THE DERIVE - NEWSLETTER #77

ISSN 1990-7079

THE BULLETIN OF THE

DERIVE

USER GROUP

+ CAS-TI

Contents:

1 Letter of the Editor

2 Editorial - Preview

3 User Forum

Roland Schröder
5 The Birthday of the King

Heinrich Ludwig
8 A Modern & Efficient Runge-Kutta-Method

Josef Böhm
23 Runge-Kutta unveiled

Wolfgang Alvermann
35 The Octahedron of Horror

40 TIME 2010 – Keynotes, Lectures and Workshops

March 2010

Interesting and recommended websites:

www.atm.org.uk/mti/216/data-from-web.html
A rich resource for data (UK Census, Australian Census and many many others)

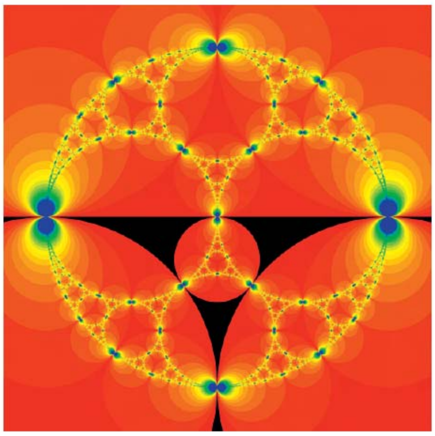
Websites for Dynamic Systems (Prof. Ossimitz, University of Klagenfurt)

wwwu.uni-klu.ac.at/gossimit/linklist.php (in English)

wwwu.uni-klu.ac.at/gossimit/sdyn.sdlv.htm Online Kurs zur Systemdynamik

wwwu.uni-klu.ac.at/gossimit/sw/sdsw.htm Software for Dynamic Systems

Do you like graphics? Do you want to learn about hyperbolic Geometry? Do you like fractals? Then go to:



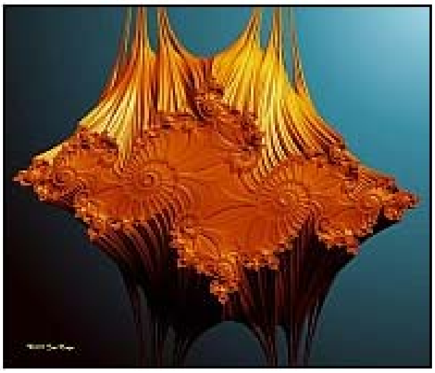
www.spektrumverlag.de/sixcms/media.php/767/master2007-5-1-small2.pdf

www-m10.ma.tum.de/bin/view/MatheVital/IndrasPearls/
(Interactive Materials – Cinderella)

www.pdfone.com/keyword/the-world-of-hyperbolic-geometry.html
Download the pdf-book (and many others)

Recommended Reading:
D. Mumford, C. Series, D. Wright: *Indra´s Pearls*, Cambridge University Press, 2006

Wonderful Fractals can be found at:



http://users.telenet.be/jos.leys/
Fractals by the famous Fractal´s artist Jos Leys (Galleries of Ultra-fractal images and animations)

https://www.fractalus.com/

http://www.ultrafractal.com A fractal generating program

http://infinitezoom.com A fractal gallery

And finally download a pdf-book for Linear Algebra from
http://archives.math.utk.edu/tutorials/linear.algebra.krm/

Please share your favourite websites and readings with us, Josef.

Dear DUG Members,

Springtime has come to Austria and it is time to publish DNL#77. This newsletter is the first one in DUG´s **twentieth** (**20th**) year of existence. I hope that we will celibrate this – remarkable for a journal like this – jubilee at TIME 2010 in Malaga. We will hold the DUG general assembly in the frame of our conference and I am very happy that some of the first members announced attending TIME 2010. Our vice president – Bärbel Barzel – will be there giving a keynote. Member #1 – Bernhard Kutzler – will give two lectures and there will be many others. We are very proud to announce more than 90 lectures and workshops for TIME 2010. Have a look on pages 40-42 and browse the accepted abstracts on the conference website.

Please note the recommended websites. I checked them all and they all worked. In my opinion they are really worth visiting. *Indra´s Pearls* (*The Vision of Felix Klein*) is a great adventure.

I´d like to inform you that TI-Nspire CAS version 2 is out. Among other changes the most remarkable novelty is the availability of colors for plotting and having a kind of text and dialogue boxes for programming and within the notes. You can see an example on page 32. Updating is an easy process and works fine. Heinrich Ludwig´s improvement of Runge-Kutta method is a true gem. His contribution inspired me to deal with Runge-Kutta, too – but only with the "classic" method – working with Voyage and NspireCAS.

Wolfgang Alvermann provided the "Octahedron of Horror". This is the translation of a headline of an article in the German journal *"Der Spiegel"* which read there "Das Oktaeder des Grauens". The presented problem was part of a (central) end examination which appeared to be too difficult for the majority of the students. You are invited to making your own opinion and downloading the *Spiegel* article (URL is given on page 39).

Best regards until summer

As announced earlier we will put all proceedings of ACDCA- and DERIVE Conferences on the RFDZ-website http://rfdz.ph-noe.ac.at/index.php?id=133. We are starting with TIME 2008.

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

Attend TIME2010 in Malaga, Spain
http://www.time2010.uma.es

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: ++43(0)6604070480
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: June 2010
Deadline 15 May 2010

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
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
Practise Working with times
Huffman-Code with DERIVE and TI-CAS, J. Böhm, AUT
Tutorials for the NSpireCAS, G. Herweyers, BEL
Some Projects with Students, R. Schröder, GER
Dirac Algebra, Clifford Algebra, D. R. Lunsford, USA

and others

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

Danny Ross Lunsford **antimatter33@yahoo.com**

Hello Prof. Boehm,

Are you going to make PDFs of the newsletters in the range 19-current? These are a valuable historical resource for serious DERIVE users.

Even though TI has stopped selling it, I use it constantly.

-ross in atlanta

DNL:

Dear Derivian,

many thanks for your kind and encouraging words. Yes, I intend to proceed revising and converting to pdf-files the DNLs which have not been published on the web so far. The next one is ready and I hope that it will appear on our website within the next week.

I also will update regularly an index for the revised issues and the other ones as well.

The index can be downloaded from the DUG-site.

Best regards

Josef

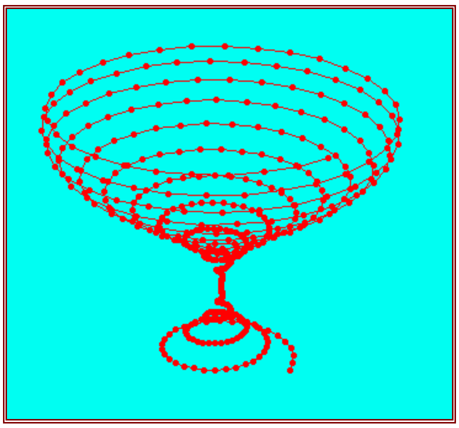
Danny Ross Lunsford **antimatter33@yahoo.com**

Sehr gut :)

Recently I was working with my old favorite APL2 - I have a half-hearted project to implement APL2 in Derive - some things are done - but I came up against the notion of an empty array with structure, e.g. 3 0 2 reshape empty creates an empty array with dimensions 3 0 2. Although Derive's array engine is fully capable of handling heterogeneous arrays of any dimension and includes the notion of an empty array [], I could not find a natural way to implement structured empty arrays (having one dimension = 0).

In the process of this, it struck my how much more felicitous is Derive than APL2 in actual use. e.g. matrix multiplication in APL2 is very ungainly and always requires an explicit operator +.x to be placed between factors. I work in the Dirac algebra of spacetime and with higher Clifford algebras, and this seemingly minor issue becomes a huge drain on one's energy because it is almost impossible to enter expressions of any complexity without syntax errors. In Derive one proceeds in a way almost like on paper! I was thinking of publishing my Dirac algebra work in the newsletter, and later a more general article on Clifford algebras abstractly.

-ross in atlanta



Phil Ignacio pdy1971@gmail.com

Is there a Math wizard that help with this please?
Thanks
Phil

This is my problem:
Given Y_1 and its first derivative as $t^2 + 5t$ and $2t + 5$ and Y_2 and its first derivative $t^2 – 5t$ and $2t – 5$.

The Wronskian is $$W = \begin{vmatrix} Y_1 & Y_2 \\ Y_1' & Y_2' \end{vmatrix} = 10t^2.$$

Now assume we **<DO NOT>** know Y_2, however we **<DO>** have the values of W and Y_1.
How do I get back to the original Y_2 and its first derivative? I tried with $Y_2 = a$ and $Y_2' = b$ and failed.

Ignacio Larrosa Cañestro ilarrosa@MUNDO-R.COM

You are looking for a function y(t). You must substitute a = y(t) and b = y'(t). Then, you can get a differential equation. You can solve it with DSOLVE1_GEN, by example, see the attached file

Ignacio showed a unique solution. I believe that the answer is not unique. However I followed Igancio´s advice, Josef.

```
#21:  y(x) :=
```

$$\#22:\quad \mathrm{DET}\begin{bmatrix} t^2 + 5\cdot t & y(t) \\ 2\cdot t + 5 & y'(t) \end{bmatrix} = 10\cdot t^2$$

$$\#23:\quad (t^2 + 5\cdot t)\cdot y'(t) - (2\cdot t + 5)\cdot y(t) = 10\cdot t^2$$

$$\#24:\quad \mathrm{DSOLVE1\_GEN}(-(2\cdot t + 5)\cdot z - 10\cdot t^2,\ t^2 + 5\cdot t,\ t,\ z)$$

$$\#25:\quad \frac{z + 10\cdot t}{t\cdot(t + 5)} = c$$

$$\#26:\quad \mathrm{SOLVE}\left(\frac{z + 10\cdot t}{t\cdot(t + 5)} = c,\ z\right) = (z = t\cdot(c\cdot(t + 5) - 10))$$

For applying DSOLVE I replace $y(t)$ by z (as a function of t).

