. We extend our results for initial value problems to boundary value problems in ODE.

6 Data Acquisition and mathematical modelling – A case study

(Workshop 90 min)

Anna C.M. Bekker, Stephan V. Joubert and Temple H. Fay, TUT, Pretoria, South Africa.

A mathematical model is derived of a motor-vehicle tyre tread surface striking a speed bump on a stretch of otherwise smooth horizontal road. A mathematical idea is outlined to derive the model and a simple experiment, involving a typical vinyl record player needle, is described to measure the coefficient of damping of the tangential vibration of the rubber tyre.

7 A Discreet Compartmental Model for Lead Metabolism in the Human Body

(Lecture 25 min)

Charlotta E Coetzee, Stephan V Joubert and Frederika E Steyn, TUT, Pretoria, South Africa

A real-life example of a mathematical technique, employing a so-called transfer matrix, is developed for the compartmental analysis of lead metabolism in the human body. The technique is demonstrated with the aid of Bert's four-compartment biokinetic model. The results produced by this time-discrete approach correspond almost perfectly with those of a continuous-time method. The powerful calculation tools of the Computer Algebra System (CAS) "Scientific Workplace" are employed to illustrate the results using tables and graphs.

8 Expanding Student Perspectives: A Workshop on Forensic Applications of Mathematics

(Workshop 90 min)

Patricia Leinbach and Carl Leinbach, Adams County Pennsylvania Coroners Office and Gettysburg College, PA, USA

In this workshop you will be a member of a crime scene investigative team conducting a forensic investigation of the scene. Your job will not be to solve the crime, but to determine the manner and cause of death of an individual found at the scene. You will gather and analyze "evidence" gathered at the scene. (In fact, because this is a workshop on using such investigations in the classroom, you may be asked to create some "evidence" related to the description you are given.) The workshop will conclude with your team presenting their results and analysis to their colleagues at the workshop.

9 Using Learning Objects with TI-Nspire CAS

(Workshop 90 min)

Wade Ellis, Jr., West Valley College, San Jose, CA, 95130, USA

In this workshop, participants will work with a variety of learning objects developed for TI-Nspire CAS. They will also examine and discuss activities (lessons) that incorporate these learning objects. A learning object is a TI-Nspire document that allows students to act on a mathematical object, observe the consequences of their actions, and then reflect on the mathematical meaning of those consequences. These learning objects are intended to be use as investigative tools by students to generate and enhance mathematical understanding. These objects provide a platform for the development of investigative activities as well as problem-solving activities.

10 A Trinomial Factoring Investigation with Pre-service teachers

(Lecture 50 min)

Michael Meagher, Michael Todd Edwards, Asli Ozgun-Koca, Brooklyn College - CUNY, Miami University, Wayne State University, USA

Pre service teachers tend to accept the secondary mathematics curriculum as a static set and tend not to question the inclusion or exclusion of certain topics. Factoring of trinomials is a standard topic in school mathematics curricula worldwide and much class time is devoted to this topic. Building on work from Usiskin we present an activity, using the CAS and spreadsheet capabilities of the TI-Nspire, which engages students in establishing what percentage of trinomials with integer co-efficients are factorable. The task, which involves algebra, probability and statistics was then used as a vehicle to discuss with students the inclusion of specific topics in the school curriculum and how such decisions are made. The presentation will engage participants in the activity itself and will report the results of the students´ discussion of the curriculum.

11 Separatrices

(Lecture 25 min)

Phindele M. Skhosana, Temple H. Fay and Stephan V. Joubert, TUT, Pretoria, South Africa

In this presentation we examine two by two first order systems of ordinary differential equations and show how to determine phase plane portraits and identify separatrices when there is a saddle point. In order to do so we describe how to use a computer algebra system to generate trajectories from contour plots when possible and from numerical investigations. In many cases we can determine the equation of the separatrix. Generating a phase plane portrait is useful, for at a glance one can observe what initial values give rise to bounded solutions, periodic solutions and other important features. It also permits the instructor to concentrate on the qualitative aspects of the model under investigation rather than the calculational difficulties associated with finding solutions.

12 Modelling Cha Cha dance in using the "function"-tools within the Cabri 2 Plus or the TI NSpire environment

(Lecture 50 min)

Jean-Jacques Dahan, IREM of Toulouse, France

We will show how to model Cha Cha dance in programming the movement of two points (modelling the two feet of the dancer). We will use the special features of Cabri 2 plus and TI Nspire to lead these points with functions defined on different intervals. We will compare the two different approaches:
The one with Cabri where it is possible to superimpose the graphic and the algebraic frames on the same page.
The one with TI Nspire where it is possible to display these different frames on different pages in the same screen

13 TI-Nspire Learning Technology

(Lecture 50 min)

Gosia Brothers, Texas Instruments, Dallas, TX, USA

Come to our presentation to learn more about TI-Nspire Technology.

14 Improving Algebraic Environment
(Workshop 90 min)
Gosia Brothers, Texas Instruments, Dallas, TX, USA

Come to our session to give us your feedback on TI-Nspire Algebraic Environment and discuss your wishes for the future versions of products from Texas Instruments.

15 Improving Graphs and Geometry
(Workshop 90 min)
John Good, TI-Team

Come to our session to give us your feedback on TI-Nspire Graphs and Geometry and discuss your wishes for the future versions of products from Texas Instruments.

16 Some Important Functionalities of a CAS when Teaching Mathematics to Future Engineers
(Lecture 25 min)
Michel Beaudin, ETS, Montreal, Canada

We are teaching mathematics at Ecole de technologie superieure (ETS), an engineering school in Montreal, Canada where every student has a Voyage 200 calculator on his desk and has access to computer labs where CAS like Derive, Maple and also Matlab program are installed. In Vienna (Visit-me 2002), we showed many examples of how Derive 5 and the TI-92 Plus were used when teaching to engineering students. For the past 4 years, we used the Voyage 200 and Derive 6.10 for teaching single and multiple variable calculus, linear algebra, differential equations, complex analysis. The talk will show examples of the importance of 2D implicit plotting, 3D plotting and Odes plotting when teaching to future engineers. These features are not yet implemented into Nspire and we hope that it will be on board soon. If we agree that we have to make a move from Derive to Nspire, we don´t agree to leave higher mathematics subjects to the competitors. ´

17 Nanopowder Production in a Plasmachemical Reactor: Computational Fluid Dynamics Modelling and Simulation.
(Lecture 25 min)
Phethedi J. Kekana; Andrei V. Kolesnikov; Stephen V. Joubert., TUT, Pretoria, South Africa

In this present study the development of a more realistic two-dimensional mathematical model capable of predicting the main aerosol phenomena such as TiO2 nanoparticle formation, growth and deposition in a high temperature plasmachemical reactor and the simulation results are presented. The TiO2 nanoparticles were produced in a tubular reactor from the oxidation of TiCl4 vapor in an Argon atmosphere. The developed mathematical model consists of mass, momentum and energy conservation equations. The particle dynamics processes include particle formation by nucleation, growth by condensation and coagulation as well as the loss of product particle by deposition on the wall of the plasmachemical reactor. The aim of this model was to predict both the axial and radial profiles of the flow velocity, temperature and concentrations profiles of TiCl4, O2, TiO2 and Cl2 and most importantly the evolution of the particle size distribution of TiO2 nanoparticle and TiO2 nanoparticle deposition to the reactor wall by numerical integration of stiff nonlinear partial differential equations

(PDE) in the FLUENT CFD program incorporating the Method of Moment technique. In this model the particle size distribution was approximated by the Gaussian lognormal function. The simulation results obtained from a two-dimensional model provides us with useful information on the influence of operational conditions (i.e. gas components flow rates, initial temperature and concentrations) and the reactor configurations on the evolution of TiO2 particle size distribution and the deposition flux at reasonable computational time and memory. The performance of the detailed two-dimensional model was validated by comparing its predicted results with the experimental and/or numerical data already published in the literature. Good correlations between the predicted and experimental results were achieved.

18 Using Science as a tool for learning mathematics

(Lecture 25 min)

Hildegard Urban, Vienna, Austria

Mathematics, as the language of numbers, is an important tool in science classes, but science is not generally considered as a tool for teaching mathematics. This article presents examples incorporating science concepts and problem solving in math classes using a motion detector (Calculator Based Ranger, CBR) and technology from Texas Instruments (TI-Nspire-CAS-handheld or. TI-Nspire-CAS-computer software). Real world data collection tools and Nspire introduce students to many fascinating concepts in mathematics and give them interactive ways to visualize relationships and patterns and enhance critical thinking. The author is investigating the mathematical and pedagogical potential of using TI technology (Graphical calculator, Voyage, Nspire) in combination with Vernier sensors and probes as devices to collect various kinds of data and of using the software to serve as a powerful analysis tool, helping students to build mathematical models. Experiences have been made in grade 9 to 11 (15- to 17-year old students) are reported. The use of technology seems to effectively enhance students´ learning. Students are actively engaged in learning as they make predictions, take measurements, analyze their data and make decisions about presenting their work. They are challenged to display their individual talents and mathematical abilities in real world problem solving situations.

19 The Many Dimensions of Decision Making A Process for Making Decisions in a Complex Environment

(Lecture 50 min)

Carl Leinbach, Professor Emeritus, Gettysburg College, PA, USA

In today's society no sound decision is made simply on the basis of one issue alone. Furthermore, reasonable people, looking at the situation may make entirely different decisions. There is no "one size fits all." The reason is that different people and different groups have different priorities and desires. The question we will consider is, how a decision maker can bring together these differing priorities into a group consensus. This presentation will present a process based on Thomas Saaty's Analytical Hierarchical Process for making complex decisions. It will begin with a brief discussion of the underlying mathematical assumptions of the process and apply the technique to the problem of choosing an afternoon tour at the TIME 2008 Conference. If time permits, the audience will participate in constructing a group priority for an issue of current interest.

20 Teaching Mathematics to Engineering Students: To Use or Not To Use TI-Nspire CAS

(Lecture 25 min)

Michel Beaudin and Gilles Picard, ETS, Montreal, Canada

We have been using the Voyage 200 to teach a variety of math topics in a Technical Engineering School (ETS in Montreal, Canada). We have to admit that the Voyage 200 is doing a very good job but, for some problems, a much faster processor would be helpful and would allow more powerful graphic options. This fast processor is exactly what we now have in the TI-Nspire CAS. So, when you need to solve some "heavy problems", say complicated equations, polynomial systems of equations, or when you want to define special functions using a definite integral, the Voyage 200 processor cannot compete with the TI-Nspire CAS calculator. The talk will give some examples of this, using Nspire CAS and comparing with Voyage 200 results.

Since we are teaching mathematics to future engineers, we also often need implicit 2D plots, 3D plots and differential equations plotting (along with RK and Euler numerical methods). We have all of this in the Voyage 200 but it can take a lot of time to get some results or see graphs on the screen and, in 3D plotting, the results are limited mainly due to a lack of processor speed. Adding these features to Nspire CAS, and colour for the PC version, would be important and, as far as we are concerned, this is a must. If this is not done, 3D plots will continue to be done, by our colleagues and us, using Derive or Maple software. In order to make the move to Nspire CAS software, we absolutely need a much more "university level package".

21 Modelling of the telegraph equations in transmission lines

(Lecture 25 min)

Corrie Lock, Proff JC Greeff, SV Joubert UJ,TUT,TUT, South Africa

It would be difficult to imagine a world without communication systems. A plethora of guided fixed line telephones as well as a multitude of unguided systems to serve cellular phones are evident in our surrounding world. In order to optimise guided communication systems, it is necessary to determine or project power and signal losses in the system, since all systems have such losses. To determine these losses and eventually ensure a maximum output, it is necessary to formulate some kind of equation with which to calculate these losses. A mathematical derivation for the telegraph equation in terms of voltage and current for a section of a transmission line will be investigated.

In literature for engineers consulted, the formulae for voltage and current involved in the telegraphic equations are not explicitly and analytically derived, leaving a theoretical gap seldom crossed by students in Electrical Engineering.

The main aim is to address this theoretical gap, and derive from basic principles, the equations for telegraphic transmission in a guided system and secondarily to illustrate the applications thereof to real-world problems using a suitable computer algebra system in this case, Derive.

⋮
⋮

22 Comparison of an Analytical Method and Matlab to Model Electromagnetic Distribution in a Trough

(Lecture 25 min)

JJ Bruyns, J.C. Greeff, S.V. Joubert, UJ,TUT,TUT, South Africa

In designing devices that discharge electrostatic voltage it is important to know the radiation pattern. To interpret and visualize the result is just as important as being able to obtain a correct analytical result. This lecture discusses the traditional analytical method to determine the radiation pattern in a closed trough, as well as the use of a computer algebra system (CAS) to determine the radiation pattern. MATLAB is used as a computer algebra system to solve and visually display the electrostatic distribution in a trough.
The results of the analytical and numerical solutions are compared to determine the accuracy of the CAS solution. The use of the CAS system and its educational advantage is explained.

23 Teaching differential equations and its application - Using Derive 6 as a PeCAS

(Lecture 25 min)

Jose Luis Galan, Gabriel Aguilera et al, University of Malaga, Malaga, Spain

In this talk we will describe the file DIFFERENTIAL_EQUATIONS.MTH, created in DERIVE 6 in order to be used in mathematical subjects which deal with differential equations, aimed at Engineering students. Such file contains a series of programs which permit to solve differential equations problems.

The programs contained in the file can be grouped within the following blocks:

First order differential equations: separable equations and equations reducible to them, homogeneous equations and equations reducible to them, exact differential equations and equations reducible to them (integrating factor technique), linear equations, the Bernoulli equation, the Riccati equation.

- First order differential equations and nth degree in y´.
- Generic programs to solve first order differential equations.
- Cauchy problems for first order differential equations.
- Higher orders differential equations.
- Cauchy problems for higher orders differential equations.
- Applications of differential equations.

We will also show in the talk some examples of applications that have been carried out with our students of Telecommunication Engineering.

The programs have been developed using the Display function in order to be used as didactical tools with explications of what the programs do step by step, using DERIVE 6 as a Pedagogical CAS (PeCAS) or as a white-box CAS.

Finally, we include the conclusions obtained after using this file with our students and also some future work on this subject.

24 DERIVE 6 as a pedagogical CAS: using the slide bar utility and the DISPLAY function in programming

(Workshop 90 min)

Jose Luis Galan, Gabriel Aguilera et al, University of Malaga, Malaga, Spain

In this workshop we will use some examples of programming with DERIVE 6 using the display function in order to use this software as a pedagogical CAS (PeCAS). We have developed this kind of workshop with our students of Technical Telecommunication Engineering.

The main innovative aspect of this way of teaching is that students have an active role. Specifically, they have to elaborate, by themselves, utility files to solve typical problems for different subjects. In our case, this fact implies that students need to deal with programming in DERIVE 6, understand the subject and know how to solve typical problems.

This workshop will consist of two different parts:

1. Drawing classical curves using the slide bar utility.

In this first part, we will draw curves such as segment, circumference, ellipse, lemniscate, astroid, cardioid, catenary, cycloid, cissoid, folium, eight curve, limacon, rhodonea curves, hypocycloid, trisectrix, tractrix, spiral, …

Changing, with the slide bar utility, the parameter(s) of a curve, we can study properly its properties.

2. Programming and computing line integrals using the display function

The second part of the workshop will consist of the development of different programs in Derive 6 in order to solve any line integral considering both, computing the integral by means of the definition and/or using an appropriate theorem to compute it properly.

We will develop the programs using the display function which will allow us to use DERIVE 6 as a PeCAS.

We will provide some material to make the workshop easier to follow.

25 New Kids on the Block – Some First Experiences with Recent Alternatives to DERIVE

(Lecture 25 min)

Karsten Schmidt, Schmalkalden University of Applied Sciences, Germany

Although Texas Instruments finally discontinued DERIVE last summer, numerous DERIVE users are still working with this popular Computer Algebra System for a variety of reasons. But in addition to the well-known DERIVE competitors – Mathematica, Maple, and MuPAD – there are now new alternatives in the Computer Algebra System market, such as Nspire and WIRIS.

In this presentation we reflect on possible difficulties in the transition process, from a more organizational point of view (Is the licensing system appropriate? What is the price tag? How easy is the installation?), as well as from a didactic point of view (How intuitive is the software for inexperienced users? How much class time is required before the respective Computer Algebra System can be used effectively?).

These considerations are made from the background of a first-year university course in linear algebra which was transformed step by step over a period of ten years from a traditional "blackboard and transparencies" teaching approach (in a lecture hall standing in front of up to 70 students) into an interactive teaching approach using DERIVE (in a PC lab with 20 PCs and no more than 40 students).

We will also be looking at the question as to whether the new kids on the block have something interesting to offer in this context which is not available in DERIVE.

26 Introducing a computer algebra system in mathematics education – empirical evidence from Thuringia (Germany)

(Lecture 50 min)

Dr. Wolfgang Moldenhauer, Lehrplanentwicklung und Medien, Heinrich-Heine-Allee 2-4, 99438 Bad Berka, Germany

This lecture reports on the effects the use of a pocket calculator-based computer algebra system (CAS) has on the performance in mathematics of grade 11 students in Thuringia. A project started at 8 of about one hundred upper secondary schools in the federal state of Thuringia in 1999; 3 years later the former restrictions on the use of technology in math education were lifted. In 2004, more than a quarter of all Thuringian upper secondary schools used CAS in math classes. Beginning in 2000, a test was carried out each year to compare the performance of CAS and non-CAS students (from different control schools). More than 12000 students were tested. In 70% of the cases CAS students performed better than, and in the remaining 30% they performed as well as, non-CAS students. There is evidence that students in advanced courses benefit more from using CAS than students in basic courses.

27 A novel method of interpolation and extrapolation of functions by a linear initial value problem

(Lecture 50 min)

Michael Shatalov, Igor Fedotov and Stephan V. Joubert, CSIR and TUT, South Africa

The classical approach to function approximation is based on a particular choice of functions, for example polynomial, rational, exponential functions or Fourier series. The main advantage of these methods is to obtain the approximation expressions in a closed form. There are several disadvantages to the classical approach. For example, polynomial interpolation may seldom be used for the purposes of extrapolation due to the fast divergence of higher order polynomials outside of the interpolation interval. The main disadvantage of a Fourier series approximation is that it is not applicable to non-periodic functions and hence, could not be used for extrapolation purposes, et cetera. The method we propose allows us to approximate functions by means of linear combination of polynomials, trigonometric and exponential functions, products of polynomials and exponents, polynomials and periodic functions, periodic functions and exponents, and polynomials, exponents and periodic functions et cetera. It is well suited for the purposes of interpolation and extrapolation of physical and chemical processes, which are described in terms of systems of linearized ordinary differential equations (ODE). The main idea of the proposed method is the approximation of a function on a fixed interval by means of linear ODE with unknown constant coefficients. Initial values of the problem are also considered as unknowns. The goal function is formulated as a positive definite function with non-negative weight function. Unknown coefficients and initial conditions are defined by means of minimization of the goal function. Examples of practical approximation of functions are considered and compared with available commercial algorithms of interpolation, extrapolation and smoothing. The methods we discuss will be readily understood by undergraduate students that have been taught ODE using DERIVE or some other CAS.

28 A CAS Approach to Understanding from Beginning Algebra to Advanced Calculus and Abstract Algebra

(Lecture 50 min)

William C. Bauldry and Wade Ellis, Appalachian State University (NC) and West Valley College (CA), USA

The presenters will use TI-Nspire CAS to demonstrate ways to foster and enhance student understanding of mathematics in courses from secondary school Beginning Algebra to university Linear Algebra, Advanced Calculus, and Abstract Algebra courses. The presentation will involve the Action/Consequence/Reflection Principle developed by Tom Dick and Gail Burrill whereby students act on mathematical objects, transparently observe the consequences of their actions, and then reflect on the mathematical meaning of those consequences.

29 Heat transfer in a one dimensional domain of variable cross-sections

(Lecture 25 min)

RS Lebelo, I Fedotov, M Shatalov and HM Djouosseu Tenkam, TUT, Pretoria, South Africa

In this paper the method of approximating solutions of partial differential equations with variable coefficients is studied. This is done by considering heat flow through a one dimensional domain model, with variable cross-sections of N sections. The heat transfer process is described by heat equation. This study is based on finding eigenvalues using software such as Derive and Mathcad. The corresponding eigenfunctions automatically satisfy the boundary conditions at the endpoints and boundary conditions of the first kind at the endpoints will be considered for the conic section. The authors show how a student can solve different eigenvalues and eigenfunctions using above mentioned softwares. This is a central point of finding the analytical solution of partial differential equations.

30 Dipstick Readings

(Lecture 25 min)

Rambane D.T., TUT, Pretoria, South Africa

In garages fuel (petrol or diesel) is often stored in cyclindrical tanks underground. To measure the volume of fuel in the tank a deepstick is used. We investigate the mathematics involved and how Derive can be used.

31 A probabilistic approach to function approximation

(Lecture 25 min)

PH Kloppers, TH Fay, SV Joubert, TUT, South Africa

In this talk we will investigate a new and novel approach to function approximation for functions defined over a bounded real interval which, without loss of generality is assume to be the unit interval [0,1]. This approach is based upon an idea by J. Kolibal and C. Saltiel. We will give a derivation of their technique and show how to interpret it statistically using the Gaussian distribution. Since Bernstein polynomials lie at the foundation of the Kolibal-Saltiel technique which, they call the Bernstein function approximation, we will call our technique the Gauss-Bernstein approximation technique.

These ideas are quite general and have wide applicability in function approximation, high frequency filtering, data interpolation and multidimensional data regularization.

This technique can be taught at the undergraduate level because the use of a computer algebra system such as Mathematica is essential for the purposes of visualization and easing any algebraic, technical barriers that students may encounter.

32 Deep Learning and Fun in First Year using Maple

(Lecture 50 min)

Bill Blyth and Alexandra Labovic, RMIT University, Melbourne, Australia

Mathematics educators and their students need to embrace technology. Studies have shown that some students have some negative attitudes towards using computer packages and programming. Our objective is for our graduates to be expert users of the powerful CAS (computer algebra system), Maple. Thus we want to ensure that students´ initial experience is positive - particularly in the first semester when any negative attitudes need to be overcome.

In the first semester of a traditional calculus course, the weekly Maple lab sessions are not used to directly support the lectures ... nearly half of the work is at school level! The purpose is for the students to enjoy the experience of using Maple.

Student work in groups of size 2 to 4. After Maple introductions, they complete an Introduction to Animation session and then choose an extended animation project from a list of five problems. They have to demonstrate their animations in the lab for assessment. Students enjoy the animation project.

Following the animation projects, Spot the Curve uses plots and animations to understand horizontal and vertical translation of curves: students identify the translation used and appreciate automatic marking within Maple. Student feedback has been very positive.

Our trapezoidal rule assignment is now more fun: it´s disguised as a Fish Pond (a trout farm). Trapezoidal rule is used to approximate the cross-sectional area (and hence the number of trout) - the students download a template for individualized fish ponds, with automatic marking. These projects are enjoyable deep learning activities.

33 Remarks on Duffing's Equation

(Lecture 50 min)

TH Fay, TUT, South Africa

We discuss some of the interesting features of the forced Duffing equation

d/dt(dx/dt) + k*dx/dt-a*x + x^3=F*cos(b*t)

through numerical investigations using a computer algebra system (in our case Mathematica version 5.0). We discuss the unforced damped and undamped equation briefly, and concentrate on solving numerically the differential equation for a variety of values of the parameter F holding all other parameters fixed and generally holding the initial conditions for x(0) and x(0) to be the "at rest" conditions of zero. In doing so, many striking and fascinating trajectories representing interesting motions and other phenomena can be discovered including: stability, periodic solutions (both harmonic and subharmonic), almost periodic solutions, and aperiodic solutions. In particular, chaos is often claimed to be evident in the trajectories and solutions of this Duffing equation and it is the purpose of this article to elaborate on this. These studies naturally give rise to computer laboratory problems suitable for student research and small group projects. Numerical investigations should go hand-in-hand with theoretical studies as the one cross fertilizes the other. As an Addendum, a list of student research problems is attached.

Accepted Papers for the ACDCA strand of TIME 2008

Introduction to TI-Nspire CAS, Bernhard Kutzler , Workshop 90 min

Exercising Control: Didactical Influences, Kathleen Pineau , Lecture 50 min

CAS and calculation competence of students, Rainer Heinrich , Lecture 50 min

Basic Skills and CAS, Josef Böhm, Lecture 50 min

Linking geometry, algebra and calculus with GeoGebra, Josef Böhm, Lecture 25 min

Modelling Cha Cha dance with Cabri 3D, Jean-Jacques Dahan, Workshop 90 min
(was moved to the DERIVE & TI-conference in order to accomplish Jean-Jacques lecture on Modelling Cha Cha Cha with CabriII+ and NSpire)

Spreadsheets and Interactive White Boards in the Primary Mathematics Classroom,
Philip Oostenbroek, Br Adrian Story, Dr Anne Williams (Lecture 50 min)

Why Proof in Dynamic Geometry?, Michael de Villiers, Lecture 25 min

Argumentation Schemes and the Use of Sketchpad, Angel Homero, Lecture 25 min

Neither a Tractor, nor a Matrix but a Tractrix!, Susan Steyn, Lecture 25 min

Sustainability of mathematics education by using technology demonstrated with the topic of exponential growth, Helmut Heugl, Lecture 50 min

Tasks in Calculus: Results of a 9-Year Evolution,
Genevive Savard and Kathleen Pineau, Lecture 25 min

Game proramming / a "future" method to teach mathematics, Sofia Backstrom & Marie Rudenstam, Lecture 50 min

CAS-exercises during the Central Examination in North Rhine-Westphalia (Germany),
Dirk Warthmann, Lecture 25 min

Roots of transcendental algebraic equations: A method of bracketing roots and selecting initial estimations, J.N. Mwambakana, M. Shatalov, and I. Fedotov, Lecture 25 min

Application of eigenfunction orthogonalities to vibration problems, H.M Djouosseu Tenkam, I. Fedotov, M. Shatalov, Lecture 25 min

A School-Oriented Review of Computer Algebra Systems for Solving Equations and Simplifications. Issues of Domain, Eno Tonisson, Lecture 50 min

Defining a stability boundary for three species competition models, Quay van der Hoff, Johanna C. Greeff, Lecture 25 min

Technology - can it be trusted?, Johanna C Greeff, Stephan V Joubert, Lecture 25 min

Identification of Dynamical Systems Parameters from Experimental Data using Numerical Methods, Pete N.A and Fedotov I, Lecture 25 min

Numerical Computation of Special Functions with Application to Physics, Motsepe KA; Fedotov I; Shatlov M., Lecture 25 min

An error analysis of the numerical method of lines, Judith N.M. Bidie, Temple H. Fay,
Stephan V. Joubert, Lecture 25 min

Using Matlab for teaching Mathematical Modelling, Ansie Harding, Lecture 25 min

Recurring Decimals, etc. and Fractions in Derive 6 and on the TI-89

Peter Schofield, Trinity & All Saints, Leeds, UK

Recurring decimal expansions of rational numbers is a key topic in the teaching of numbers and number systems. It is therefore surprising that neither Derive nor the TI-89 appear to have a method of displaying a recurring decimal in a precise form. One of the most common notations for recurring decimals is to put a dot over the first and last digit of the recurring string (for example, $1.2\dot{3}45\dot{6}$). To convert this into a notation suitable for Derive and the TI-89 note that what is required is an indicator as to where the fixed digits end and the recurring digits begin. A single quotation mark will suffice and so, in Derive notation, $1.2\dot{3}45\dot{6}$ becomes "1.2'3456" (note the string format - at the Users level one has to resort to strings to display this form of notation). There are two transforms involved, Decimals to Quotients and Quotients to Decimals.