A family of solution functions:

```
#27:  VECTOR(y(t) = t·(c·(t + 5) - 10), c, -5, 5)
```

$$\#28:\quad [y(t) = -5\cdot t\cdot(t + 7),\ y(t) = -2\cdot t\cdot(2\cdot t + 15),\ y(t) = -t\cdot(3\cdot t + 25),\ y(t) = -2\cdot t\cdot(t + 10),\ y(t) = -t\cdot(t + 15),\ y(t) = -10\cdot t,\ y(t) = t\cdot(t - 5),\ y(t) = 2\cdot t^2,\ y(t) = t\cdot(3\cdot t + 5),\ y(t) = 2\cdot t\cdot(2\cdot t + 5),\ y(t) = 5\cdot t\cdot(t + 3)]$$

As you can see one of the solution functions is $y = t^2 – 5t$ ($c = 1$).

The Birthday of the King

Roland Schröder, Celle, Germany

Dodon, the fairy king, captured 100 enemies during a campaign. He jailed them in 100 solitary cells. At the occasion of his birthday some of them should be given freedom. Imaginatix, Dodon's court mathematician, suggested a very strange procedure for selecting the lucky prisoners:

The cell doors are numbered from 1 through 100. 100 days prior to His Majesty's birthday all doors are opened – but the guards take care that nobody is leaving his cell. Next day every second door will be closed and on the third day the state of every third day will be changed: locked doors are opened and open doors are locked. The same will happen with every 4th door on the 4th day. This procedure will go on until the celebration day. Then the prisoners whose doors are open can leave the jail. Which doors are now open? What do you think?

Let us perform the procedure for the first five days for cells #1 through #20 (by hands): (1 … door is open, 0 … door is closed)

```
1   [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]
2   [1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0]
3   [1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0]
4   [1, 0, 0, 1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 0, 0, 1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 0, 0, 1]
5   [1, 0, 0, 1, 0, 1, 1, 1, 0, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1]
```

In the language of DERIVE the […] expression is a VECTOR, say *v*, and its *k*th component is addressed as *v* SUB *k* or *v*↓*k*.

The door #*k* will be unchanged on day #*i* if *k* is not a multiple of *i*. This statement can be described by an IF – THEN – ELSE construct:

IF. *k* is a multiple of *i*
THEN: the state of the door will be changed
ELSE: the state of the door remains unchanged.

The statement "the state of the door will be changed" shows the IF – THEN – ELSE structure again:

IF.	the door is open	*v*↓*k* = 1
THEN:	the door will be closed	0
ELSE:	the door will be opened	1

These considerations lead us to a nested IF – THEN – ELSE construct:

IF. *k* is a multiple of *i*
THEN: IF. door is open
  THEN: door will be closed
  ELSE: door will be opened
ELSE: door remains unchanged.

We have to translate "*k is a multiple of i*" into the language of DERIVE: MOD(k,i) = 0 which says that the remainder of *k* divided by *i* is 0.

This is in Meta-language (but not yet in DERIVE syntax):

IF MOD(*k*,*i*)=0
  THEN
  IF *v*↓*k* =1
    THEN door will be closed
    ELSE: door will be opened
  ELSE: door remains unchanged.

In DERIVE syntax it reads: `IF(MOD(k,i)=0,IF(v↓k= 1,0,1),v↓k).`

We will have our DERIVE procedure as general as possible. The door numbers *k* shall run from 1 through *n* (the Dodon case will be *n* = 100).
What happens on day *i* can be described in another vector with vector *v* as its base:

```
VECTOR(IF(MOD(k,i) = 0, IF(v↓k = 1, 0, 1), v↓k), k, 1, n).
```

We can check this. Take the state of day #4 (the first 20 doors = dim(*v_*)) from above, name it *v_* then our procedure should return the state of the 20 doors on day *i* = 5:

```
v_ := [1, 0, 0, 1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 0, 0, 1, 1, 1, 1, 1]

VECTOR(IF(MOD(k, 5) = 0, IF(v_  = 1, 0, 1), v_ ), k, 1, DIM(v_))
                              k               k

[1, 0, 0, 1, 0, 1, 1, 1, 0, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 0]
```

It makes sense to show the number of the day together with the state of the doors, so we include this number and define a function of three variables (*i* = number of the day, *v* = last state of the doors, *n* = total number of the cell doors).

```
F(i, v, n) := [i, VECTOR(IF(MOD(k, i) = 0, IF(v  = 1, 0, 1), v ), k, 1, n)]
                                               k              k
```

As we want to work recursive (the state of one day is always the consequence of applying the same rule on the state one day before – state(v_i) = rule(state(v_{i-1})). Each recursive procedure needs an initial element which is in our case obviously the state of the doors on day #1 – all doors are open = all elements of the vector are 1:

```
start(n) := VECTOR(1, k_, n)
```

The ITERATES command is an excellent tool for programming recursive procedures. The recursion which creates the vector v_{i+1} from the vector v_i (and the day# *i*+1 from the day# *i*, of course).

```
prison(n, m) := ITERATES(F(i + 1, v, n), [i, v], [1, start(n)], m - 1)

prison(20, 10)
```

This command should show the state of the first *n* = 20 doors for the first *m* = 10 days:

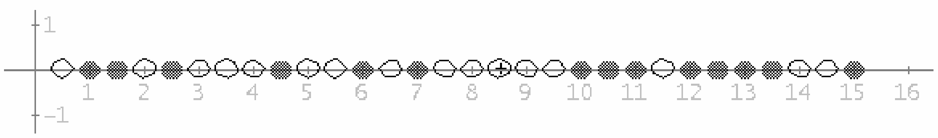
1	[1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1]
2	[1,	0,	1,	0,	1,	0,	1,	0,	1,	0,	1,	0,	1,	0,	1,	0,	1,	0,	1,	0]
3	[1,	0,	0,	0,	1,	1,	1,	0,	0,	0,	1,	1,	1,	0,	0,	0,	1,	1,	1,	0]
4	[1,	0,	0,	1,	1,	1,	1,	1,	0,	0,	1,	0,	1,	0,	0,	1,	1,	1,	1,	1]
5	[1,	0,	0,	1,	0,	1,	1,	1,	0,	1,	1,	0,	1,	0,	1,	1,	1,	1,	1,	0]
6	[1,	0,	0,	1,	0,	0,	1,	1,	0,	1,	1,	1,	1,	0,	1,	1,	1,	0,	1,	0]
7	[1,	0,	0,	1,	0,	0,	0,	1,	0,	1,	1,	1,	1,	1,	1,	1,	1,	0,	1,	0]
8	[1,	0,	0,	1,	0,	0,	0,	0,	0,	1,	1,	1,	1,	1,	1,	0,	1,	0,	1,	0]
9	[1,	0,	0,	1,	0,	0,	0,	0,	1,	1,	1,	1,	1,	1,	1,	0,	1,	1,	1,	0]
10	[1,	0,	0,	1,	0,	0,	0,	0,	1,	0,	1,	1,	1,	1,	1,	0,	1,	1,	1,	1]

You may compare with `prison(10,20)`.

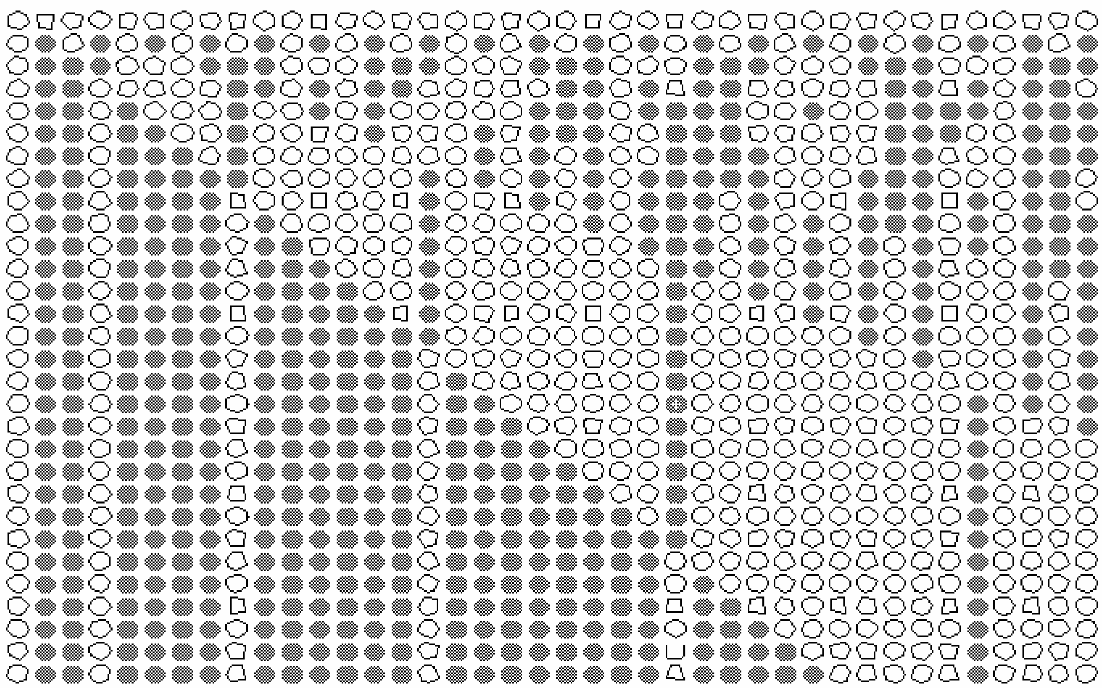
Play a little bit with this function until you get an idea which doors will stay open on Dodon's great day. Can you find the rule behind?

It would be great if we could add a visualisation of the situation(s). Day #5 for the first 30 doors and the history for the first 40 days could look like this:

#15: doors(30, 5)



#18: time_doors(40, 30)



We leave this for the reader. The procedures can be found in the file.

RK6-A Modern Runge-Kutta-Method

Heinrich Ludwig, Raubling, Germany

1. Ein Kurzportrait der Verfahren

Runge-Kutta-Verfahren sind eine Familie von Lösungsverfahren für gewöhnliche Differentialgleichungen (DGl) erster Ordnung $y' = f(x,y)$ mit gegebenen Anfangsbedingungen. Sie werden eingesetzt, wenn die analytische Lösung der DGl nicht bekannt ist und eine Näherungslösung als Ersatz genügt.