1. Decimals to Quotients in Derive 6

In Derive there does not seem to be an instruction (like "expr" on the TI-89) for converting a suitable string into a number. However, for non-negative integers, it is not difficult to construct a simple procedure to do this:

```
EXPR(x, n := 0) :=
  Prog
    x := NAME_TO_CODES(x)
    Loop
      If x = []
        RETURN n
      n := 10·n + FIRST(x) - 48
      x := REST(x)
```

This might be a shorter function:

```
expr(x) := APPEND(CODES_TO_NAME(NAME_TO_CODES(x)))
```

In the following main procedure, after storing the sign of the decimal input in s, the input string is split into three sub-strings: a – digits before the decimal point; b - fixed digits after the decimal point; x - recurring digits. These are converted into numbers using EXPR (b and x are provided with appropriate denominators) and the rest is left to the exact arithmetic rational number calculator of Derive 6.

```
DtoQ(x, flag := 0, a := "", b := "", s := 1, c, d) :=
  Prog
    If FIRST(x) = "-"
      [s := -1, x := REST(x)]
    Loop
      If FIRST(x) = "." ∨ x = ""
        Prog
          x := REST(x)
          Loop
            If FIRST(x) = "'" ∨ x = ""
              Prog
                x := REST(x)
                [c := 10^DIM(b), d := 10^DIM(x) - 1]
                If flag = 0
                  [a := EXPR(a), b := EXPR(b), x := EXPR(x)]
                RETURN s·(a + b/c + IF(d = 0, 0, x/(c·d)))
            [b := APPEND(b, FIRST(x)), x := REST(x)]
      [a := APPEND(a, FIRST(x)), x := REST(x)]
```

For example, Simplify DtoQ("1.2'3456") to obtain: $\frac{6858}{5555}$

I have also made use of a simple flag variable (default value = 0) so that, when flag = 1, the procedure attempts to display the arithmetic structure of the calculation.

For example, Simplify DtoQ("1.2'3456",1) to obtain: $1 + \frac{2}{10} + \frac{3456}{99990}$.

2. Quotients to Decimals in Derive 6

To form a decimal expansion of a fraction the following procedure uses the standard method of dividing of the denominator into a sequence of numerators (scaled up by base10 each time). If (at some stage) the remainder becomes zero the decimal terminates, if a non-zero remainder is repeated then the decimal becomes recurring.

```
QtoD(x, flag := 0, a, b, c, d := "", a_list := []) :=
  Prog
    If x = 0
       RETURN "0"
    [a := NUMERATOR(ABS(x)), b := DENOMINATOR(ABS(x))]
    c := APPEND(IF(x < 0, "-", ""), STRING(FLOOR(a, b)))
    Loop
      a := MOD(a, b)
      If a = 0
         If flag = 0
            If d = ""
               RETURN c
               RETURN APPEND(c, ".", d)
            RETURN APPEND(a_list, [a])
      If MEMBER?(a, a_list)
         If flag = 0
            RETURN APPEND(c, ".", INSERT("'", d, POSITION(a, a_list)))
            RETURN APPEND(a_list, [a])
      a_list := APPEND(a_list, [a])
      [a := 10, d := APPEND(d, STRING(FLOOR(a, b)))]
```

In this case, it does not matter whether you Simplify or Approximate since Derive is returning a string. Try Simplifying QtoD(6858/5555) to obtain "1.2'3456". In fact, in the Algebra window, the double quotes are not shown – so this looks like a number!

Remarks

(i) QtoD simplifies the recurring decimal format down to its shortest (simplest) form. For example, QtoD(DtoQ("1.2012'012")) Simplifies to "1.2'0120".

(ii) Setting flag = 1 in QtoD will display the sequence of remainders used in the calculation of the decimal expansion string. For example, Simplify QtoD(1/7,1) to display:

[1, 3, 2, 6, 4, 5, 1].

(iii) I never cease to be amazed at the calculating power of Derive. On my laptop Derive processed the 4293 symbols of QtoD(777/8777) in less than 2 seconds!

(iv) A useful instruction for investigations is:

$$D\_list(n) := \left[VECTOR\left(QtoD(m \cdot n),\ m,\ \frac{1}{n} \right) \right]'$$

Simplify D_list(1/7) - and note how the six recurring digits of 1/7 are repeated cyclically in the multiples of 1/7.

3. Decimals to Quotients and Quotients to Decimals on the TI-89

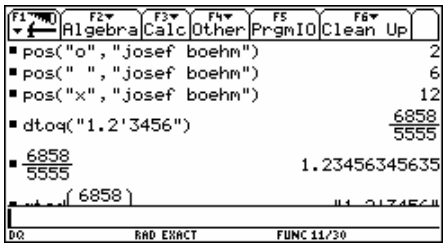
Having successfully composed and tested the above procedures in Derive 6 I thought of trying something similar on my TI-89 Titanium calculator. Using 2nd>VAR-LINK>f1 I first created a new folder DQ to store the three function procedures below. It is also useful to make DQ the default folder before entering or transferring the functions. The TI-89 does not appear to have a function to calculate the first position of a character in a string or member in a list. The function pos(a,b) is a DIY (Do It Yourself) version of this.

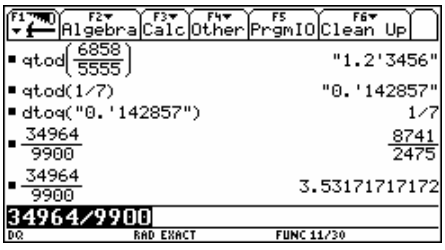
```
:pos(a,b)
:Func
:Local m,n
:1→n:dim(b)→m
:While a≠mid(b,n,1)
:If n>m Then:Return n
:EndIf
:n+1→n
:EndWhile
:Return n
:EndFunc
```

```
:dtoq(x)
:Func
:Local a,b,c,d,s
:dim(x)→a
:If left(x,1)="-" Then
:-1→s:a-1→a:right(x,a)→x
:Else:1→s:EndIf
:dq\pos(".",x)+1→b
:dq\pos("'",x)→c
:If c<a Then
:10^(c-b)*(10^(a-c)-1)→d
:expr(right(x,a-c))/d→d
:Else::0→d:EndIf
:If c>2 Then
:expr(left(x,c-1))→c
:Else:0→c:EndIf
:Return s*(c+d)
:EndFunc
```

```
:qtod(x)
:Func
:Local a,b,c,d,p,q,e
:If x=0 Then:Return "0"
:EndIf
:{}→e:""→d:""→c:0→q
:getNum(abs(x))→a
:getDenom(abs(x))→b
:If x<0 Then:"-"→c:EndIf
:c&string(floor(a/b))→c
:Loop
:mod(a,b)→a
:If a=0 Then
:If d="" Then:Return c
:Else:Return c&"."&d
:EndIf:EndIf
:dq\pos({a},e)→p
:If p≤q Then
:left(d,q-p)→a
:right(d,p)→b
:Return c&"."&a&"'"&b
:EndIf
:augment({a},e)→e
:q+1→q:10*a→a
:d&string(floor(a/b))→d
:EndLoop:EndFunc
```

Opting for functions allows the operations to be carried out on the HOME screen – this is useful for testing inverse properties of dtoq and qtod. Although the TI-89 is much slower and the display more limited you can still, for example, display the recurring decimal expansions of multiples of 1/7 by ENTERing seq(qtod(m/7),m,1,7). However, on the TI-89, there does not appear to be a method of setting up a flag variable with a default value – hence these functions only display the end results.





4. Recurring Expansions using General Number Bases in Derive 6

In Derive (using Options>Mode Settings>…) it is possible to set the InputBase and OutputBase to any integer between 2 and 36, inclusive. To adapt the Derive 6 procedures (in sections 1 and 2 above) to work with these we first need a function to convert a base setting into a strict number output:

```
C_B(b) := [IF(b = Decimal, RETURN 10), IF(b = Binary, RETURN 2), IF(b = Octal,
   RETURN 8), IF(b = Hexadecimal, RETURN 16), RETURN b]
```

Then, in the local variable list of each procedure, add an assigned variable

```
base := C_B(OutputBase)
```

and within the procedure replace each occurrence of "10" by "base".

This accommodates number bases ≤ 10.

For number bases > 10 (decimal) Derive uses letters A, B, C, … for successive digits beyond 9 and, if the leading digit is >9, the number is prefixed by a zero. To follow this notation extend EXPR to:

```
EXPR(x, n := 0, d, base := C_B(OutputBase)) :=
  Prog
    x := NAME_TO_CODES(x)
    Loop
      If x = []
        RETURN n
      d := FIRST(x) - IF(FIRST(x) > 64, 55, 48)
      If 0 ≤ d < base
        n := base·n + d
        RETURN "DIGIT ERROR"
      x := REST(x)
```

and alter the last line of QtoD to:

```
[a :* base, d := APPEND(d, (STRING(FLOOR(a, b)))↓(-1))]
```

Using alternative number bases can be confusing. In Derive, a straightforward method is to set the InputBase to Decimal and select the OutputBase.

For example, with InputBase:=Decimal and OutputBase:=Octal enter: `DtoQ(QtoD(17/33))`.

In the Algebra window this is displayed as: $\mathrm{DtoQ}\left(\mathrm{QtoD}\left(\frac{21}{41}\right)\right)$

Highlighting and Simplifying QtoD… yields: `DtoQ(0.'40760337017)`.

(This is the Octal recurring expansion of the Octal fraction!)

and Simplifying again gets back to: $\frac{21}{41}$

In the reverse direction you have follow Derive notation when entering the string.

For example, with OutputBase:=Hexadecimal enter: `QtoD(DtoQ("0E1.ABC'123"))`.

In the Algebra window this is displayed as: `QtoD(DtoQ(0E1.ABC'123))` .

Highlighting and Simplifying DtoQ… yields: $\mathrm{QtoD}\left(\frac{4B348CCD}{555000}\right)$

(The fraction is Hexadecimal.)

and Simplifying again gets back to: `0E1.ABC'123`.

Opportunities for experimentation are many.
Try some activities using OutputBase:=Binary.
(On the TI-89, to carry out similar activities using MODE settings of Binary or Hexadecimal, you would have to rewrite parts of the coding for the functions dtoq and qtod.)

Working with these procedures I have extended my own experience of recurring decimals and recurring expansions well beyond hand-calculated examples. In my opinion, some form of precise recurring decimal/expansion notation ought to be a part of the number format of any IT product dedicated to mathematics and mathematical education.

Reading Peter´s contribution on periodic decimal numbers I remembered the procedure which I learned and applied in school time. See two examples:

$$\begin{aligned} x &= 3{,}53\overline{171} \\ 100x &= 353{,}171\,171\ldots\ldots \\ 100000x &= 353171{,}171171\ldots\ldots \\ \hline 99900x &= 352818 \\ x &= \frac{352818}{99900} = \frac{19601}{5550} \end{aligned} \qquad \begin{aligned} y &= 0{,}2\overline{3456} \\ 10y &= 2{,}3456\,3456\ldots\ldots \\ 100000y &= 23456{,}3456\,3456\ldots\ldots \\ \hline 99990y &= 23454 \\ y &= \frac{23454}{99990} = \frac{1303}{5555} \end{aligned}$$

Now I tried to transfer this algorithm to DERIVE step by step. First I am doing it really stepwise and then I put all together to a function:

```
#1:   Notation := Decimal

#2:   PrecisionDigits := 20

#3:   NotationDigits := 20

#4:   [x := 3.53'171, y := 0.2'3456, w := 10.'23]

#5:   codes(z) := NAME_TO_CODES(z)
```

Positions of decimal point and quotation mark

```
#6:   [dp(z) := POSITION(., z), qp(z) := POSITION(', z)]
```

$$\#7: \quad \begin{bmatrix} dp(x) & qp(x) \\ dp(y) & qp(y) \\ dp(w) & qp(w) \end{bmatrix} = \begin{bmatrix} 2 & 5 \\ 2 & 4 \\ 3 & 4 \end{bmatrix}$$

The integer part of the number

```
      int_p(z) := CODES_TO_NAME(VECTOR((codes(z)) , i, dp(z) - 1))
#8:                                                    i

#9:   [int_p(x), int_p(y), int_p(w)] = [3, 0, 10]
```

Lengths of preperiod and period

```
#10:  [ppl(z) := qp(z) - dp(z) - 1, pl(z) := DIM(z) - qp(z)]
```

$$\#11:\quad \begin{bmatrix} ppl(x) & pl(x) \\ ppl(y) & pl(y) \\ ppl(w) & pl(w) \end{bmatrix} = \begin{bmatrix} 2 & 3 \\ 1 & 4 \\ 0 & 2 \end{bmatrix}$$

Values of preperiod and period

```
      pp(z) :=
        If ppl(z) = 0
#12:        0
            CODES_TO_NAME(VECTOR((codes(z))↓i,i, dp(z) + 1, qp(z) - 1))/10^ppl(z)
```

$$\#13:\quad per(z) := \frac{\text{CODES\_TO\_NAME}(\text{VECTOR}((codes(z))_{i}, i, qp(z) + 1, DIM(z)))}{10^{DIM(z) - dp(z) - 1}}$$

$$\#14:\quad \begin{bmatrix} pp(x) & per(x) \\ pp(y) & per(y) \\ pp(w) & per(w) \end{bmatrix} = \begin{bmatrix} 0.53 & 0.00171 \\ 0.2 & 0.03456 \\ 0 & 0.23 \end{bmatrix}$$

The truncated number (including one full period) is:

```
#15:  tr_numb(z) := int_p(z) + pp(z) + per(z)

#16:  [tr_numb(x), tr_numb(y), tr_numb(w)] = [3.53171, 0.23456, 10.23]
```

Shifting decimal point at the begin of the period by a multiplication by the respective power of 10

$$\#17:\quad \begin{bmatrix} tr\_numb(x)\cdot 10^{ppl(x)} \\ tr\_numb(y)\cdot 10^{ppl(y)} \\ tr\_numb(w)\cdot 10^{ppl(w)} \end{bmatrix} = \begin{bmatrix} 353.171 \\ 2.3456 \\ 10.23 \end{bmatrix}$$

Shifting decimal point at the begin of the 2nd period by a multiplication by the respective power of 10

$$\#18:\quad \begin{bmatrix} tr\_numb(x)\cdot 10^{ppl(x)+pl(x)} + per(x)\cdot 10^{ppl(x)} \\ tr\_numb(y)\cdot 10^{ppl(y)+pl(y)} + per(y)\cdot 10^{ppl(y)} \\ tr\_numb(w)\cdot 10^{ppl(w)+pl(w)} + per(w)\cdot 10^{ppl(w)} \end{bmatrix} = \begin{bmatrix} 353171.171 \\ 23456.3456 \\ 1023.23 \end{bmatrix}$$

Building the difference of the shifted numbers

#19:
$$\begin{bmatrix} \mathrm{tr\_numb}(x)\cdot 10^{\mathrm{ppl}(x)+\mathrm{pl}(x)} + \mathrm{per}(x)\cdot 10^{\mathrm{ppl}(x)} - \mathrm{tr\_numb}(x)\cdot 10^{\mathrm{ppl}(x)} \\ \mathrm{tr\_numb}(y)\cdot 10^{\mathrm{ppl}(y)+\mathrm{pl}(y)} + \mathrm{per}(y)\cdot 10^{\mathrm{ppl}(y)} - \mathrm{tr\_numb}(y)\cdot 10^{\mathrm{ppl}(y)} \\ \mathrm{tr\_numb}(w)\cdot 10^{\mathrm{ppl}(w)+\mathrm{pl}(w)} + \mathrm{per}(w)\cdot 10^{\mathrm{ppl}(w)} - \mathrm{tr\_numb}(w)\cdot 10^{\mathrm{ppl}(w)} \end{bmatrix}$$

#20:
$$\begin{bmatrix} 352818 \\ 23454 \\ 1013 \end{bmatrix}$$

and finally dividing them by the respective power of 10 and changing to Output Rational in order to receive the fraction:

#21: `Notation := Rational`

#22:
$$\begin{bmatrix} \dfrac{\mathrm{tr\_numb}(x)\cdot 10^{\mathrm{ppl}(x)+\mathrm{pl}(x)} + \mathrm{per}(x)\cdot 10^{\mathrm{ppl}(x)} - \mathrm{tr\_numb}(x)\cdot 10^{\mathrm{ppl}(x)}}{10^{\mathrm{ppl}(x)+\mathrm{pl}(x)} - 10^{\mathrm{ppl}(x)}} \\ \dfrac{\mathrm{tr\_numb}(y)\cdot 10^{\mathrm{ppl}(y)+\mathrm{pl}(y)} + \mathrm{per}(y)\cdot 10^{\mathrm{ppl}(y)} - \mathrm{tr\_numb}(y)\cdot 10^{\mathrm{ppl}(y)}}{10^{\mathrm{ppl}(y)+\mathrm{pl}(y)} - 10^{\mathrm{ppl}(y)}} \\ \dfrac{\mathrm{tr\_numb}(w)\cdot 10^{\mathrm{ppl}(w)+\mathrm{pl}(w)} + \mathrm{per}(w)\cdot 10^{\mathrm{ppl}(w)} - \mathrm{tr\_numb}(w)\cdot 10^{\mathrm{ppl}(w)}}{10^{\mathrm{ppl}(w)+\mathrm{pl}(w)} - 10^{\mathrm{ppl}(w)}} \end{bmatrix}$$

We see that we can cancel by 10^ppl(w) and then we obtain a nice formula:

#23:
$$\begin{bmatrix} \dfrac{\mathrm{tr\_numb}(x)\cdot 10^{\mathrm{pl}(x)} + \mathrm{per}(x) - \mathrm{tr\_numb}(x)}{10^{\mathrm{pl}(x)} - 1} \\ \dfrac{\mathrm{tr\_numb}(y)\cdot 10^{\mathrm{pl}(y)} + \mathrm{per}(y) - \mathrm{tr\_numb}(y)}{10^{\mathrm{pl}(y)} - 1} \\ \dfrac{\mathrm{tr\_numb}(w)\cdot 10^{\mathrm{pl}(w)} + \mathrm{per}(w) - \mathrm{tr\_numb}(w)}{10^{\mathrm{pl}(w)} - 1} \end{bmatrix} = \begin{bmatrix} \dfrac{19601}{5550} \\ \dfrac{1303}{5555} \\ \dfrac{1013}{99} \end{bmatrix}$$

Which can be rewritten as

```
      fract(z) := tr_numb(z) + ──────────
#24:                            pl(z)
                               10     - 1
```

```
#25:  [fract(x), fract(y), fract(w)] = [ 19601/5550, 1303/5555, 1013/99 ]
```

I collect the whole procedure in a function:

```
      dn_to_fr(z, z_, dp, qp, ppl, pl, int_p, pp, per) :=
        Prog
          [z_ := NAME_TO_CODES(z), dp := POSITION(".", z), qp := POSITION("'", z)]
          int_p := CODES_TO_NAME(VECTOR(z_↓i, i, dp - 1))
#26:      [ppl := qp - dp - 1, pl := DIM(z) - qp]
          pp := IF(ppl = 0,0,CODES_TO_NAME(VECTOR(z_↓i,i,dp + 1, qp - 1))/10^ppl)
          per := CODES_TO_NAME(VECTOR(z_↓i,i,qp + 1,DIM(z)))/10^(DIM(z) - dp - 1)
          int_p + pp + per + per/(10^pl - 1)

#27:  [dn_to_fr(x), dn_to_fr(y), dn_to_fr(w)]

      [ 19601    1303    1013 ]
#28:  [──────, ──────, ──────]
      [ 5550     5555     99  ]

#29:  [3.5317117117117117117, 0.23456345634563456345, 10.232232323232323232]
```

Let´s try a special case:

```
#30:  dn_to_fr(10.'9) = 11
```

I tested my function copying the decimal expansion of 777/8777 (= xx consisting of 4293 numbers) from Peter´s file and tried to convert it to a fraction

```
75549732254756750598154266833770080893243705138429987467243933006722114617750939956705024495841403 66~
86795032471231628118947248490372564657627891078956363222057650677908169078272758345676199156887319 12~
95431240742850632334510652842656944286202574911701036800729178534806881622422239945311609889483878 31~
83320041016292582887091261250996923778056283468155406175230716645778739888344536857696251566594508 37~
41597356727811325054118719380198245414150620941096046485131593938703429417796513615130454597242793 66~
52614788652159052067904751053890851088071094907143670958186168394667881964224678136037370399 9
```

```
#32:  dn_to_fr(xx) = 777/8777
```

It took 0.125 sec calculation time. (Peter´s function DtoQ needs 0.234 sec).

Is there also another way for the reverse task?

Students might find out – analysing the calculation from page 22 – that the uncancelled denominator of the resulting fraction is always a number starting with a sequence of *m* nines followed by a sequence of *n* zeros. The number of zeros gives the length of the preperiod and the number of nines the length of the period. (Let them explain, why.)

So we are looking for the minimum multiple of the given denominator which can be written as such a sequence of *m* nines and *n* zeros.

My first task was to program an algorithm which delivers these possible denominators:

```
nines(k_, l := [9], l0, l1 := [9], a, a0, k, k_, i, flag) :=
  Prog
    l := [9]
    l1 := [9]
    k := 1
    Loop
      l0 := l1
      a0 := FIRST(l0)·10
      l1 := [a0]
#33:  Loop
        If DIM(l0) = 0 exit
        a := a0 + FIRST(l0)
        l1 := APPEND(l1, [a])
        l0 := REST(l0)
      l := APPEND(l, l1)
      k :+ 1
      If k = k_
         RETURN l
```

#34: nines(8) = [9, 90, 99, 900, 990, 999, 9000, 9900, 9990, 9999, 90000, 99000, 99900, 99990, 99999, 900000, 990000, 999000, 999900, 999990, 999999, 9000000, 9900000, 9990000, 9999000, 9999900, 9999990, 9999999, 90000000, 99000000, 99900000, 99990000, 99999000, 99999900, 99999990, 99999999]

Function fr_to_dn (see file Recurring_Josef, alas …

#36: $\left[\text{fr\_to\_dn}\left(\frac{19601}{5550}\right),\ \text{fr\_to\_dn}\left(\frac{1303}{5555}\right),\ \text{fr\_to\_dn}\left(\frac{1013}{99}\right)\right]$

#37: [3.53'171, 0.2'3456, 10.'23]

#38: $\text{fr\_to\_dn}\left(\frac{93612037}{83325000}\right) = 1.12345'6789$

#39: PrecisionDigits := 20

#40: NotationDigits := 20

#41: 1.12345678967896789 67

#48: $\text{fr\_to\_dn}\left(\frac{777}{8777}\right)$

#49: 0.'08852683149139797197220006836048763814515210208499
85541756864532300330409023584368235160077475219323
51190611826364361399111313660704113022672895066651
16076108009570468269340321294291899282214879799475
04625726330181155292241084653070525236413353081918
59587558391249857582317420530933120656260681326193
55770764498120086589951008317192662640993505753674

… it works much slower than Peter´s one (3900 sec for finding the correct result #49!)

Some final comments:

(1) It might be nice to check the derived formula #24 by manually calculation.

(2) It is easy to extend the function for negative numbers.

(3) I believe that this could be a fine problem for students – to analyse the procedure in general terms and then reproduce it using a CAS and finally write a function, which does the work.

(4) Last but not least many thanks to Peter for this inspiring paper. Josef

Was verbirgt sich hinter Dr. Pest?

What is hidden behind Dr Pest?

Benno Grabinger, Neustadt/Weinstraße, Germany

Abbildung 1

Der erste Griff am Morgen geht (oder sollte gehen) zur Zahnpastatube. Egal ob von Dr. Pest oder einem anderen Hersteller, das Ding, das alleine auf seinem Verschluss stehen bleibt, verdient einige Beachtung. Industriell entsteht das Gebilde aus einem Zylinder, dessen ein Ende nach der Befüllung zusammengequetscht wird, so dass sich eine Falz ausbildet (Abbildung 2).

Abbildung 2

The first grip in the morning is (or should be) to the toothpaste tube. We would like to investigate the form of the tube and then calculate its volume. It is produced as a cylinder and its bottom is pressed together to a fold after filling it. Then we find the wellknown form.

Can we compose this form using known geometric shapes?

Das Endprodukt liegt gut in der Hand, keine Ecken oder Kanten stören das glatte Erscheinungsbild.

Lässt sich eine solche Tube aus bekannten geometrischen Grundkörpern zusammensetzen?

Dazu muss eine Strecke AB, die Falz, mit den Punkten des Grundkreises der Tube verbunden werden. Da die breiteren Seiten der Tube sich eben anfühlen, sucht man im Grundkreis diejenigen beiden Punkte P und Q, die eine zur Falz parallele Tangente besitzen.

(Abbildung 3)

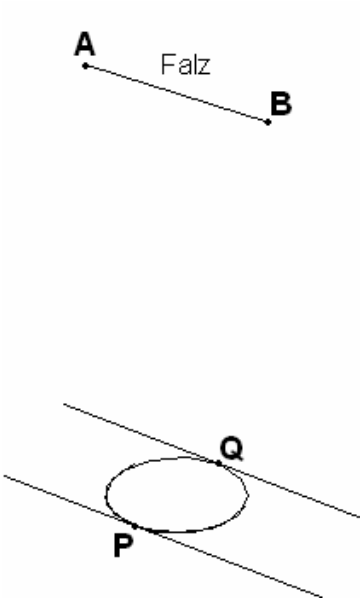


Abbildung 3

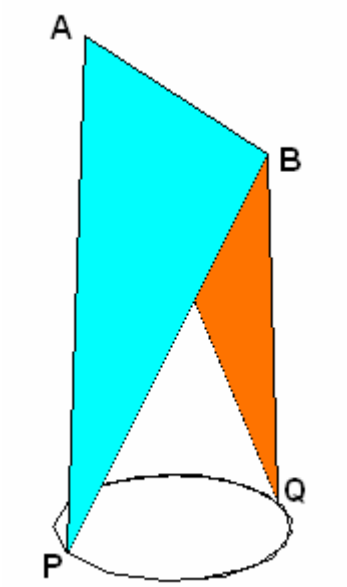


Abbildung 4

The triangles APB and AQB are parts of the surface. The broad lateral faces of the tube feel nearly plane, so fold AB must be connected with those points of the base circle which have tangents parallel to AB. Connecting the points of the two semicircles with A and B we obtain to oblique circular cones. Figure 5 shows the two cones and figure 6 shows the complete figure including its cap.

Die Dreiecke APB und AQB sind Teil der Tubenoberfläche.

Verbindet man nun noch jeden Punkt des Kreises mit A bzw. B, so entstehen zwei schiefe Halbkegel. (In der Abbildung 5 sind die zuvor gezeichneten Dreiecke wieder weggelassen, um die Kegel besser sehen zu können. Abbildung 6 zeigt die fertige Tube nebst Schraubverschluss.)

Ich füge die DERIVE-Konstruktion ein, da sie einen interessanten und wesentlichen Bestandteil dieses Beitrags darstellt.

I include the DERIVE file, because the plotting procedure of the 3D-figure is an interesting and integrating part of this contribution.
Josef

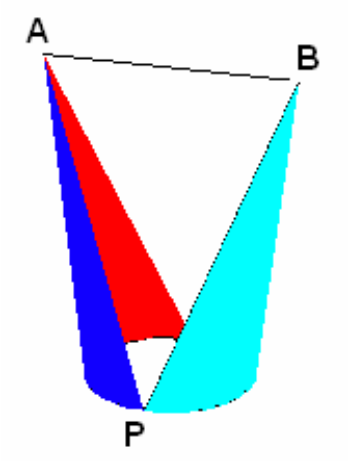


Abbildung 5

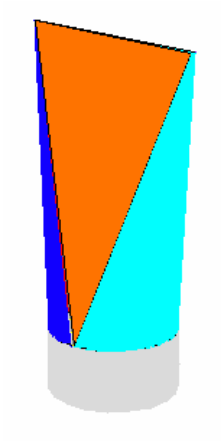


Abbildung 6

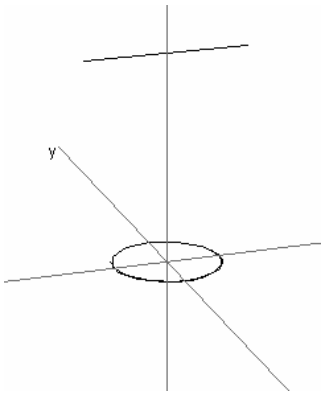
Grundkreis und Strecke AB – Base Circle and segment AB

#1: [1.5·SIN(t), 1.5·COS(t), 0]

#2: a := [−2.25, 0, 14.5]

#3: b := [2.25, 0, 14.5]

#4: [a, b]



p und q sind die Kreispunkte in denen die Tangente parallel zur Strecke.
p and q are points on the circumference with tangents parallel to segment AB.