Computeralgebrasysteme bringen numerische Verfahren zum Lösen von DGln mit, in vielen Fällen ein explizites Runge-Kutta-Verfahren (RK). Auch DERIVE stellt mit der Funktion RK ein solches Verfahren zur Verfügung. Es ist das „klassische" RK-Verfahren, das Runge schon 1895 veröffentlicht hat. Heute ist es nur noch von historischer Bedeutung; seine Weiterentwicklungen sind wesentlich effizienter und haben zudem ergänzende nützliche Eigenschaften. Mein Ziel war es, DERIVE mit einem modernen RK-Verfahren auszustatten, um für Simulationen in meinem Physikunterricht ein leistungsfähiges Werkzeug zur Verfügung zu haben.

Den Algorithmus eines RK-Verfahrens zu erklären würde den Rahmen dieses Beitrags sprengen. Viele Bücher, die in die Grundlagen der numerischen Mathematik einführen, behandeln das Thema. Besonders ausführlich ist das Buch von E. Hairer, S. P. Nørsett, G. *Wanner: Solving Ordinary Differential Equations*, 2. Auflage, Springer-Verlag, 1993. Ein guten Überblick über die jüngere Entwicklung auf dem Gebiet bietet der Aufsatz von W. H. Enright, D. J. Higham, B. Owren, Ph. W. Sharp: *A Survey of the Explicit Runge-Kutta Method*, TR 291/94, Department of Computer Science, University of Toronto, 1994 (zur Zeit verfügbar über www.cs.toronto.edu/pub/reports/na/rk.survey.96.ps.gz)[1].

Ich habe das Verfahren von Papacostas, Papageorgiou und Tsitouras (SIAM J. *Numerical Analysis*, 33 (1996) S. 917-936) für eine Umsetzung in DERIVE-Code gewählt. Das Verfahren gehört zu den effizientesten, die in den letzten Jahren veröffentlicht wurden. Der Algorithmus selbst ist in Ausdruck #7 formuliert. Er benötigt eine Reihe von Koeffizienten, die in der Matrix A5 und in den Vektoren B5, C5 und D5 zusammengefasst werden.

1. A short description of the methods

Runge-Kutta-Methods are a family of solution methods for ordinary first order differential equations with given initial conditions. They are applied if the analytical solutions of the ODE is unknown and an approximative solution is sufficient.

Computer algebra systems offer numerical methods for solving differential equations which in most cases include an explicit Runge-Kutta-procedure (RK). So does DERIVE with function RK making such a procedure available. This is the "classic" method published by Runge in 1885. Nowadays it is only of historic importance, its further developments are much more efficient and contain additional useful features. It was my aim to implement a modern RK-procedure in DERIVE in order to have a modern and efficient tool for simulations in my physics teaching available.

Explaining the algorithm of a RK-method would break the contents of this contribution. There are many books, treating the basics of numerical mathematics dealing with this subject. Especially comprehensive is the book written by E. Hairer, S. P. Nørsett, G. Wanner: *Solving Ordinary Differential Equations*, 2. Edition, Springer-Verlag, 1993. A good survey over the latest development on this field is given by a paper published by W. H. Enright, D. J. Higham, B. Owren, Ph. W. Sharp: *A Survey of the Explicit Runge-Kutta Method*, TR 291/94, Department of Computer Science, University of Toronto, 1994 (available for download at www.cs.toronto.edu/pub/reports/na/rk.survey.96.ps.gz).[1]

[1] Sie können die pdf-Datei unter downloadbaren Dateien zu diesem DNL finden.
[1] You can find the pdf-file among the downloadable files connected with this DNL.

I chose the method of Papacostas, Papageorgiou and Tsitouras (SIAM J. *Numerical Analysis*, 33 (1996) S. 917-936) for transferring it into DERIVE-Code gewählt. This method is one of the most efficient published in the last years. The algorithm can be found in expression #7. It needs a couple of coefficients which are collected in matrix A5 and in the vectors B5, C5, and D5.

#1: [CaseMode := Sensitive, InputMode := Word, Notation := Decimal]

#2: [PrecisionDigits := 20, PrecisionDigits := 20]

#3: $$A5 := \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{700}{12757} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{2277859}{142293924} & \frac{1321930}{17797209} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{2147197}{125328051} & \frac{23026816}{342342975} & -\frac{384743}{34198910} & 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{291386539}{55002258} & -\frac{119621649}{271157072} & \frac{2798806734}{97726297} & -\frac{1837543755}{55776281} & 0 & 0 & 0 & 0 & 0 \\ -\frac{43016312}{9839773} & \frac{45481347}{269218664} & -\frac{1447053163}{71668982} & \frac{3538651296}{143594951} & \frac{34474522}{90990629} & 0 & 0 & 0 & 0 \\ \frac{527404067}{134932851} & \frac{92630627}{108312167} & \frac{1860989051}{82010587} & -\frac{1403078476}{52174661} & -\frac{138841401}{117689617} & \frac{82333737}{51439001} & 0 & 0 & 0 \\ \frac{1407304125}{293501414} & \frac{1076954593}{980453270} & \frac{3324345790}{119511403} & -\frac{1676391597}{50682665} & -\frac{86106570}{53800223} & \frac{203157280}{102205023} & -\frac{3785594}{184956015} & 0 & 0 \\ \frac{18933895}{59901407} & 0 & \frac{919731731}{308106572} & -\frac{3933591191}{1335303448} & \frac{35451175}{164731901} & \frac{73396207}{298398716} & \frac{189726995}{268693409} & -\frac{104915077}{200742721} & 0 \end{bmatrix}$$

#4: $$B5 := \left[\frac{18933895}{59901407}, 0, \frac{919731731}{308106572}, -\frac{3933591191}{1335303448}, \frac{35451175}{164731901}, \frac{73396207}{298398716}, \frac{189726995}{268693409}, -\frac{104915077}{200742721}, 0\right]$$

#5: $$D5 := \left[-\frac{35243815}{177455297}, 0, -\frac{10187410}{61778981}, \frac{8257013237}{12296726964}, \frac{29595575}{112455507}, \frac{59599904}{213941517}, \frac{46892093}{154074174}, -\frac{841}{4121}, \frac{1}{20}\right]$$

#6: $$C5 := \left[0, \frac{700}{12757}, \frac{487}{5394}, \frac{207}{2830}, \frac{4783}{8682}, \frac{3736}{5943}, \frac{6631}{6733}, 1, 1\right]$$

Expression #7 (the Method):

```
RK6(r, v, v0, h, n, na, f1, f2, f3, f4, f5, f6, f7, f8, M) :=
  Prog
    M := VECTOR(0, i, n + 1)
    M↓1 := v0
    na := n
    Loop
      If n < 1 exit
      f1 := LIM(r, v, v0)
      f2 := LIM(r, v, v0 + h·ADJOIN(C5↓2, A5↓2↓1·f1))
      f3 := LIM(r, v, v0 + h·ADJOIN(C5↓3, A5↓3↓1·f1 + A5↓3↓2·f2))
      f4 := LIM(r, v, v0 + h·ADJOIN(C5↓4, A5↓4↓1·f1 + A5↓4↓2·f2 + A5↓4↓3·f3))
      f5 := LIM(r, v, v0 + h·ADJOIN(C5↓5, A5↓5↓1·f1 + A5↓5↓2·f2 + A5↓5↓3·f3 + A5↓5↓4·f4))
      f6 := LIM(r, v, v0 + h·ADJOIN(C5↓6, A5↓6↓1·f1 + A5↓6↓2·f2 + A5↓6↓3·f3 + A5↓6↓4·f4 + A5↓6↓5·f5))
      f7 := LIM(r, v, v0 + h·ADJOIN(C5↓7, A5↓7↓1·f1 + A5↓7↓2·f2 + A5↓7↓3·f3 + A5↓7↓4·f4 + A5↓7↓5·f5 + A5↓7↓6·f6))
      f8 := LIM(r, v, v0 + h·ADJOIN(C5↓8, A5↓8↓1·f1 + A5↓8↓2·f2 + A5↓8↓3·f3 + A5↓8↓4·f4 + A5↓8↓5·f5 + A5↓8↓6·f6 + A5↓8↓7·f7))
      v0 :+ h·ADJOIN(1, B5↓1·f1 + B5↓2·f2 + B5↓3·f3 + B5↓4·f4 + B5↓5·f5 + B5↓6·f6 + B5↓7·f7 + B5↓8·f8)
      M↓(na - n + 2) := v0
      n :- 1
    RETURN M
```

Wert wurde darauf gelegt, dass die Funktion RK6 sich genauso einsetzen lässt wie die Funktion RK(r,v,v0,h,n), die DERIVE mitbringt. Vorhandene dfw- und mth-Dateien lassen sich so ohne Mühe aufwerten. Man muss in diesen Dateien nur "RK" durch "RK6" ersetzen oder in der Datei ODEApproximation.mth die vorhandene Funktion RK durch RK6 ersetzen und zu RK umbenennen. Natürlich müssen A, B5, C5 und D5 auch in die Zusatzdatei aufgenommen werden.

Value was set on the demand that function RK6 can be used in the same way as function RK(r,v,v0,h,n) which is implemented in DERIVE. So it is easy to improve existing dfw- and mth-files by only replacing RK by RK6 or by replacing RK by RK6 in the utility file ODEApproximation.mth and then rename RK6 as RK. Of course, A, B5, C5 and D5 must also be included in the utility file.

Das eigentliche RK-Verfahren besteht nur aus der Schleife innerhalb der Funktion RK6. Das Ergebnis eines Schleifendurchgangs dient als Startwert für den nächsten Durchgang. Nach *n* Durchgängen gibt RK6 sein Ergebnis zurück.

Ein Mangel von RK, der notwendigerweise auch RK6 anhaftet, ist, dass die Schrittweite aller RK-Einzelschritte dieselbe ist. Die Folge sind von Schritt zu Schritt stark schwankende lokale Diskretisierungsfehler über das Integrationsintervall [v0↓1,v0↓1+h·n]. Moderne RK-Module haben die Fähigkeit, die Schrittweite selbsttägig so zu steuern, dass der lokale Diskretisierungsfehler annähernd konstant bleibt. Die Verfahren werden dadurch nicht nur erheblich effizienter; bei manchen Anfangswertproblemen ist die fortwährende Anpassung der Schrittweite unbedingt notwendig. Aus diesem Grund habe ich RK6 um eine Schrittweitensteuerung (step size control) zur Funktion RK6A ergänzt. RK6A wird an der Stelle erklärt, wo sie für eines der folgenden Beispiele verwendet wird.