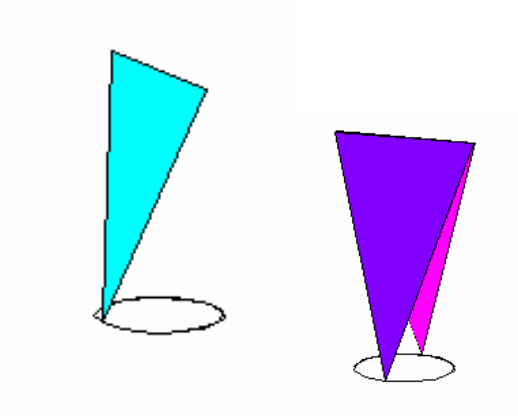
#5: p := [0, 1.5, 0]

#6: q := [0, −1.5, 0]

Die vier Dreiecke im Raum werden erzeugt.
The four triangles in space are generated.

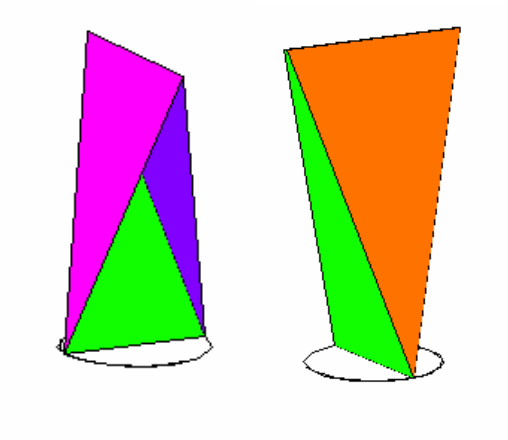
#7: $\begin{bmatrix} a & b \\ b & p \\ p & a \end{bmatrix}$

#8: $\begin{bmatrix} b & a \\ a & q \\ q & b \end{bmatrix}$



#9: $\begin{bmatrix} p & q \\ q & a \\ a & p \end{bmatrix}$

#10: $\begin{bmatrix} p & q \\ q & b \\ b & p \end{bmatrix}$



k(t) sind die Punkte auf dem Grundkreis die einerseits mit a, andererseits mit b verbunden werden.
Auf diese Weise entstehen 2 schiefe Halbkegel.

k(t) are points of the base which are connected with a and with b.
In this way two oblique cones are created.

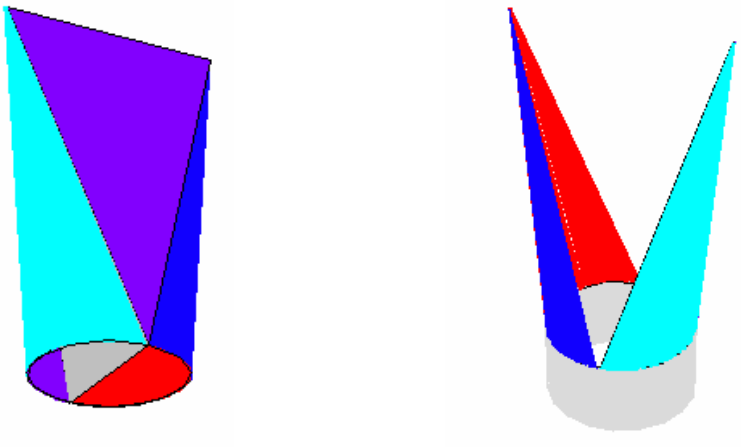
#11: $k(t) := [1.5 \cdot \cos(t), 1.5 \cdot \sin(t), 0]$

#12: $\text{VECTOR}\left([a, k(t)], t, \frac{\pi}{2}, \frac{3}{2} \cdot \pi, 0.1\right)$

#13: $\text{VECTOR}\left([b, k(t)], t, -\frac{\pi}{2}, \frac{1}{2} \cdot \pi, 0.1\right)$

Der Verschluss – the cap

#14: $deckel := [1.5 \cdot \cos(t), 1.5 \cdot \sin(t), s]$



End of the DERIVE file for plotting the figure.

Welches Volumen besitzt die Tube?
Das Volumen der beiden Halbkegel ist schnell angegeben. Ist r der Grundkreisradius und bezeichnet h die Höhe der Tube, so liefern die Halbkegel den Anteil

$$2 \cdot \frac{1}{2} \cdot \frac{1}{3} \cdot r^2 \cdot \pi \cdot h = \frac{1}{3} \cdot r^2 \cdot \pi \cdot h$$

Der Restkörper besteht aus 4 Dreiecken, d.h. es handelt sich um eine auf einer ihrer Kanten stehenden Pyramide. (Abbildung 7)

What is the volume of the tube?
It is easy to find the volume of the two half cones. With $2r$ = diameter of the base and h = height of the tube we receive

$$2 \cdot \frac{1}{2} \cdot \frac{1}{3} \cdot r^2 \cdot \pi \cdot h = \frac{1}{3} \cdot r^2 \cdot \pi \cdot h$$

The remaining body is formed by four triangles and forms a pyramid standing on one of its edges (figure 7).

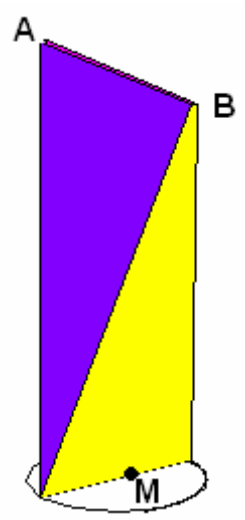


Abbildung 7

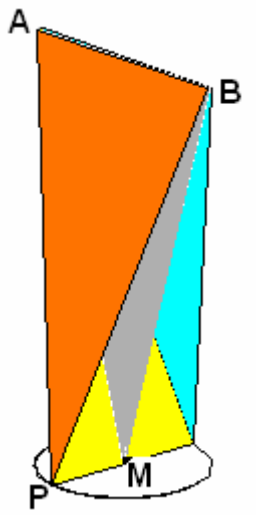


Abbildung 8

Die Berechnung desVolumens dieser Pyramide vereinfacht sich, wenn diese durch einen Schnitt in zwei gleich große Teile zerlegt wird. Die Schnittebene geht durch die Punkte A, B und M. In der Abbildung 8 wurde das vordere begrenzende Dreieck weggelassen um die Unterteilung der Pyramide in die zwei Teile zu sehen.

For calculating the volume of this pyramid we divide it into two equal parts by an intersection plane ABM (which can be seen gray coloured in figure 8). ABM is base of pyramid PMBA.

Its height is half diameter of the tube r = PM. Its area is $\frac{1}{2} \cdot f \cdot h$ with f =AB.

The volume of this solid is given by $2 \cdot \frac{1}{3} \cdot \left(\frac{1}{2} \cdot f \cdot h \right) \cdot r = \frac{1}{3} \cdot f \cdot h \cdot r$ and the total volume

$$V = \frac{1}{3} \cdot r^2 \cdot \pi \cdot h + \frac{1}{3} \cdot f \cdot h \cdot r.$$

Assuming that we fold the tube so that *f* equals half of the perimeter of the cylinder = *r* – i.e. $f = \frac{1}{2} \cdot 2 \cdot r \cdot \pi = r \cdot \pi$ – we obtain the final value for *V*:

$$V = \frac{1}{3} \cdot r^2 \cdot \pi \cdot h + \frac{1}{3} \cdot r \cdot \pi \cdot h \cdot r = \frac{2}{3} \cdot r^2 \cdot \pi \cdot h.$$

The volume of the tube is 2/3 of the volume of the starting cylinder.

The tube shown in figure 1 is given by *r* = 1.5 cm and *h* = 14.5 cm. Hence it is filled with $\frac{2}{3} \cdot 1{,}5^2 \cdot \pi \cdot 14{,}5 \approx 68$ cm^3 (ml) toothpaste. The label promises 75 ml. A similar difference can be observed at other tubes. It might be explained by a bump out of the tube (which can also be found at milk boxes). The missing ml are hidden in these bumps.

Damit ist das Volumen des Restkörpers gleich $2 \cdot \frac{1}{3} \cdot \left(\frac{1}{2} \cdot f \cdot h \right) \cdot r = \frac{1}{3} \cdot f \cdot h \cdot r$.

Für das Tubenvolumen *V* ergibt sich dann zu: $V = \frac{1}{3} \cdot r^2 \cdot \pi \cdot h + \frac{1}{3} \cdot f \cdot h \cdot r$

Nimmt man an, dass der Falzvorgang so stattfindet, dass der halbe Zylinderumfang gleich *f* wird, d.h. $f = \frac{1}{2} \cdot 2 \cdot r \cdot \pi = r \cdot \pi$, dann gilt für *V*:

$$V = \frac{1}{3} \cdot r^2 \cdot \pi \cdot h + \frac{1}{3} \cdot r \cdot \pi \cdot h \cdot r = \frac{2}{3} \cdot r^2 \cdot \pi \cdot h$$

Das Tubenvolumen beträgt damit $\frac{2}{3}$ des Volumens des ursprünglichen Zylinders der befüllt wird.

Für die Tube aus Abbildung 1 ergeben sich beim Abmessen die folgenden Werte:
r = 1,5 cm und *h* = 14,5 cm.

Die Rechnung liefert dann: $\frac{2}{3} \cdot 1{,}5^2 \cdot \pi \cdot 14{,}5 \approx 68$ cm^3(ml) Zahnpasta.

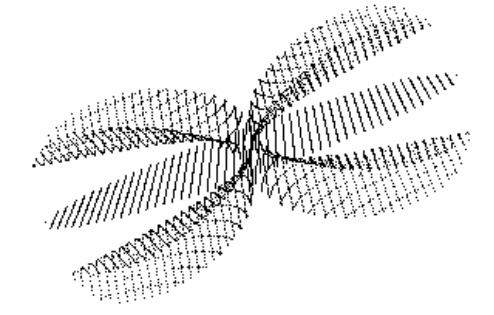
Die Tubenaufschrift verspricht dagegen 75 ml Inhalt. Dieser Effekt ist auch bei weiteren untersuchten Tuben zu beobachten. Eine mögliche Erklärung ist eine Ausbeulung der Tube wie dies auch von Milchtüten bekannt ist. In dieser Ausbeulung versteckt sich dann das fehlende Volumen.

Sämtliche Abbildungen wurden mit dem Programm DERIVE angefertigt. Für Hinweise zur Volumenberechnung bedanke ich mich bei Herrn Dr. Chr. Fahse aus Neustadt. Für die technische Unterstützung bedanke ich mich bei Dr. Klaus Wagner aus Neustadt.

All figures were produced by using DERIVE. I am very grateful for advice to the calculation of te volume (Dr Chr Fahse, Neustadt) and for rechnical support (Dr. Klaus Wagner, Neustadt).

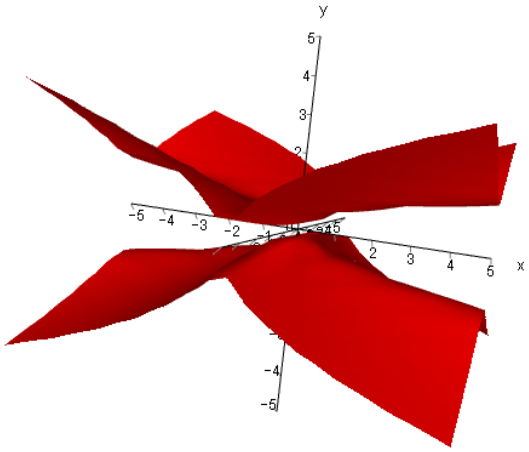
I made a typo in surface #5 from DNL#69 and received #5a:

Surface #5a: $\left(y^2+y^2\right)^3 = x^2y^2\left(z^2+1\right)$

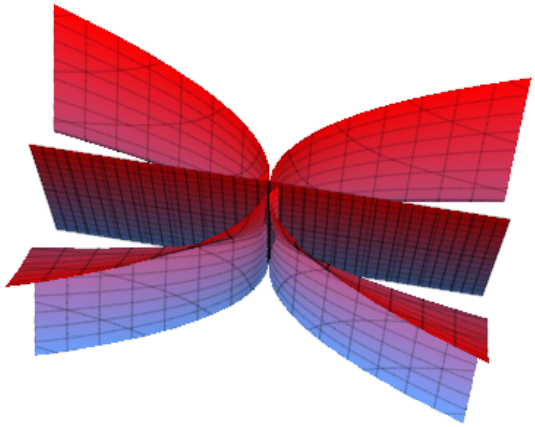


DERIVE[1]

Surfer[2]



Autograph



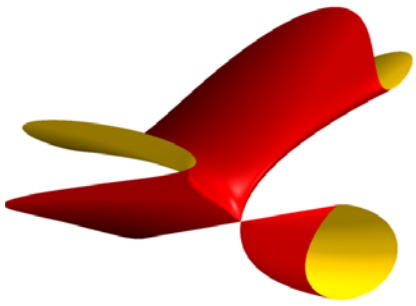
MUPAD]

[1] Produced using `implicit_peter.dfw` from DNL#64

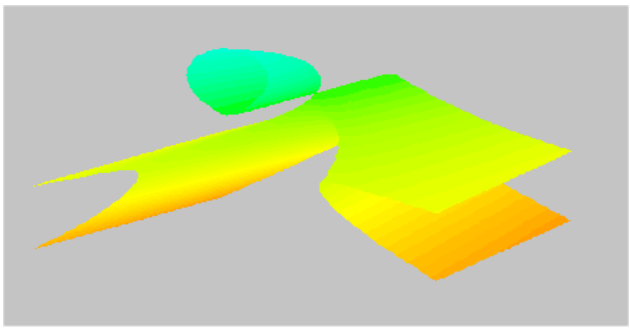
```
                  6    2  2   2          2    2    2
ImplicitDots(8·y  = x ·y ·(z  + 1) ∧ x  + y  + z  ≤ 25, [-5, -5, -5], [5, 5, 5], 0.2).
```

[2] You can download Surfer for free at **www.imaginary2008.de**. The surface looks a bit strange because the Surfer plots are bounded by a sphere.

Surface #6: $\left(y^2-z^3\right)^2 = \left(x+y^2\right)\cdot z^3$ - The UFO

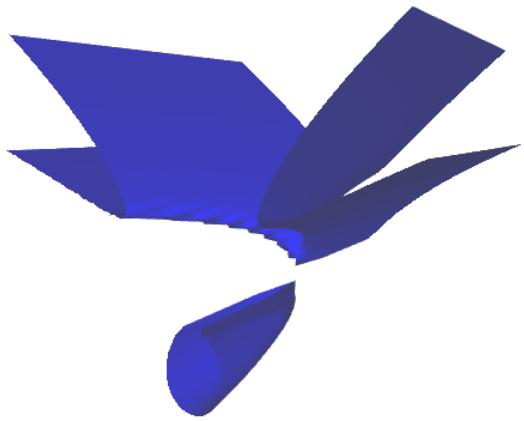


Surfer



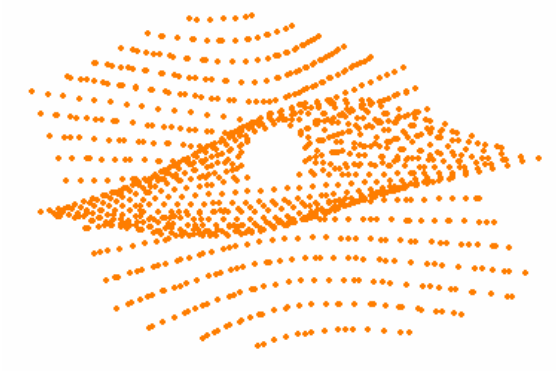
DPGraph

DERIVE[3]

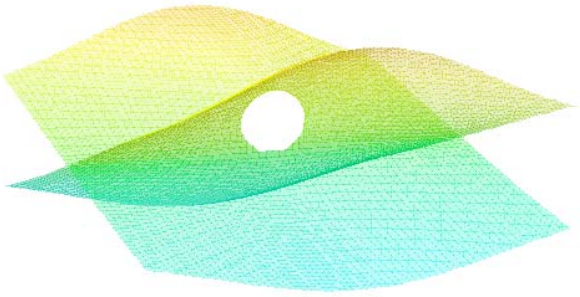


Autograph

Surface #7: $y^2 + z^3 = z^4 + x^2 z^2$

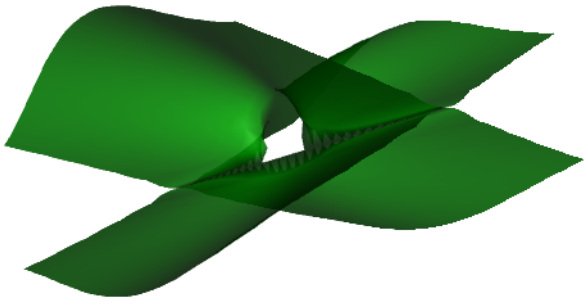


DERIVE

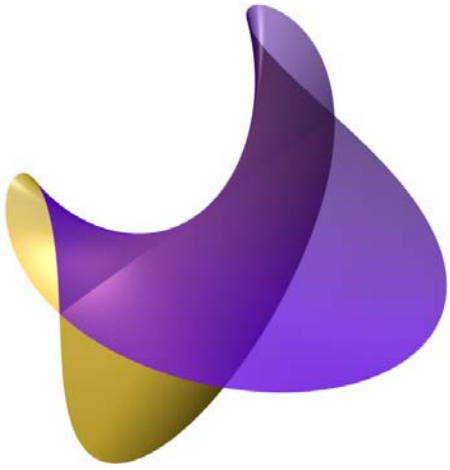


DPGraph

```
                    2    3    4    2  2
VECTOR(ContourPts_XY(y  + z  = z  + x ·z , 1, -3, 3, -3, 3, 0.2, 0.2), 1, -3, 3, 0.2)
```



Autograph



Surfer

[3] The next DERIVE plots are produced by using the plot routines from `polycontour.dfw` (presented in DNL#63). They are included in file `FAZ5.dfw`

An obstinate system of linear equations

Dear Joseph,
DUG6.mth contains 4 equations which I have been unable to get Derive to solve directly. I have managed it in the past by a different and more circuitous route and so I know the answers quoted are correct. I got Mathematica to provide them. I seem able to copy expressions from Mathematica to Derive provided I avoid Greek characters but not the other way round.
Mathematica will sometimes do things that Derive won't but Derive is quicker to use and can be about 1000 times faster than Mathematica.
Have you any idea why Derive won't cope with these 4 equations? It refused to cope when I put the 4 solutions back into the equations to check them.
DUG9 is far worse. I have been able to get Mathematica to come up with unsimplified answers but each was about a million terms long (I copied one into a Word document and did a word count.) I have been unable to simplify even a single answer and have sent it off to Wolfram to see what they say.
I would be grateful if you have anything to say.
I look forward to hearing from you, yours Arthur Lister.

#1: CaseMode := Sensitive

#2: InputMode := Word

Solve for p1p2, p1q1, p1q2, p2q1, p2q2 and q1q2

#3:

$$\begin{bmatrix} p1p2 + p1q1 + p1q2 - p1 \\ -\frac{p1p2}{Rp1p2} + p2q1 + p2q2 - p2 \\ \frac{p1q1}{Rp1q1} + \frac{p2q1}{Rp2q1} - q1q2 + q1 \\ \frac{p2q2}{Rp2q2} + \frac{p1q2}{Rp1q2} + \frac{q1q2}{Rq1} + q2 \\ -\frac{p1p2\cdot(Rp1p2 - 1)}{Rp1p2} - \frac{\left(\frac{p1q1\cdot(Rp1q1 - 1)}{Rp1q1} + \frac{p2q2\cdot(Rp2q2 - 1)}{Rp2q2}\cdot LA2\right)\cdot(LA1 - 1)}{LA1 - LA2} \\ -\frac{q1q2\cdot(Rq1q2 - 1)}{Rq1q2} - \frac{\left(\frac{p1q2\cdot(Rp1q2 - 1)}{Rp1q2} + \frac{p2q1\cdot(Rp2q1 - 1)}{Rp2q1}\cdot LB2\right)\cdot(LB1 - 1)}{LB1 - LB2} \end{bmatrix}$$

These equations have been simplified down to a standard form. The idea was to solve them and then reinsert the coefficients a5, b5, e5, c6, d6 and f6 but I wasn't able to make Derive solve this!

$$\#4:\quad \begin{bmatrix} p1p2 + p1q1 + p1q2 - p1 \\ -\dfrac{p1p2}{Rp1p2} + p2q1 + p2q2 - p2 \\ \dfrac{p1q1}{Rp1q1} + \dfrac{p2q1}{Rp2q1} - q1q2 + q1 \\ \dfrac{p2q2}{Rp2q2} + \dfrac{p1q2}{Rp1q2} + \dfrac{q1q2}{Rq1q2} + q2 \\ a5\cdot p1p2 + b5\cdot p1q1 + e5\cdot p2q2 \\ f6\cdot q1q2 + c6\cdot p1q2 + d6\cdot p2q1 \end{bmatrix}$$

DERIVE does not disclose the solution. Artur sent another equation with only four variables – together with the MATHEMATICA-solution.

#1: CaseMode := Sensitive

#2: InputMode := Word

Solve for p1p2, p1q2, p2q1 and q1q2.

#3:

$$\begin{bmatrix} \dfrac{Rp1q1\cdot(Rp2q1\cdot}{Rp1p2\cdot(Rp1q2\cdot(Rp2q2\cdot(Rq1q2\cdot q2 + } \\ \dfrac{LA1\cdot(LA2\cdot Rp1q1\cdot(Rp2q2 - 1)\cdot(Rp1p2\cdot(p2 - p2q1) + p1p2) + Rp2q2\cdot(Rp1p2\cdot(Rp1q1\cdot(p1 - p1q2) - p1 + p1p2}{LB1\cdot(LB2\cdot Rp1q2\cdot Rq1q2\cdot p2q1\cdot(Rp2q1 - 1) + Rp2q1\cdot(Rp1q2\cdot(Rq1q2\cdot(p1q2 + q1q2) - q1} \end{bmatrix}$$

Here are the *MATHEMATICA* solutions p1p2_, p1q2_, p2q1_ and q1q2_:

#4: $p1p2\_ = \dfrac{}{LA1\cdot(LA2\cdot(Rp2q2 - 1)\cdot(LB1\cdot(LB2\cdot(Rp2q1 - 1)\cdot(Rp1q1\cdot Rq1q2 - Rp1q2) + (Rp1q2 - Rq1q2)\cdot(Rp1p2\cdot Rp2q}$

#5: $p1q2\_ = -\dfrac{Rp1q2\cdot(LA1\cdot(LA2\cdot(Rp2q2 - 1)\cdot(LB1\cdot(LB2\cdot(Rp2q1 - 1)\cdot(Rp1p2\cdot p2 + Rp1q1\cdot(Rq1q2\cdot q2 + q1) + p1) + }{LA1\cdot(LA2\cdot(Rp2q2 - 1)\cdot(LB1\cdot(LB2\cdot(Rp2q1 - 1)\cdot(Rp1q1\cdot Rq1q2 - Rp1q2)}$

(1) I remove the denominators and obtain four linear equations:

```
#8:   eq1 := Rp1q1·(Rp2q1·(q1 - q1q2) + p2q1) + Rp2q1·(p1 - p1p2 - p1q2)

#9:   eq2 := Rp1p2·(Rp1q2·(Rp2q2·(Rq1q2·q2 + q1q2) + Rq1q2·(p2 - p2q1)) + Rp2q2·Rq1q2·p

#10:  eq3 := LA1·(LA2·Rp1q1·(Rp2q2 - 1)·(Rp1p2·(p2 - p2q1) + p1p2) + Rp2q2·(Rp1p2·(Rp1c
       LA2·Rp1q1·(p1p2 - Rp1p2·(Rp2q2·(p1p2 + p2 - p2q1) - p2 + p2q1)) + Rp1p2·Rp2q2·

#11:  eq4 := LB1·(LB2·Rp1q2·Rq1q2·p2q1·(Rp2q1 - 1) + Rp2q1·(Rp1q2·(Rq1q2·(p1q2 + q1q2)
       Rp2q1·(Rq1q2·(p2q1 + q1q2) - q1q2)) - Rp2q1·Rq1q2·p1q2·(Rp1q2 - 1)
```

The expressions are so large that I show only part of the screen. The file is among mth70.zip.

DERIVE is also unable to solve this equation applying SOLVE or SOLUTIONS. I tried to fool my old friend DERIVE.

(2) In order to have an easier reading I substitute for p1p2, p1q2, p2q1 and q1q2 the variab

```
#12:  eq1 := SUBST(eq1, [p1p2, p1q2, p2q1, q1q2], [w, x, y, z])

#13:  eq1 := - Rp2q1·x + Rp1q1·y - Rp2q1·(Rp1q1·z - Rp1q1·q1 - p1 + w)

#14:  eq2 := SUBST(eq2, [p1p2, p1q2, p2q1, q1q2], [w, x, y, z])

#15:  eq2 := Rp1p2·Rp2q2·Rq1q2·x - Rp1q2·(Rp1p2·Rq1q2·y - Rp1p2·Rp2q2·z - Rp1p2·Rq1q2·(

#16:  eq3 := SUBST(eq3, [p1p2, p1q2, p2q1, q1q2], [w, x, y, z])

#17:  eq3 := - x·(LA1·Rp1p2·Rp2q2·(Rp1q1 - 1) + Rp1p2·Rp2q2·(1 - Rp1q1)) - y·(LA1·LA2·R
        + LA1·(LA2·Rp1q1·(Rp2q2 - 1)·(Rp1p2·p2 + w) + Rp2q2·(Rp1p2·(Rp1q1·p1 - p1 + w)
        p2)) + Rp1p2·Rp2q2·(Rp1q1 - 1)·(w - p1)

#18:  eq4 := SUBST(eq4, [p1p2, p1q2, p2q1, q1q2], [w, x, y, z])

#19:  eq4 := x·(LB1·Rp2q1·Rq1q2·(Rp1q2 - 1) - Rp2q1·Rq1q2·(Rp1q2 - 1)) + Rp1q2·(LB2·Rq1
        LB2)·(Rq1q2 - 1))
```

(3) The system is of the form:

$$\begin{aligned} a11\,w + a12\,x + a13\,y + a14\,z + a10 &= 0 \\ a21\,w + a22\,x + a23\,y + a24\,z + a20 &= 0 \\ a31\,w + a32\,x + a33\,y + a34\,z + a30 &= 0 \\ a41\,w + a42\,x + a43\,y + a44\,z + a40 &= 0 \end{aligned}$$

and I determine all coefficients, starting with a11, a21, a31, a41 in one step:

```
#20:  col1 := d/dw [eq1, eq2, eq3, eq4]

#21:  col2 := d/dx [eq1, eq2, eq3, eq4]

#22:  col3 := d/dy [eq1, eq2, eq3, eq4]

#23:  col4 := d/dz [eq1, eq2, eq3, eq4]

#24:  const := SUBST([eq1, eq2, eq3, eq4], [w, x, y, z], [0, 0, 0, 0])
```

I try applying ROW_REDUCE

```
#25:  sys_mat := [col1, col2, col3, col4, -const]'

#26:  ROW_REDUCE(sys_mat)
```

I cancel calculation - no result after waiting a long time.
Another try:

```
#27:  sols := (SOLUTIONS([a11·w + a12·x + a13·y + a14·z + a10 = 0, a21·w + a22·x + a23·
        a30 = 0, a41·w + a42·x + a43·y + a44·z + a40 = 0], [w, x, y, z]))
```

```
#28:  [p1p2 := sols , p1q2 := sols , p2q1 := sols , q1q2 := sols ]
           1             2             3             4
```

This is the general solution. Now I resubstitute for the coefficients:

```
#29:  [a11 := col1 , a21 := col1 , a31 := col1 , a41 := col1 , a10 := const ]
           1             2             3             4             1

#30:  [a12 := col2 , a22 := col2 , a32 := col2 , a42 := col2 , a20 := const ]
           1             2             3             4             2

#31:  [a13 := col3 , a23 := col3 , a33 := col3 , a43 := col3 , a30 := const ]
           1             2             3             4             3

#32:  [a14 := col4 , a24 := col4 , a34 := col4 , a44 := col4 , a40 := const ]
           1             2             3             4             4
```

This should be the solution:

#34: p1p2 =

$$\frac{LA1\cdot(LA2\cdot(Rp2q2-1)\cdot(LB1\cdot(LB2\cdot(Rp2q1-1)\cdot(Rp1q1\cdot Rq1q2-Rp1q2)+(Rp1q2-Rq1q2)+Rp1q1\cdot Rq1q2\cdot(Rp2q2-Rp1q2)+Rp1q2\cdot(Rq1q2-Rp2q2))+Rp1p2\cdot(Rp1q2-1)\cdot(Rp1q1\cdot Rq1q2-Rp1q2)+Rp1q1\cdot(Rq1q2\cdot(Rp2q1\cdot(q1+q2)+p2)-Rp1q2\cdot(Rp2q2-Rq1q2+1)-Rq1q2\cdot(Rp2q1-Rp2q2))+Rp1q2\cdot(1-Rq1q2)\cdot(Rp2q1-1))-LB2\cdot(Rp1q1\cdot(Rp2q1\cdot(Rq1q2\cdot(p2+q1)-q1)-p2)+Rp1q2\cdot(Rp2q1\cdot Rq1q2-1)+Rp1q1\cdot(1-Rp2q1\cdot Rq1q2)\cdot(Rp1q2-Rp2q2)+Rp1q}{}$$

and so on for all unkowns … and finally:

I compare your_solutions from #4 to #7 with my solutions:

```
#37:  [q1q2 - q1q2_, p1p2 - p1p2_, p2q1 - p2q1_, p1q2 - p1q2_]

#38:  [0, 0, 0, 0]
```

The solutions are identical.
So this procedure could be a model how to tackle the more complicated system from the other file.