The essence of the RK-procedure is given by the loop within function RK6. The result of one run of the loop serves as initial value for the next run. RK6 returns its result after having performed *n* loops.

There is one deficiency of RK and therefore for RK6 as well: the step width is the same for all single RK-steps. The consequence of this deficiency are strong step by step changing local discretization errors over the interval of integration [v0↓1,v0↓1+h·n]. Modern RK-modules have the ability to control the step width automatically in such a way that the local discretization error approximately remains constant. These procedures do not become only more efficient but for some initial value problems perpetual adaptation of the step width is an absolute need. This was reason enough for me to improve RK6 to function RK6A by including a step size control. RK6A It will be explained when it is used for one of the following examples.

2. Der Genauigkeitsgewinn mit RK6

Die erste Aufgabe für RK6 sei das Keplerproblem, die Bewegung eines Massepunkts in einem radialen $1/r^2$-Kraftfeld bei gegebenen Anfangsbedingungen. Für den Ortsvektor $\underline{r} = [x, y]$ des Massepunkts gilt die DGl

$$\underline{r}'' = f(\underline{r})$$

wobei $f(\underline{r})$ das Kraftfeld ist, in dem sich der Massepunkt bewegt. In diesem Fall ist das Kraftfeld zeit- und geschwindigkeitsunabhängig. Gesucht ist die Bahn $\underline{r}(t)$. RK-Verfahren lösen Differentialgleichungen 1. Ordnung. Deshalb muss die o.g. DGl 2. Ordnung in ein System von gekoppelten DGln 1. Ordnung umgewandelt werden. Dabei ist $\underline{v} = [vx, vy]$ der Geschwindigkeitsvektor des Massepunkts.

$$(1) \quad \underline{r}' = \underline{v}$$
$$(2) \quad \underline{v}' = f(\underline{r})$$

Beide DGln löst das RK-Verfahren gleichzeitig, wenn man $\underline{v}$ und $f(\underline{r})$ zu einem 4-dimensionalen Vektor zusammenfasst. Die ersten beiden Parameter der Funktion RK6 im Ausdruck #8 zeigen, wie das gemeint ist. Die Spaltenselektion ↓↓[2,3] am Ende des Ausdrucks wählt aus den Ergebnisvektoren der Funktion RK6 die beiden Ortskoordinaten $[x, y] = \underline{r}$ und lässt die beiden Geschwindigkeitskoordinaten weg.

2. The win of accuracy working with RK6

The first task applying RK6 is Kepler´s Problem, i.e. the movement of a mass point in a radial $1/r^2$ field of force with given initial conditions. To the position vector $\underline{r} = [x, y]$ of the mass point applies the following DE

$$\underline{r}'' = f(\underline{r})$$

with $f(\underline{r})$ being the force of field where the mass point is moving. In this case the field is independent of time and velocity. We want to find the orbit $\underline{r}(t)$. RK-methods solve 1st order differential equations. So we need to transform the DE from above which is of 2nd order into a system of coupled 1st order DEs. $\underline{v} = [vx, vy]$ is the velocity vector of the mass point.

(1) $\underline{r}' = \underline{v}$
(2) $\underline{v}' = f(\underline{r})$

The RK-procedure solves both DEs simultaneously if one combines $\underline{v}$ and $f(\underline{r})$ to one vector of 4 dimensions. The first two parameters of function RK6 in expression #8 explain how this is meant. Selection of columns `↓↓[2,3]` at the end of the expression selects the two position coordinates $[x,y] = \underline{r}$ for ouput (and for plotting) from the resulting vectors and omits the velocity coordinates.

#8: $$\text{bahn1} := \left(\text{RK6}\left(\left[vx,\ vy,\ -\frac{1}{\sqrt{(x^2+y^2)^3}}\cdot x,\ -\frac{1}{\sqrt{(x^2+y^2)^3}}\cdot y\right],\ [t,\ x,\ y,\ vx,\ vy],\ [0,\ 0.2,\ 0,\ 0,\ 3],\ \frac{\pi}{50},\ 50\right)\right)\downarrow\downarrow[2,\ 3]$$

Dieses spezielle Anfangswertproblem wurde gewählt, weil es für $t = \pi$ eine exakte Lösung gibt. Mit ihr lässt sich die vom RK-Verfahren gefundene Lösung vergleichen und die Genauigkeit des RK-Verfahrens bewerten. Wenn die Kraft auf den Massepunkt zum Zentrum [0, 0] weist und seine Anfangsgeschwindigkeit nicht zu groß ist, stellt $[x(t),y(t)]$ eine Ellipse dar. Die Anfangswerte $x(0) = 0.2$, $y(0) = 0$, $v_x(0) = 0$ und $v_y(0) = 3$ wurden so gewählt, dass die Bahnellipse die große Halbachse $a = 1$ hat und die Bahn in der Periapsis beginnt. (**Periapsis** ist dabei der Punkt auf der Umlaufbahn mit der kleinsten Entfernung zum Hauptkörper und **Apoapsis** der mit der größten.) Das Produkt aus Schrittweite h und dem Wiederholungszähler n wurde gleich $h{\cdot}n = \pi$ gesetzt. Das entspricht der halben Umlaufdauer auf einer Ellipse mit Halbachse $a = 1$. Wenn RK6 gut rechnet, sollte der letzte der ermittelten Lösungspunkte die Apoapsis der Bahn sein, die um $2a$ von der Periapsis entfernt liegt , also bei [-1.8, 0].

Wie gut die von RK6 ermittelte Lösung die Apoapsis trifft, zeigt die folgende Abbildung. Der Fehler ist im gewählten Maßstab nicht zu erkennen. (Er beträgt 0,00047.) Zum unmittelbaren Vergleich wurde mit Ausdruck #9 das gleiche Anfangswertproblem auch mit dem „klassischen" RK-Verfahren gelöst. Der Unterschied der Ergebnisse ist deutlich.

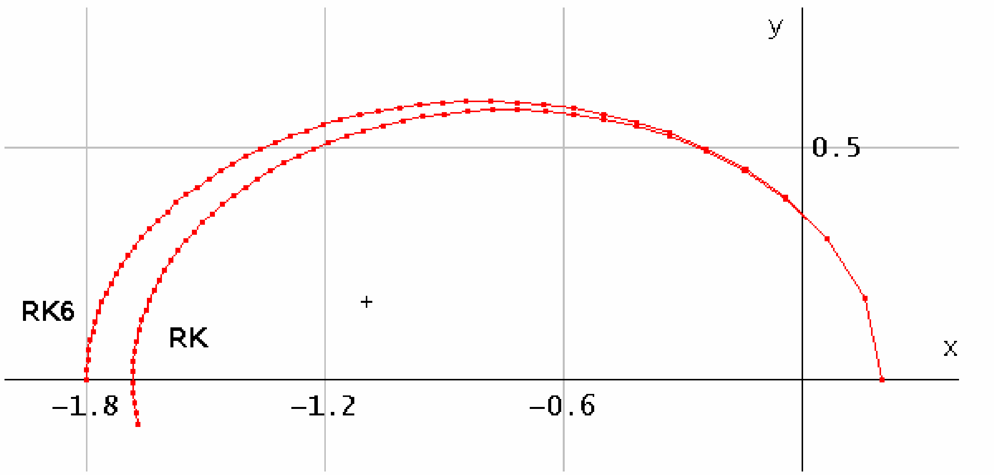
Wichtig: Die Funktionen RK6 und RK6A müssen durch Approximation vereinfacht werden (Tastenkombination Strg+G). Andernfalls steigt die Rechenzeit erheblich an. Eine exakte Vereinfachung bringt schon deshalb keinen Gewinn, weil ein RK-Verfahren ein Anfangswertproblem grundsätzlich nur näherungsweise löst. Die Plots können auch ohne vorherige Berechnung im Algebrafenster direkt gezeichnet werden - wie zB den Ausdruck #9 auf der nächsten Seite. Es muss aber die Option „Approximate before Plotting" aktiviert werden.

This special initial value problem was chosen because of its exact result for $t = \pi$. So it is possible to compare the approximative solution given by the RK-method with the exact solution and we can evaluate the accuracy of this procedure. If the force acting on the mass point is directed towards the centre [0, 0] and its initial velocity is not too high then $[x(t),y(t)]$ appears as an ellipse. The initial values $x(0) = 0.2$, $y(0) = 0$, $v_x(0) = 0$ and $v_y(0) = 3$ were chosen such that the major axis of the orbit ellipse $a = 1$ and the orbit starts at the periapsis. (**Periapsis** the point on the orbit with minimum distance from the mass M, and **aoapsis** is the point with maximum distance.) The product of step size h and the repetition counter n was set $h \cdot n = \pi$. This is equivalent to the half period of revolution on an ellipse with half major axis $a = 1$. If RK6 does really a good job then the last of the calculated solution points should be the aoapsis of the orbit which is at [-1.8, 0] – in a distance of $2a$ from the periapsis.

The quality of the solution calculated using RK6 meets the aoapsis shows the following figure. The error cannot be recognized in the chosen scale. (It is 0,00047.) To have an immediate comparison the same IVP was solved by the “classic“-RK-method (expression #9 which is part of the utility file). The difference of the results is clear.

Important: Functions RK6 and RK6A must be simplified by approximation (key combination Ctrl+G). Otherwise calculation time would increase enormously. There is no benefit in exact Simplification because a RK-method basically solves an IVP only approximatively. It is not necessary to do the calculation in the Algebra Window. You can immediately plot the highlighted expression (like #9 below) in the 2D Plot Window but you have to activate there option “Approximate before Plotting”.

#9: $$\text{bahn2} := \left(\text{RK}\left(\left[vx,\ vy,\ -\frac{1}{\sqrt{(x^2+y^2)^3}}\cdot x,\ -\frac{1}{\sqrt{(x^2+y^2)^3}}\cdot y\right],\ [t,\ x,\ y,\ vx,\ vy],\ [0,\ 0.2,\ 0,\ 0,\ 3],\ \frac{\pi}{50},\ 50\right)\right)\downarrow\downarrow[2,\ 3]$$



Wie man sieht, übertrifft RK6 RK an Genauigkeit bei weitem. Freilich kann man die Schrittweite h so weit verringern, dass auch RK keinen so großen Fehler mehr erzeugt. Eine Verringerung von h kommt aber RK6 noch viel mehr zugute als RK, da der Fehler von RK6 ungefähr proportional zu h^6 abnimmt, der von RK nur proportional zu h^4. Gerade, wenn hohe Genauigkeit erforderlich ist, sind Verfahren hoher Konvergenzordnung effizienter.

As you can see RK6 is much more accurate than RK. Though one could decrease step size *h* in such a way that even RK does not produce such errors. But then RK6 benefits much more of the decreased step size than RK because the error of RK6 declines proportional to h^6 and the error of RK declines only proportional to h^4. Especially when high accuracy is needed procedures of high order of convergence are more efficient.

3. Bewegung in drei Dimensionen

Die Physik kennt interessantere Anwendungen für RK-Verfahren als das Keplerproblem. Wie sieht etwa die Bahn eines geladenen Teilchens aus, das sich einem magnetischen Dipol nähert? Ein Beispiel dafür sind die Partikel des Sonnenwindes, die sich unter dem Einfluss des Erdmagnetfelds der Erde nähern. Die Bahn der Teilchen ist nicht eben, ihre Berechnung muss in drei Dimensionen erfolgen. Ausdruck #10 liefert die rechte Seite der Bewegungsdifferentialgleichung. Man erkennt darin das Kreuzprodukt, das für die Lorentz-Kraft kennzeichnend ist. Im Ausdruck #12 wird die Bahn berechnet.

3. Motion in three dimensions

In physics we can find applications of RK-procedures which are much more interesting than Kepler´s Problem. Let's ask for the orbit of a charged particle approaching a magnetic dipole? One example for this are the particles of the solar wind approaching earth influenced by its magnetic field. The orbits of the particles are not plane, their calculation needs three dimensions. Expression #10 delivers in its right side the differential equation of the motion. One can recognize the cross product which is significant for the Lorentz force. Expression #12 calculates the orbit of the particle.

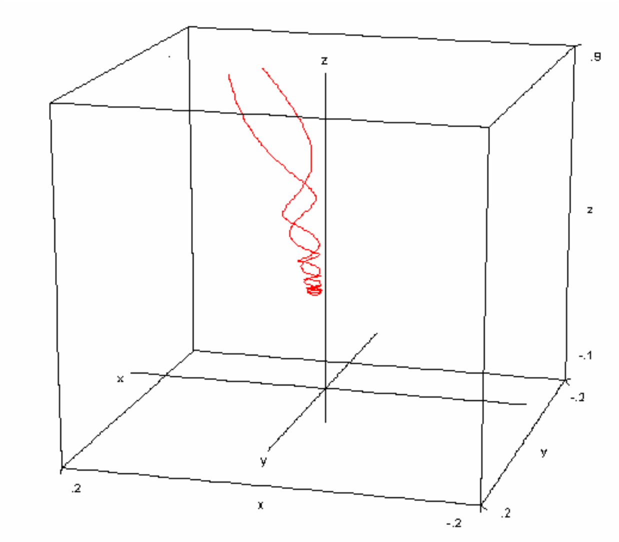
```
      magnetdipol(w, a, e_, f_, g_) :=
        Prog
          e_ := [3·w↓2·w↓4, 3·w↓3·w↓4, 2·w↓4^2 - w↓2^2 - w↓3^2]
#10:      f_ := [w↓5, w↓6, w↓7]
          g_ := CROSS(f_, e_)·a/√(w↓2^2 + w↓3^2 + w↓4^2)^5
          RETURN APPEND(f_, g_)
```

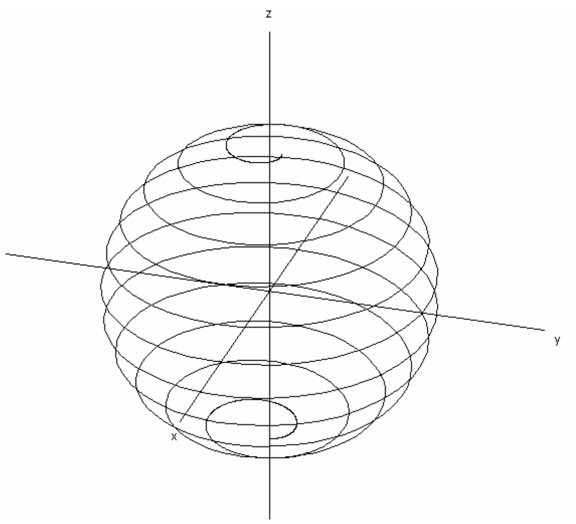
#11:
$$\left[w_5,\ w_6,\ w_7,\ \frac{a\cdot\left(w_6\cdot\left(2\cdot w_4^2 - w_3^2 - w_2^2\right) - 3\cdot w_3\cdot w_4\cdot w_7\right)}{\left(w_4^2 + w_3^2 + w_2^2\right)^{2.5}},\ \frac{a\cdot\left(3\cdot w_2\cdot w_4\cdot w_7 - w_5\cdot\left(2\cdot w_4^2 - w_3^2 - w_2^2\right)\right)}{\left(w_4^2 + w_3^2 + w_2^2\right)^{2.5}},\ \frac{3\cdot a\cdot w_4\cdot(w_3\cdot w_5 - w_2\cdot w_6)}{\left(w_4^2 + w_3^2 + w_2^2\right)^{2.5}}\right]$$

```
#12:  bahn3 := (RK6(magnetdipol(q, -1), q, [0, 0.1, 0, 1.5, 0, 0, -0.6], 0.01,
        460))↓↓[2, 3, 4]
```

Ein 3D-Graphikfenster zeigt diese Bahn (linkes Bild). Aufschlussreich ist auch das zugehörige Geschwindigkeitsdiagramm (rechtes Bild). Dazu ersetzt man in #12 die Spaltenselektion `↓↓[2,3,4]` durch `↓↓[5,6,7]`. Man sollte für dieses Bild, die Schrittweite *h* des RK-Verfahrens verringern und die Schrittzahl *n* entsprechend erhöhen, damit die Interpolationssehnen des Graphen keine deutlichen Ecken bilden. Der Verlauf des Graphen auf einer Kugeloberfläche ist kein Zufall: In einem zeitlich konstanten Magnetfeld bleibt die kinetische Energie eines Teilchens und damit $v^2 = v_x^2 + v_y^2 + v_z^2$ konstant.

A 3D-Plot Window shows this orbit (left figure). The respective velocity diagram (right figure) is very informative. Replace in #12 selection of columns `↓↓[2,3,4]` by `↓↓[5,6,7]`. One should decrease step size *h* and increase step number *n* for obtaining a smooth graph. The run of the orbit on the surface of a sphere is not pure chance: in a temporal constant magnetic field the kinetic energy of a particle which is $v^2 = v_x^2 + v_y^2 + v_z^2$ remains constant.





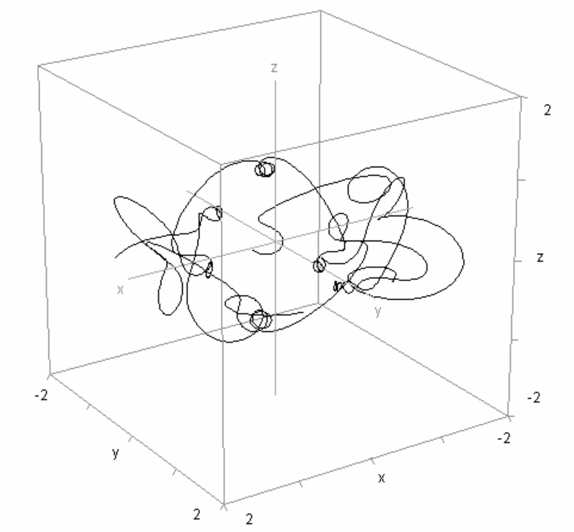
Dieselbe Differentialgleichung mit anderen Anfangswerten liefert eine völlig andere Bahn. Ein Beispiel dafür ist die chaotische Bahn eines Teilchens, das im Van-Allen-Gürtel gefangen ist. Ausdruck #13 berechnet sie. Wegen seiner reizvollen Erscheinung wird auch der Verlauf des Geschwindigkeitsvektors abgebildet (Ausdruck #14 – nächste Seite, rechtes Bild). Aber auch die Bahn selber ist sehenswert (nächste Seite, linkes Bild).

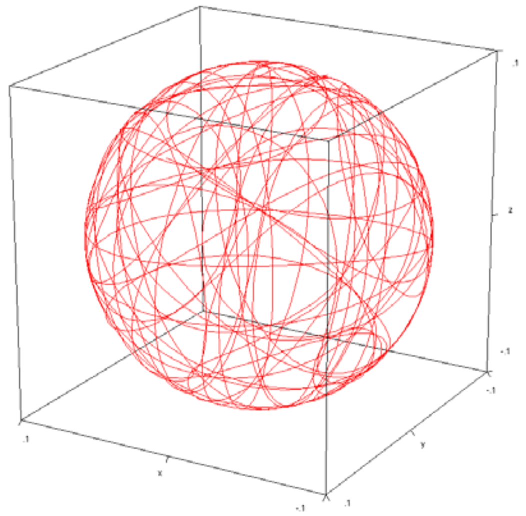
Beide Ausdrucke verwenden vorteilhaft RK6A anstelle von RK6. Die Parameterliste von RK6A unterscheidet sich von der von RK6 bzw. RK. Erläuterungen dazu finden Sie im Anhang.

The same DE with other initial values leads to a very different orbit. One example for this is the chaotic orbit of a particle which is caught in the Van-Allen-Belt. It is calculated in expression #13. In addition we show the run of the velocity vector because of its pretty appearance (next page, right figure). But the orbit itself is also worth seeing (next page, left figure).