I treid to solve Artur´s equations using the open source CAS Maxima and succeeded with my first attempts. Calculation time was incredible short. See parts of the output on the next page.

We – Artur and I – are wondering if anybody will be able to solve the given simultaneous equations in a satisfying way using DERIVE.

Here are screen shots from the wxMaxima – output.

See first the example with four unknowns:

```
(%i4)  eq4:LB1*(LB2*Rp1q2*Rq1q2*p2q1*(Rp2q1 - 1) + Rp2q1*(Rp1q2*(Rq1q2*(p1q2 + q1q2) - q1q2)
         - Rq1q2*p1q2)) + LB2*Rp1q2*(Rq1q2*p2q1 - Rp2q1*(Rq1q2*(p2q1 + q1q2) - q1q2)) - Rp2q1*R
         - 1);

(%o4)  LB1(p2q1 Rp1q2(Rp2q1 - 1)Rq1q2 LB2 + Rp2q1(Rp1q2((q1q2 + p1q2)Rq1q2 - q1q2) - p1q2 Rq1q2)) + Rp
(p2q1 Rq1q2 - Rp2q1((q1q2 + p2q1)Rq1q2 - q1q2))LB2 - p1q2(Rp1q2 - 1)Rp2q1 Rq1q2

(%i5)  sol:solve([eq1,eq2,eq3,eq4],[p1p2, p1q2, p2q1, q1q2]);
 << Ausdruck zu lang um angezeigt zu werden! >>

(%i6)  sol [1][1];
(%o6)  p1p2 = - ( Rp2q1 ( Rp1q1 ( Rq1q2 ( Rp2q2 ( Rp1p2(p2 LA2((LB1 - 1)LB2 + LA1(1 - LB1)LB2) + q2 LA2(L
Rp1p2 Rp1q2((LB1 - 1)LB2 + LA1(1 - LB1)LB2 + LA2(- LB1 + LA1(LB1 - 1) + 1)) ) + Rp1p2
(p2 LA2(LA1(LB1 - 1)LB2 +(1 - LB1)LB2) + q2 LA2(- LB1 + LA1(LB1 - 1) + 1)) + Rp1p2 Rp1q2
(p2((LB1 - 1)LB2 + LA1(1 - LB1)LB2) + q2 LA2(LB1 + LA1(1 - LB1) - 1)) ) + q1 ( Rq1q2
(Rp1p2 Rp2q2 LA2(LA1(LB2 - LB1) - LB2 + LB1) + Rp1p2 Rp1q2(LA1(LB2 - LB1) - LB2 + LB1) + Rp1p2 LA2(LB2 + L
(Rp1p2 Rp1q2((LB1 - 1)LB2 + LA1(1 - LB1)LB2 + LA2(- LB1 + LA1(LB1 - 1) + 1)) + Rp1p2 LA2(LB2 + LA1(1 - LB2
(LB2 + LA1(LB1 - LB2) + LA2(LB1 + LA1(1 - LB1) - 1) - LB1) + Rp1p2 LA2(- LB2 + LA1(LB2 - 1) + 1) ) + p1 Rq1q2
```

……

This is the second one – using the form with generalized coefficients (expression #4 from the above DERIVE file):

```
(%i5)  eq5:a5*p1p2+b5*p1q1+e5*p2q2=0;
(%o5)  e5 p2q2 + b5 p1q1 + a5 p1p2 = 0

(%i6)  eq6:f6*q1q2+c6*p1q2+d6*p2q1=0;
(%o6)  f6 q1q2 + d6 p2q1 + c6 p1q2 = 0

(%i7)  sol:solve([eq1,eq2,eq3,eq4,eq5,eq6],[p1p2,p1q1,p1q2,p2q1,p2q2,q1q2]);
(%o7)  [ [ p1p2 = ( Rp1p2 ( Rp2q1 ( Rp1q1
(q1((e5 f6 Rp2q2 + b5 f6 Rp1q2)Rq1q2 +(b5 d6 - c6 e5)Rp1q2 Rp2q2) +(b5 d6 - c6 e5) q2 Rp1q2 Rp2q2
p2(Rp1q1((e5 f6 Rp2q2 + b5 f6 Rp1q2)Rq1q2 - c6 e5 Rp1q2 Rp2q2) + Rp2q1(Rp1q1(d6 e5 Rp2q2 + b5 d6
f6 q2 Rp1q1 Rp1q2 Rp2q2 Rq1q2 ) + p1 Rp1p2
(Rp2q1(Rp1q1(b5 d6 Rp2q2 + b5 c6 Rp1q2)Rq1q2 + e5 f6 Rp2q2 Rq1q2 - c6 e5 Rp1q2 Rp2q2) + Rp1q1(b5
(Rp1p2 (Rp2q1((e5 f6 Rp2q2 + a5 f6 Rp1q2)Rq1q2 + Rp1q1((b5 d6 - a5 d6)Rp2q2 +(b5 c6 - a5 c6)Rp1
+ Rp1q1((b5 f6 - a5 f6)Rp2q2 Rq1q2 +(a5 c6 - b5 c6)Rp1q2 Rp2q2)) + Rp1q1((- e5 f6 Rp2q2 - b5 f6 R
(Rp1q1(- d6 e5 Rp2q2 - b5 d6 Rp1q2)Rq1q2 + d6 e5 Rp1q2 Rp2q2)) , p1q1 = - ( Rp1p2 ( Rp1q1 Rp2q1
(q1((e5 f6 Rp2q2 + a5 f6 Rp1q2)Rq1q2 +(a5 d6 - c6 e5)Rp1q2 Rp2q2) +(a5 d6 - c6 e5) q2 Rp1q2 Rp2q2
(Rp1q1((e5 f6 Rp2q2 + a5 f6 Rp1q2)Rq1q2 - c6 e5 Rp1q2 Rp2q2) + Rp1q1 Rp2q1(d6 e5 Rp2q2 + a5 d6 Rp1
Rq1q2 ) + p1 ( Rp1p2(Rp1q1(a5 f6 Rp2q2 Rq1q2 - a5 c6 Rp1q2 Rp2q2) + Rp1q1 Rp2q1(a5 d6 Rp2q2 + a5 c
(e5 f6 Rp2q2 Rq1q2 - c6 e5 Rp1q2 Rp2q2) + d6 e5 Rp1q1 Rp2q1 Rp2q2 Rq1q2 ) + Rp1q1 Rp2q1(d6 e5 q2 Rp
+ e5 f6 q2 Rp1q1 Rp1q2 Rp2q2 Rq1q2 ) / ( Rp1p2 ( Rp2q1
```

Titbits from Algebra and Number Theory (35)
or Yet Another Treatise on RSA

(c) Johann Wiesenbauer, Vienna University of Technology

Hm, RSA again - not exactly very imaginative you might think. In fact, a huge amount of articles are referring to it, as a simple Google search proves. Isn't everything said and done, when it comes to RSA? Well, I will come back to this question before long. Let me point out first that I'm focussing here mostly on the mathematical background of RSA and the implementation of certain algorithms rather than the cryptographical environment. In particular, I will assume here a basic knowledge what RSA is all about (if needed, an excellent reference for cryptography is

http://www.cacr.math.uwaterloo.ca/hac/).

Basically, if n is a natural number that is a product of two "big" primes p and q, and x is a message coded as an element of the residue class ring **Z**n, i.e. $x \in \{0,1,2,..,n-1\}$, then the encryption E is a bijective mapping from **Z**n onto **Z**n of the form x -> x^e mod n for some fixed natural number e. Now it's a simple algebraic exercise to prove that two exponents e1 and e2 induce the same mapping if and only if

$$e1=e2 \bmod \lambda(n), \text{ where } \lambda(n)=\operatorname{lcm}(p-1,q-1).$$

In particular, we may assume w.l.o.g. that $e < \lambda(n)$. This innocuous looking assertion has the important implication that the mapping induced by the exponent e1 is the identity (represented by the exponent e2=1), if and only if $e1 = 1 \bmod \lambda(n)$, which in turn implies that the mappings belonging to the exponents e and d are inverse to each other if and only if

$$de = 1 \bmod \lambda(n).$$

For some of you this might be the first surprise as you might have been used to the condition

$$de = 1 \bmod \phi(n), \text{ where } \phi(n)=(p-1)(q-1).$$

Furthermore, in the "classical" RSA the exponents e and d were supposed to be $< \phi(n)$ rather than $< \lambda(n)$. Well, as we know, it works too, because $\lambda(n)$ is a divisor of $\phi(n)$, but the exponent d is usually not best possible as to its size. Let's do a small example, just to see what can happen. We use n= 15251=101*151 for this, and compute d for e=301, using the classical RSA-scheme. Since $\lambda(n)= \operatorname{lcm}(100,150)=300$ and $301=1 \bmod 300$, we know that the mapping belonging to exponent e is the identity mapping. Hence, we would d expect to be 1. Unfortunately this not true.

```
#1:   d := INVERSE_MOD(301, EULER_PHI(15251))

#2:                                          d := 14701
```

The number d looks quite inconspicuous, doesn't it? Nevertheless, the corresponding mapping is actually the identity!!!

```
                                  14701
#3:                 EVERY(MOD(x      , 15251) = x, x, 0, 15250) = true
```

Admittedly, this is the worst case, and usually d is only a few bits larger than necessary, but all the same is clearly better, as it is optimal, when it comes to size, and its computation is still very simple. Sadly enough, even though, still rather few authors use λ(n), one of them is

http://www.staff.uni-mainz.de/pommeren/Kryptologie/Asymmetrisch/

(Sorry, as it is only for people with a basic command of German!)

This suggests the following question: What is general relationship between ϕ(n) and λ(n) for any positive integer n? Well, ϕ(n) is the number of elements of Zn* = {a | 1<=a<=n and gcd(a,n)=1}, and ϕ is called Euler's ϕ-function, after its first investigator, whereas λ(n) is the smallest positive integer k such that a^k= 1 mod n for all a in Zn*. In short, ϕ(n) is the order and λ(n) the so-called exponent of Zn*. ϕ(n) is the library function euler_phi(n), which we already used above, and λ(n) , which is usually called after Carmichael, can be implemented as follows.

```
      carmichael_λ(n) :=
        Prog
#4:       If MOD(n, 8) = 0
             n :/ 2
          LCM(VECTOR(f_↓1^f_↓2·(1 - 1/f_↓1), f_, FACTORS(n)))

#5:                                 carmichael_λ(15251) = 300
```

This raises the following interesting question: Does our RSA-scheme above work for an arbitrary n using this definition of λ(n) ? Unfortunately, the answer is no in general, though it does work if n is squarefree, i.e. the product of different prime numbers, which is at least a slight generalization of the original RSA, where n is the product of 2 different primes. On the other hand, even if n is not squarefree, it still works for "most" x, or more precisely it works, unless gcd(x,n/gcd(x,n)) > 1.

Let's test this for n=45, hence λ(n)=12, and e=d=5.

```
                      5 5
#6:    SELECT(MOD((x ) , 45) ≠ x, x, 0, 44) = [3, 6, 12, 15, 21, 24, 30, 33, 39, 42]
```

$$\#7:\quad \text{SELECT}\left(\text{GCD}\left(x, \frac{45}{\text{GCD}(x, 45)}\right) > 1, x, 0, 44\right) = [3, 6, 12, 15, 21, 24, 30, 33,$$

```
        39, 42]
```

However, as for all these "generalizations" of RSA, we are not so much concerned about those exceptional values of x - for an n with "big" prime factors the probability to encounter any of those is virtually zero -, but about the fact that the factorization problem becomes far easier, if n splits up into many primes! And anyone, who can factor n, can compute λ(n) easily (see the algorithm #4 above!) and a fortiori the secret key d, which is the inverse of e mod λ(n).