Both calculations use the advantage of RK6A compared to RK6. The parameter list of RK6A differs from the RK6 parameter list. Explanations are given in the appendix.

```
#13:  bahn4 := (RK6A(magnetdipol(q, 1), q, [0, 1.82145, -0.595132, 0.0132261,
        0.0150255, 0.0746424, 0.0648287], 500, 0.2, [2, 3, 4], -8))↓↓[2, 3, 4]

#14:  (RK6A(magnetdipol(q, 1), q, [0, 1.82145, -0.595132, 0.0132261, 0.0150255,
        0.0746424, 0.0648287], 10, 0.2, [2, 3, 4], -8))↓↓[5, 6, 7]
```

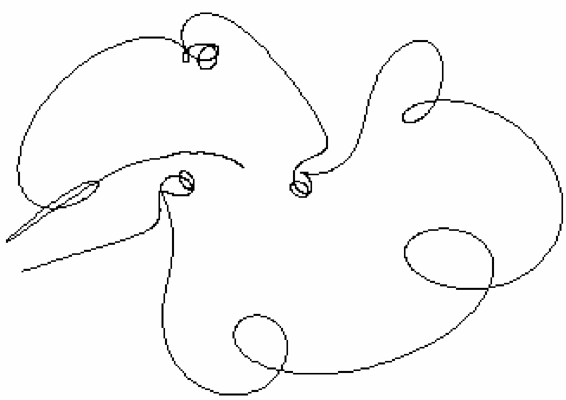

#13


……#14

Schon eine winzige Änderung einer Anfangsbedingung ändert die Bahn nach kurzem Verlauf gänzlich. bahn5 entwickelt für $v↓1$ > 285 eine besonders interessantes Detail. Die Berechnung erfordert viel Zeit, denn die Schrittweite sinkt gegen Ende der Integration beträchtlich. In der Statusleiste zeigt RK6A den Rechenfortschritt an.

A tiny change of the initial values changes the orbit completely. bahn5 shows an especially remarkable detail for $v↓1$ > 285. Its calculations needs much time because the step size becomes very small at the end of the integration process. You can follow the calculation progress in the status bar (bottom left).

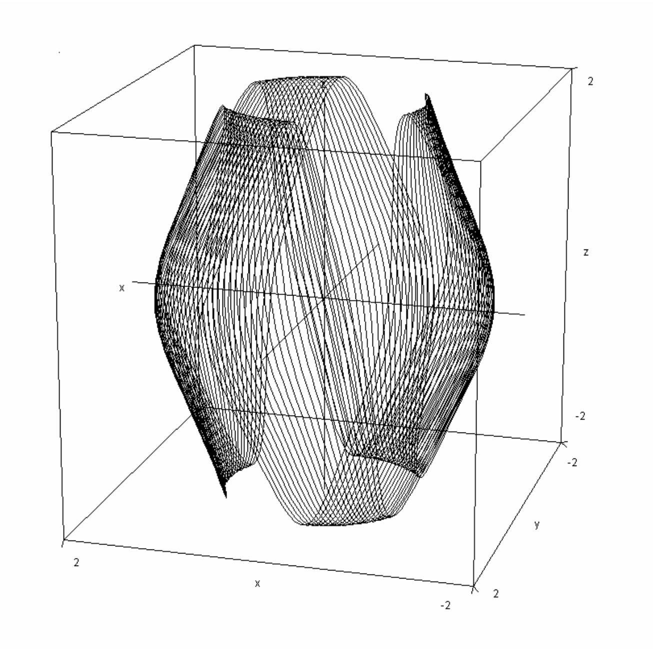
```
#15:  bahn5 := (RK6A(magnetdipol(q, 1), q, [0, 1.82145, -0.595132, 0.0132261,
        0.0150255, 0.0746423, 0.0648287], 289, 0.2, [2, 3, 4], -10))↓↓[2, 3, 4]
```

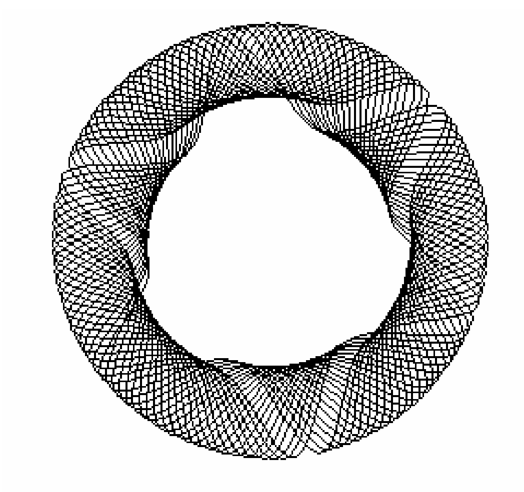

bahn5

Nicht weniger reizvoll ist die Bahn eines geladenen Teilchens in einem oszillierenden elektrischen Quadrupolfeld. Ein solches Feld ist das Wirkungsprinzip einer „Paul-Falle" oder „Quadrupole Ion trap". Auch diese Anwendung liefert eine dreidimensionale Bahn. Lässt man den Kasten im 3D-Fenster rotieren, so erkennt man die Eleganz dieser schönen Bahnkurve erst richtig. Ausdruck #16 liefert die rechte Seite der Differentialgleichung, Ausdruck #17 berechnet die Bahn. Rechenzeit bei 3 GHz CPU-Takt ist ca. 40 s.

The orbit of a charged particle in an oscillating quadrupolar field is not less attractive. Such a field is the action principle of a "Paul-Trap" or "Quadrupole Ion Trap". This application delivers a three dimensional orbit, too. Rotating the box in the 3D Plot Window gives an impression of the elegance of this beautiful orbit. Expression #16 gives the right side of the differential equation and expression #17 calculates the orbit. Calculation time is approximately 30s (3 GHz CPU-beat).

```
      ionfall(w, a, o, p) := APPEND(w          , a·COS(w ·o + p)·[w , w , - 2·w ])
#16:                                 [5, ..., 7]         1          2   3       4
#17:  bahn6 := (RK6(ionfall(q, 1, 2.098823851, 0), q, [0, 0, 1, 2, 0.56, 0, 0], 0.1,
        4000))↓↓[2, 3, 4]
```





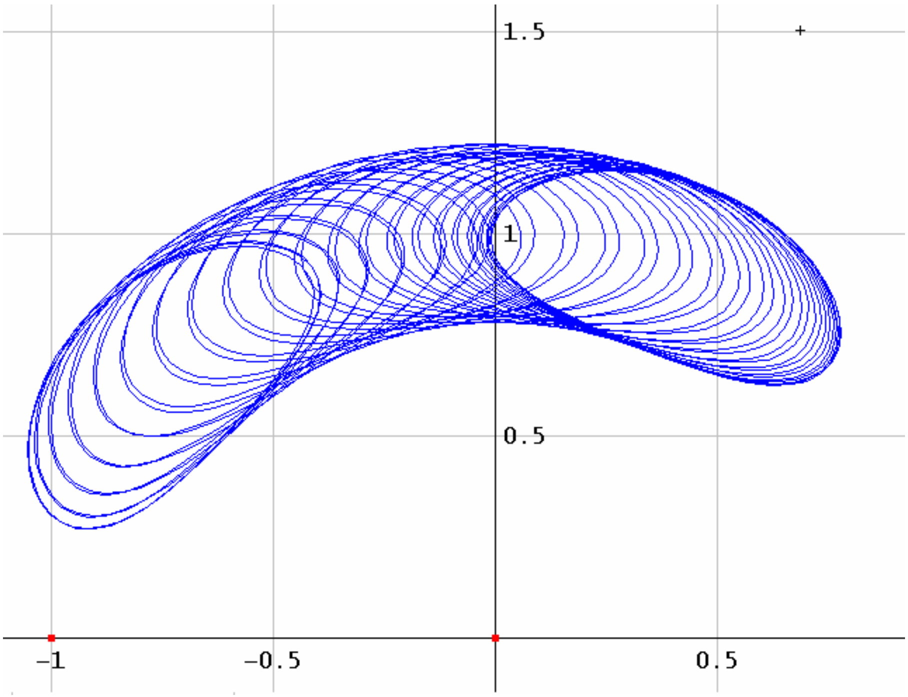
4. Das Drei-Körper-Problem

Das Drei-Körper-Problem liefert eine Vielfalt an Bahnkurven. Ich möchte drei von ihnen vorstellen. Die ersten beiden sind Lösungen des eingeschränkten Drei-Körper-Problems, bei dem die Masse des dritten Körpers so klein ist, dass ihr Einfluss auf die beiden anderen vernachlässigt werden kann. Die sog. Trojaner im Gravitationsfeld von Sonne und Jupiter sind ein Beispiel dafür. Mit den Ausdrücken #18 und #19 erfolgt die Berechnung. Die Hauptmassen haben die Positionen [-0.9999,0] und [0.0001,0] (im Bild als rote Punkte eingetragen).
Die Rechenzeit beträgt bei 3 GHz CPU-Takt ca. 25 sSekunden.

4. The Three Body Problem

The Three Body Problem delivers a variety of orbits. I´d like to present three of them. The first both of them are solutions of the restricted Three Body Problem: in this case the mass of the third body is so little that its influence on both other bodies can be neglected. The so called Trojans in the gravity field of sun and Jupiter are an example for this. Expressions #18 and #19 do the calculation. The main masses have positions [-0.9999,0] and [0.0001,0] (- which are the red points in the graph). Calculation time is approximately 25 s.

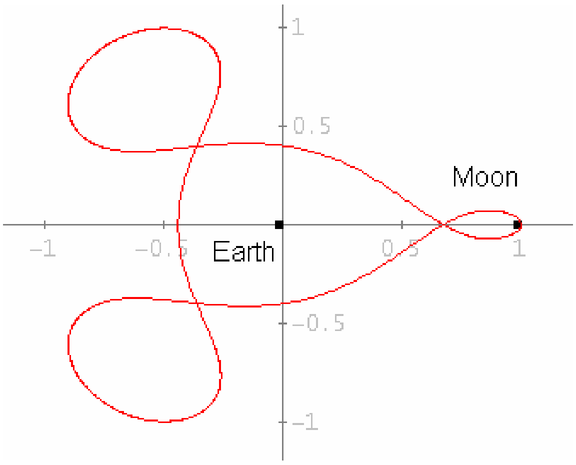
```
      dreikoerp(w, m1, m2, d1, d2, om, k1, k2) :=
        Prog
#18:      k1 := - m1/((w↓2 - d1)^2 + w↓3^2)^(3/2)
          k2 := - m2/((w↓2 - d2)^2 + w↓3^2)^(3/2)
          RETURN [w↓4, w↓5, k1·(w↓2 - d1) + k2·(w↓2 - d2) + om·(om·w↓2 + 2·w↓5),

        k1·w↓3 + k2·w↓3 + om·(om·w↓3 - 2·w↓4)]

#19:  bahn7 := (RK6(dreikoerp(q, 0.9999, 0.0001, 0.0001, -0.9999, 1), q, [0, -0.6,

        1.03923, 0.33773, 0.195], 0.16, 1651))↓↓[2, 3]
```

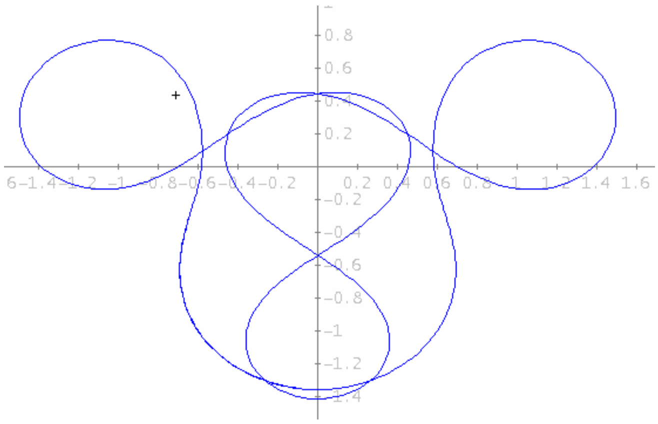


Im Zusammenhang mit dem Mondlandeprogramm der NASA hat Richard Arenstorf periodische Bahnen für Raumfahrzeuge im Erde-Mond-System untersucht (*New periodic solutions of the plane three-body-problem corresponding to elliptic motion in the lunar theory*, J. Different. Equ. 4 (1968), 202-256). Auch ohne Kenntnis dieser Arbeit kann man solche Bahnen mit etwas Geschick selber finden. Hier ist ein Beispiel einer solchen Bahn, wobei die Erde-Mond-Bewegung zu einer Rotation mit konstanter Winkelgeschwindigkeit und konstantem Erde-Mond-Abstand vereinfacht ist. Die Erde ist bei [-0.0123, 0] zu finden, der Mond bei [0.9877, 0]. Man beachte, dass das Raumfahrzeug in der kleinen rechten Schleife dem Mond sehr nahe kommt und sehr stark beschleunigt wird.
(Rechenzeit zum Auswerten des Ausdrucks #20 ca. 15 s bei 3 GHz CPU-Takt.)

Richard Arenstorf investigated in connection with the moon landing program of NASA periodic orbits for space crafts in the Earth-Moon-system (*New periodic solutions of the plane three-body-problem corresponding to elliptic motion in the lunar theory,* J. Different. Equ. 4 (1968), 202-256). Even without knowing this paper but being a little bit skilful one can find such orbits by himself. Here is an example of such an orbit – the earth-moon-motion is simplified to a rotation with constant angular velocity and constant distance between earth and moon. You can find the earth at [-0.0123, 0], and the moon at [0.9877, 0]. Please note that the space craft is approaching moon very close in the right loop and that is accelerated very much. (Calculation time for evaluating expression #20 is approx. 15s.)

```
#20:  bahn8 := (RK6(dreikoerp(q, 0.987722529, 0.012277471, -0.012277471, 0.987722529,
          1), q, [0, 1.002, 0, 0, -1.35661886266766633], 0.01, 1120))↓↓[2, 3]
```





```
(RK6A(dreikoerp(q, 0.987722529, 0.012277471, -0.012277471, 0.987722529, 1), q, [0, 1.002, 0, 0,
  -1.35661886266766633], 11.2, 0.001, [2, 3], -10))↓↓[4, 5]
```

Ich lade Sie ein, das Geschwindigkeitsdiagramm (rechtes Bild) mit den Funktionen RK, RK6 und RK6A zu zeichnen und die Ergebnisse zu vergleichen.

I'd like to invite you comparing RK, RK6 and RK6A plotting the velocity diagram (right figure).

Die Umrundung des Mondes muss sehr kleinschrittig erfolgen, da sonst erhebliche Diskretisierungsfehler entstünden. Hingegen vertragen die großen Schleifen in der linken Hälfte des Koordinatensystems etwa 20-fach größere Integrationsschritte bei gleichem lokalem Diskretisierungsfehler pro Schritt. RK6A ist darum für dieses Anfangswertproblem viel besser geeignet als RK6. Der folgende Ausdruck #21 berechnet die Schleifenbahn noch einmal mit RK6A. Die Rechenzeit ist kürzer, das Ergebnis wesentlich genauer.

Looping the moon must be performed in very small steps, because otherwise we would find serious discretisation errors. On the other hand the big loops in the left half of the system of coordinates tolerate 20-fold larger integration steps having the same local discretisation error per step. So RK6A is much more appropriate for this IVP than RK6. The following expression #21 calculates the orbit of the loop once more but using RK6A. Calculation time is less and the result is significantly more accurate.

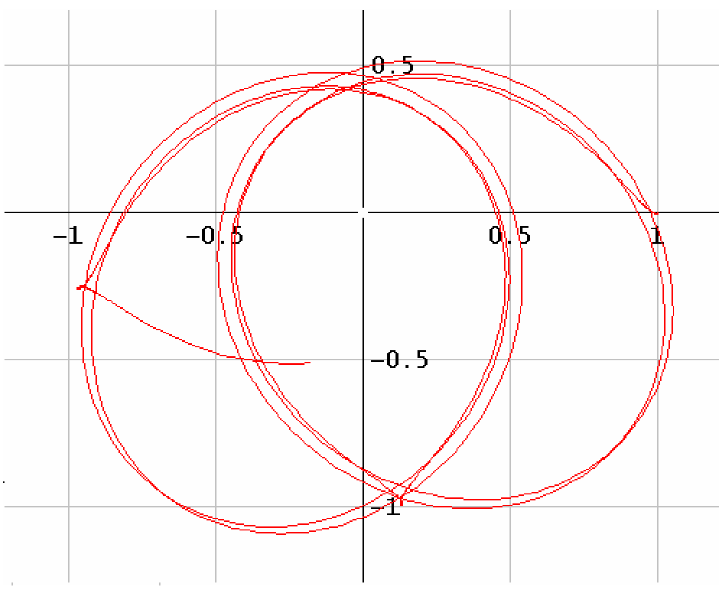
```
#21:  bahn8a := (RK6A(dreikoerp(q, 0.987722529, 0.012277471, -0.012277471,
        0.987722529, 1), q, [0, 1.002, 0, 0, -1.35661886266766633], 11.2, 0.001, [2,
        3], -10))↓↓[2, 3]
```

Aussagekräftiger als die obige Darstellung im rotierenden Bezugssystem ist die folgende Darstellung der Bahn in einem Inertialsystem. Dazu müssen die Positionsdaten um den Ursprung gedreht werden, wobei der Drehwinkel proportional zurzeit ist. Dies leistet die Funktion `dreh(a, Ω)`. Weil **bahn8a** aus den errechneten RK6A-Ergebnissen die Zeitinformation ausblendet, berechnet Ausdruck #23 die Bahn noch einmal, schließt aber die 1. Spalte der RK6A-Ergebnismatrix mit ein. Wenn man Ausdruck #24 approximiert, erhält man die Bahndaten im Inertialsystem. Das folgende Bild zeigt die Bahn.

More meaningful than the presentation in the rotating reference system (as given above) is the following presentation of the orbit in an inertial system. For this purpose we have to rotate the position data around the origin with the rotation angle being proportional to time. Function `dreh(a,Ω)` does the job. #23 calculates the orbit once more – including the first column of the RK6A-result matrix – because bahn8a from #21 does not show the time information. Approximating #24 gives the orbit data for the inertial system. The following figure shows the orbit.

#22: dreh(a, Ω) := $\text{VECTOR}\left(\begin{bmatrix} \cos(\Omega\cdot a_{i,1}) & \sin(\Omega\cdot a_{i,1}) \\ -\sin(\Omega\cdot a_{i,1}) & \cos(\Omega\cdot a_{i,1}) \end{bmatrix}\cdot\left[a_{i,2}, a_{i,3}\right], i, \text{DIM}(a)\right)$

```
#23:  bahn8b := (RK6A(dreikoerp(q, 0.987722529, 0.012277471, -0.012277471,
        0.987722529, 1), q, [0, 1.002, 0, 0, -1.35661886266766633], 23, 0.001, [2,
        3], -10))↓↓[1, 2, 3]

#24:  dreh(bahn8b, -1)
```



Man verfolge das Raumfahrzeug: Bei [1.002, 0] nahe dem Mond startet es, läuft gegen den Uhrzeigersinn auf einer Kepler-Ellipse um den Schwerpunkt Erde-Mond gut zweidreiviertel Umdrehungen weit. Inzwischen hat sich der Mond gut eindreiviertel Umdrehungen im gleichen Drehsinn um den Systemschwerpunkt gedreht, Raumfahrzeug und Mond begegnen einander wieder. Während des Vorbeiflugs am Mond wird der Geschwindigkeitsvektor des Raumfahrzeugs in kurzer Zeit um ca. 270° im Uhrzeigersinn gedreht. Das Raumfahrzeug durchläuft eine kleine Schleife. Anschließend läuft das Raumfahrzeug auf einer neuen Ellipse um den Systemschwerpunkt bis zur nächsten Begegnung mit dem Mond.

We will follow the space craft: It starts at [1.002, 0] which is close to the moon, and then it runs counter clockwise on a Kepler ellipse around the earth-moon centre of gravity about two and three quarters of rotation. In the meanwhile moon has rotated about one and three quarters in the same direction of rotation around the centre of gravity of the system. Moon and space craft are meeting again. During passing moon the velocity vector of the space craft is rotated clockwise about approximately 270° within a short while. The space craft makes a small loop. And then it runs on a new ellipse around the system centre of gravity until its next meeting with the moon.