Let's turn to a realistic RSA-implementation now. To this end I will provide a routine setup(b,e) that sets a number of global variables, namely

- the primes p,q with p ≠ q of about the size 2^(b/2)
- their product n = p*q, whose binary representation has exactly b bits, where b=1024 by default
- the public exponent e, where e =2^16+1 by default
- the corresponding private exponent d = e^(-1) mod λ(n)
- the values dp and dq, which are d mod (p-1) and d mod (q-1) by definition, respectively, and will be used for decryption later on

```
      setup(b := 1024, e0 := 2^16 + 1) :=
        Prog
          e := e0
          Loop
            p := NEXT_PRIME(RANDOM(2^FLOOR(b, 2)))
            If GCD(e, p - 1) = 1 exit
          Loop
#8:         n := 2^(b - 1) + RANDOM(2^(b - 1))
            q := NEXT_PRIME(FLOOR(n, p))
            n := p·q
            d := INVERSE_MOD(e, LCM(p - 1, q - 1))
            dp := MOD(d, p - 1)
            dq := MOD(d, q - 1)
            If p ≠ q ∧ FLOOR(LOG(n, 2)) = b - 1 ∧ NUMBER?(d) exit
```

The following call of setup() that provides all global variables above for the default values of b=1024 and e=2^16+1 takes usually only tenths of a second.

```
#9:                                    setup() = true
```

Now, let's assume that the message m is given as string of symbols from the extended ASCII-character set. In a first step, we concatenate all 8-bit representations of the characters of m, getting thus a single binary string s. Next we split up an expanded version of s into k binary blocks of the fixed length l1 for some l1 < b. (We will choose the maximal possible value l1=b-1 in our implementation below.) In order to get that expanded version of s mentioned above, we first adjoin the bit 1 at the rightmost position, and thereafter the smallest number of zeros such that the length of the new s becomes divisible by l (zeropadding). Now after deleting leading zeros these k binary blocks of length l are interpreted as binary representations of nonnegative integers x1,x2,…,xk all below n, which are encrypted then into y1,y2,..,y2, using the RSA-scheme at issue. These y-values are usually converted into binary blocks of some fixed length l2, where l2>=b, and concatenated. For didactic reasons I decided to use the explicit vectors (x1,x2,..,xk) and (y1,x2,…,yk) of decimal values in the following though. In the following, I wrote separate routines for the source encoding/decoding and the actual encrypting/decrypting, respectively. Furthermore I give two versions of decrypt(c), where the faster second one makes use of the Chinese Remainder Theorem (CRT) and the global variables dp and dq, we provided in the setup above.

```
      encoding(t, b_, c_ := 1, s_ := 0) :=
        Prog
          b_ := FLOOR(LOG(n, 2))
          t := NAME_TO_CODES(t)
          Loop
            If t = [] exit
            c_ :+ 8
            s_ := SHIFT(s_, 8) + FIRST(t)
            t := REST(t)
#10:        s_ := 2·s_ + 1
          s_ := SHIFT(s_, MOD(-c_, b_))
          t := []
          c_ := SHIFT(1, b_)
          Loop
            If s_ = 0
               RETURN t
            t := ADJOIN(MOD(s_, c_), t)
            s_ := SHIFT(s_, -b_)

                                    e
#11:  encrypt(t) := MAP_LIST(MOD(x , n), x, encoding(t))

      decoding(v, b_, s_ := 0, t_ := "") :=
        Prog
          b_ := FLOOR(LOG(n, 2))
          Loop
            If v = [] exit
            s_ := SHIFT(s_, b_) + FIRST(v)
            v := REST(v)
          Loop
#12:        b_ := MOD(s_, 2)
            s_ := SHIFT(s_, -1)
            If b_ = 1 exit
          Loop
            If s_ = 0
               RETURN t_
            t_ := ADJOIN(CODES_TO_NAME(MOD(s_, 256)), t_)
            s_ := SHIFT(s_, -8)
```

```
                                                  d
#13:  decrypt(c) := decoding(MAP_LIST(MOD(y , n), y, c))

                                                 ⎡      dp          dq     ⎤
#14:  decrypt(c) := decoding(MAP_LIST(CRT(⎣MOD(y , p), MOD(y , q)⎦, [p, q]), y, c))
```

Let's check it using a famous piece of text from a speech of Abraham Lincoln ("Gettysburg address") consisting of 1494 characters. I also used here for the very first time a timing function that will give you the run time in seconds on the screen itself rather than on the right side of the status bar. (See the examples below as to its use.) In particular you can see that the total time for both encrypting and decrypting was only 0.48s on my PC and most of this time, namely 0.41s, was spent on decrypting a vector with 12 numbers each with about 308-309 digits. Hence, the RSA decryption took about 0.034s per block.

```
      timing(t, r) :=
        Prog
          t := APPEND(STRING(RANDOM(0) - t), "s")
#15:      Loop
            If DIM(t) < 4
               t := ADJOIN("0", t)
               RETURN INSERT(".", t, -3)

#16:  t := November 19, 1863 -- Four score and seven years ago, our fathers brought
        forth upon this continent a new nation: conceived in liberty, and dedicated
        to the proposition that all men are created equal. Now we are engaged in a
        great civil war...testing whether that nation, or any nation so conceived
        and so dedicated... can long endure. We are met on a great battlefield of
        that war. We have come to dedicate a portion of that field as a final
        resting place for those who here gave their lives that that nation might
        live. It is altogether fitting and proper that we should do this. But, in a
        larger sense, we cannot dedicate...we cannot consecrate... we cannot hallow
        this ground. The brave men, living and dead, who struggled here have
        consecrated it, far above our poor power to add or detract. The world will
        little note, nor long remember, what we say here, but it can never forget
        what they did here.   It is for us the living, rather, to be dedicated here
        to the unfinished work which they who fought here have thus far so nobly
        advanced. It is rather for us to be here dedicated to the great task
        remaining before us...that from these honored dead we take increased
        devotion to that cause for which they gave the last full measure of
        devotion... that we here highly resolve that these dead shall not have died
        in vain...that this nation, under God, shall have a new birth of freedom...
        and that government of the people...by the people...for the people... shall
        not perish from the earth.

#17:                          [DIM(t), DIM(n)] = [1494, 309]

#18:         [DIM(encoding(t)), timing(RANDOM(0), encoding(t))] = [12, 0.07s]

#19:          [DIM(encrypt(t)), timing(RANDOM(0), encrypt(t))] = [12, 0.08s]

#20:                                  DIM(n) = 309

#21:                       SOLVE(decrypt(encrypt(t)) = t) = true

#22:                  timing(RANDOM(0), decrypt(encrypt(t))) = 0.48s
```

Now let's discuss some security aspects of RSA. In the first place, the primality tests to find the primes p and q in our routine setup() are only probabilistic, i.e. there is a tiny chance that one of them is actually composite. This raises the interesting question: Is it possible that our RSA-encryption and decryption routines above still work, if one of the numbers p and q is not prime or even both? Surprisingly enough, the answer is yes.

To be more precise, there are actually 3 properties we really need regarding the positive integers p and q:

1) a^p = a mod p for all integers a
2) a^p = a mod q for all integers a
3) gcd(p,q) = 1

An integer n>1 that fulfils a^n=a for all integers a without being a prime is called a Carmichael number. Unfortunately, those numbers exist and there are even infinitely many of them (http://www.math.dartmouth.edu/~carlp/PDF/paper95.pdf) Due to a theorem by Korselt, they can be characterized as squarefree integers n > 1 such p - 1 divides n - 1 for any prime divisor p of n. It can be easily proven then that a Carmichael number must always be odd and has at least 3 different prime factors. Furthermore, any n of the form n=(6k+1)(12k+1)(18k+1) for some positive integer k, where 6k+1,12k+1 and 18k+1 are prime, fullfills the conditions of Korselt's theorem.

We use the latter fact to construct Carmichael numbers greater or equal to a given number s, which are used then in an RSA-emulation for out text t above that works indeed. Needless to say that this is a horror scenario in practice, as the factoring problem for n and hence the computation of λ(n) and d becomes so much easier for an attacker. Note also the relatively small value of λ(n), which forces d to be rather small, too!

```
      Carmichael?(n) :=
        Prog
          If PRIME?(n) ∨ (n − 1)·MOD(n, 2) = 0
#23:         RETURN false
          If SOME(e_ > 1, e_, (FACTORS(n)) COL 2)
             RETURN false
          EVERY(MOD(n − 1, p_ − 1) = 0, p_, (FACTORS(n)) COL 1)

#24:  ITERATE(IF(PRIME(6·k + 1) ∧ PRIME(12·k + 1) ∧ PRIME(18·k + 1), k, k + 1),
        k, 1000) = 1025
```

```
#25:  NSOLVE((6·k + 1)·(12·k + 1)·(18·k + 1) = 1000, k) = (k = −0.56104944 −

        0.7932487967·i ∨ k = −0.56104944 + 0.7932487967·i ∨ k = 0.8165433244)

#26:   NSOLVE((6·k + 1)·(12·k + 1)·(18·k + 1) = 1000, k, Real) = (k = 0.8165433244)

      find_CN(s, k_) :=
        Prog
          k_ := CEILING(RHS(NSOLVE((6·k + 1)·(12·k + 1)·(18·k + 1) = s, k, Real)))
#27:      Loop
            If PRIME(6·k_ + 1) ∧ PRIME(12·k_ + 1) ∧ PRIME(18·k_ + 1)
               RETURN (6·k_ + 1)·(12·k_ + 1)·(18·k_ + 1)
            k_ :+ 1

      ⎡                        10                              10
#28:  ⎣p := find_CN(RANDOM(10   )), q := find_CN(RANDOM(10   )), GCD(p, q), n := p·q,

        carmichael_λ(n)⎦

#29:       [p := 1299963601, q := 9624742921, 1, n := 12511815466282418521, 140400]

      ⎡      16
#30:  ⎣e := 2   + 1, d := INVERSE_MOD(e, 140400), dp := MOD(d, p − 1),

        dq := MOD(d, q − 1)⎦

#31:                   [e := 65537, d := 68273, dp := 68273, dq := 68273]

#32:           [DIM(encrypt(t)), timing(RANDOM(0), encrypt(t))] = [190, 0.08s]

#33:                         SOLVE(decrypt(encrypt(t)) = t) = true

#34:                   timing(RANDOM(0), decrypt(encrypt(t))) = 0.44s

#35:                      FACTOR(n) = 601·1171·1201·1801·2341·3511
```

We have already mentioned several times that anyone who can factor n, can also compute the secrret exponent d and hence decrypt any messages that use this RSA-scheme. Interestingly enough, the converse is also true: Anyone who knows d, can also factor n. In particular, two users should never share the same modulus n.

The idea of the so-called common modulus attack is very simple. Take any x with 0<x<n and check if gcd(x,n)=1. If the unlikely case that x and n aren't coprime, this gcd(x,n) is one of the prime divisors of n and we are finished. Otherwise we have x^(ed) =x mod n and since x is invertible also x^(ed-1)=1 mod n. Now looking at the congruence y^2=1 mod n, on the one hand it has the obvious solutions y= ±1 as well as two other solutions ±u, where u is a solution of the system of congruences x=1 mod p, y=-1 mod q using the CRT. On the other hand, we know that y=x^((ed-1)/2) mod is also a solution. If y=±u mod n, this obviously implies that y±1 is either divisible by p or q, but not both, and hence n can be factored into n=gcd(y+1,n)gcd(y-1,n). If y=-1 mod n, we were unlucky and must choose another x. The same goes for the case, where y=1 and (ed-1)/2 is odd. However, if y=1 mod and (ed-1)/2 is still even, on can replace y by y=x^((ed-1)/4) mod n and repeat as above. As you can see the chances of a success for each try are 50% and the number of tries is k, if 2^k is the biggest power of 2 that divides ed-1.

Okay, let's try this out for a new standard setup of RSA. (The number in the textbox is the number of random values of x that were needed for a success. It is often 1 and rarely bigger than 2.)

```
#36:                                  setup() = true

      findfactor(n, e, d, c_ := 0, k_, x_, y_) :=
        Loop
          x_ := RANDOM(n)
          y_ := GCD(x_, n)
          If y_ > 1
             [DISPLAY(c_), RETURN y_]
          c_ :+ 1
#37:      k_ := e·d − 1
          Loop
            k_ :/ 2
            y_ := MODS(x_^k_, n)
            If y_ ≠ ±1
               [DISPLAY(c_), RETURN GCD(y_ − 1, n)]
            If y_ = −1 ∨ (y_ = 1 ∧ ODD?(k_)) exit

1

#38:  findfactor(n, e, d) =
        1813178269192617663103865449435130243024150198769632074073503976900611164996~
        3854618333374751591222286492608664190901804009784939978589851055114333847621~
        389

2

#39:  findfactor(n, e, d) =
        6556616639787462592763064194321098871915177239834438525857041211204202630274~
        2297981875002873189458022619691366502304442132361484618947489970815071021403~
        29
```

Okay, as we saw factoring n and finding the secret exponent d are computationally equivalent. On the other hand, we don't know whether the problem of RSA-decryption is as hard as the problem of factoring n (or finding d for that matter). There is a variant of RSA, the so-called Rabin variant, where we can prove this though. For this variant we use the exponent e=2, which is actually forbidden in RSA since the mapping x->x^2 mod n is no longer a permutation of **Z**n. As a rule, if y is a square mod n, the equation y=x^2 mod n has got 4 solutions. They can be found by finding first the solutions ±u mod p and ±v mod q, respectively, and then solving x=±u mod p, x = ±v mod q for all 4 possible choices of signs, again by using the CRT. Usually only one of the 4 solutions makes sense when applying the source decoding, which enforces its uniqueness.

Let's conclude with a demonstration of the Rabin variant using again an RSA-setup, setting e:=2 thereafter and simply ignoring the values referring to d.

```
#40:                          [setup(), e := 2] = [true, e := 2]

#41:  c := encrypt(Let's call it a day!)
```

```
#42:  c :=
        [420181115085122636734810678510068725656273958426574586581465678044090692724~
        100166454575073001866676495022953508567831870357278701261247112778103296272O~
        580919484528440740957603328164006986281330036770376072516237788829756785870S~
        948311248595246088925438878870256999734458041809910553869370055842811550527O~
        188]

#43:  u := SQUARE_ROOT(FIRST(c), p)

#44:  u :=
        588507420652404793492163254968375364657989733281623938363192987114962389645 3~
        614971660770269776583128798800580155410817389272963106843568275271062113635 5~
        97

#45:  v := SQUARE_ROOT(FIRST(c), q)

#46:  v :=
        428253393980448213874394934468733284760109362805496826964664614350106405504 2~
        867821717973223363590534050896168611511899007737186187568363751017153292935 2~
        88

#47:                  STRING?(decoding([CRT([u, v], [p, q])])) = true

#48:  decoding([CRT([u, v], [p, q])]) =
        ë–ÅÓÉøVmg=:∏Š_Ê1(mpöÅÑJoξíbË!åù AËâÛ6ÒEZvóιJs vø1(D¿aÕ↓$ü •JÅŒ{αøÎΛF·ɣ√~
        äKÁvšw@nÈM\[ ]FÖJvξ∪ñ ωÓñ–δ•φ3 6–Γ κ µ<ð[eÑLΦøe√»√ã

#49:                  STRING?(decoding([CRT([–u, v], [p, q])])) = true

#50:               decoding([CRT([–u, v], [p, q])]) = Let's call it a day!

#51:                  STRING?(decoding([CRT([u, –v], [p, q])])) = false

#52:                 STRING?(decoding([CRT([–u, –v], [p, q])])) = false
```

Okay, for the Rabin variant decrypting is as hard as factoring n. But how hard is factoring n? As you might conclude from the last plaintext, I will postpone a treatment of the question until one of the future issues of this series.

(As always, comments and questions referring to this article to
j.wiesenbauer @tuwien.ac.at)

Latest News:

We will have a DERIVE & CAS-TI User Group Meeting in the frame of TIME 2008 in Buffelspoort Ressort, South Africa.