Der Sonderfall dreier Körper mit gleicher Masse soll das abschließende Beispiel sein. Bei geeigneten Anfangsbedingungen laufen die drei Körper auf einer gemeinsamen Bahn, die einer Lemniskate ähnelt, einander hinterher. Notwendig dafür ist: Gesamtdrehimpuls $L = 0$, Gesamtimpulsänderung $p' = 0$, folglich kann man $p = 0$ wählen und auch Schwerpunkt $\underline{s} = [0, 0]$. Legt man die Startörter $\underline{r}_1(0) =$ $= [-1, 0]$, $\underline{r}_2(0) = [0, 0]$ und $\underline{r}_3(0) = [1, 0]$ fest, hat man nur noch zwei freie Parameter, um die Bahn zu beeinflussen. Ich habe den Betrag u und die Richtung α der Geschwindigkeit der Masse im Ursprung als Parameter gewählt. Nach einigem Probieren war die Bahn gefunden: $u = 1.271$ und $\alpha = 0.994$. Es macht nichts, dass diese Werte noch nicht sonderlich genau sind. Im Gegensatz zum vorigen Beispiel ist die Bahn auch unter kleinen Störungen stabil. Ausdruck #25 liefert die wechselseitigen Anziehungskräfte zwischen den Körpern, Ausdruck #26 berechnet die Bahnen der drei Körper.

Die Funktion `pu3` dient dazu, dass man diese auf einmal zeichnen kann. Man benutze dafür den Ausdruck #28. Rechenzeit bei 3 GHz CPU-Takt ca. 10 s.

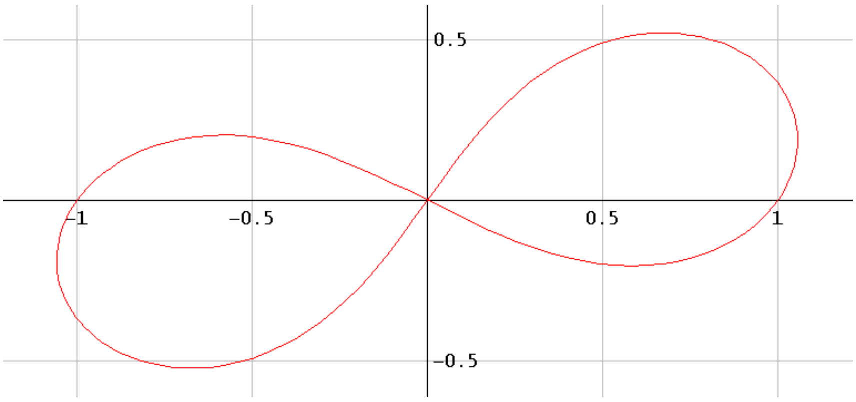
The final example is the special case of three bodies of the same masses. Applying appropriate initial values the three bodies follow each other on a common orbit which is similar to a leminiscate. Necessary for this is a total angular momentum $L = 0$ and a total change of the momentum $p' = 0$. So one can choose $p = 0$ and the centre of gravity $\underline{s} = [0, 0]$. Fixing the start positions $\underline{r}_1(0) = [-1, 0]$, $\underline{r}_2(0) = [0, 0]$ and $\underline{r}_3(0) = [1, 0]$, only two free parameters are remaining in order to influence the orbit: I chose the absolute value u and the direction α of the velocity of the mass which is located in the origin. After some tries I found appropriate values: $u = 1.271$ and $\alpha = 0.994$. It does not matter that these values are not very accurate. In contrary to the example above this orbit remains stable even under small disturbances. Expression #25 gives the mutual forces of attraction between the three bodies: expression #26 calculates their orbits.

Function `pu3` serves to plot them simultaneously. Please use expression #28. Calculation time is approximately 10 s.

```
      dreigleich(w, k1x, k1y, k2x, k2y, k3x, k3y, d1, d2, d3) :=
        Prog
          d1 := - ((w↓7 - w↓11)^2 + (w↓6 - w↓10)^2)^(- 3/2)
          d2 := - ((w↓11 - w↓3)^2 + (w↓10 - w↓2)^2)^(- 3/2)
          d3 := - ((w↓3 - w↓7)^2 + (w↓2 - w↓6)^2)^(- 3/2)
          k1x := d1·(w↓6 - w↓10)
#25:      k1y := d1·(w↓7 - w↓11)
          k2x := d2·(w↓10 - w↓2)
          k2y := d2·(w↓11 - w↓3)
          k3x := d3·(w↓2 - w↓6)
          k3y := d3·(w↓3 - w↓7)
          RETURN [w↓4, w↓5, k3x - k2x, k3y - k2y, w↓8, w↓9, k1x - k3x, k1y - k3y, w↓12, w↓13, k2x - k1x, k2y - k1y]
```

#26: $\text{bahn9}(u, \alpha) := \text{RK6}\left(\text{dreigleich}(q), q, \left[0, -1, 0, -\frac{1}{2}\cdot u\cdot \text{COS}(\alpha), -\frac{1}{2}\cdot u\cdot \text{SIN}(\alpha), 0, 0, u\cdot \text{COS}(\alpha), u\cdot \text{SIN}(\alpha), 1, 0, -\frac{1}{2}\cdot u\cdot \text{COS}(\alpha), -\frac{1}{2}\cdot u\cdot \text{SIN}(\alpha)\right], 0.04, 158\right)$

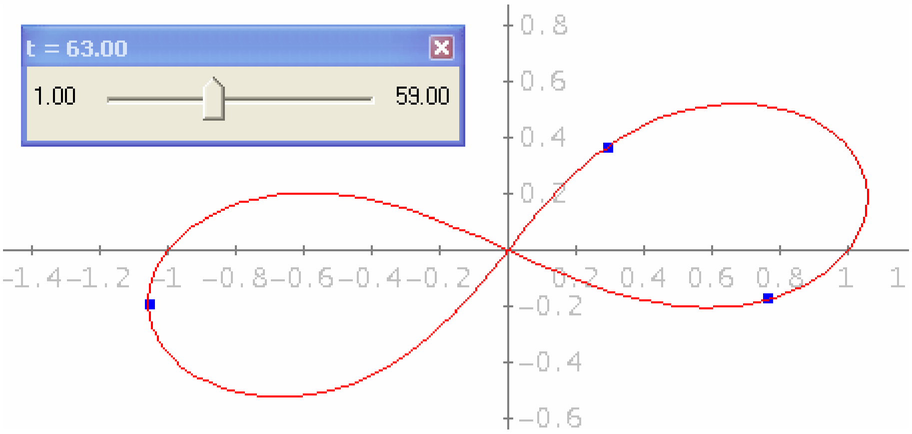
```
      pu3(v, a, m) :=
        Prog
#27:      m := bahn9(v, a)
          RETURN [m↓↓[2, 3], m↓↓[6, 7], m↓↓[10, 11]]

#28:  pu3(1.271, 0.994)
```



Zum Schluss möchte ich noch die Anregung aus dem DNL 62, S.21ff, aufgreifen und auf das letzte Beispiel anwenden. Im DNL wird beschrieben, wie man mit einem Schieber aus einem Vektor von Punktkoordinatenpaaren ein Koordinatenpaar dergestalt selektiert, dass DERIVE veranlasst wird, den entsprechenden Punkt in einem 2D-Diagramm zu aktualisieren. Das Vorgehen führt zu dem Eindruck als bewege sich der Punkt, wenn der Schieber bewegt wird. Um die drei oben genannten. Körper zu animieren, zeichnet man den Ausdruck #30. Zuvor muss im 2D-Graphikfenster ein Schieber für den Parameter t mit Bereich $1 \le t \le 159$, Schrittzahl 158, eingefügt sein. Wichtige Einstellung: Punkte nicht verbinden. Vor dem Hintergrund einer bereits gezeichneten Bahn sollte man eine andere Punktfarbe und große Punkte wählen. Dieser „Trick" lässt sich natürlich auch bei allen vorigen Beispielen einsetzen.

Finally I'd like to come back to the suggestion given in DNL 62, pp 21, and apply it to the last example. There is described how to use a slider for selecting a pair of coordinates from a vector of such pairs that DERIVE is induced to move the respective point in a 2D-plot. This leads to the impression as if the point is moving when moving the slider. For animating the three bodies from above one has to plot expression #30 after having introduced a slider bar for parameter t with $1 \le t \le 159$, 158 intervals with point size large and points not connected. It is recommended to plot the orbit and the points in different colours. Of course, this "trick" can be used for all examples above, too.

```
#29:  mv(a, t) := [VECTOR(a_{1,t,i}, i, 2), VECTOR(a_{2,t,i}, i, 2), VECTOR(a_{3,t,i}, i, 2)]

#30:  mv(pu3(1.271, 0.994), t)
```



5. Schlussbemerkung

Im Mathematik-Unterricht des Gymnasiums (im Bundesland Bayern, wo ich unterrichte) werden Runge-Kutta-Verfahren kaum Verwendung finden und zwar schon deshalb nicht, weil Differentialgleichungen im Lehrplan nicht erscheinen. Anders ist es in Physik. Der neue Lehrplan der 10. Klasse sieht vor, konkrete eindimensionale Anfangswertprobleme durchzurechnen. Ein Beispiel ist die senkrecht beschleunigte Rakete, die bei konstanter Schubkraft zeitlinear Masse verliert. Zur Berechnung wird das Eulerverfahren verwendet, das tatsächlich ein Runge-Kutta-Verfahren darstellt, wenn auch das primitivste. Auf diese Vorbereitung aufbauend ist es denkbar, dass Schüler in einem nachfolgenden Seminar mit Hilfe eines Fertigbausteins, wie RK6 einer ist, sich weitere, auch kompliziertere Bewegungsaufgaben erschließen.

Für Lehrer können RK-Verfahren Hilfsmittel zum Erstellen spezifischer Simulationssoftware sein. Oft findet man nicht die Software, die der eigenen Methode entspricht. Dann ist ein CAS mit angeschlossener graphischer Darstellung der Rechenergebnisse nicht selten ein geeignetes Instrument. Mit RK6, und besonders RK6A, ist nun auch DERIVE in der Lage Abläufe zu simulieren, denen anspruchsvolle Anfangswertprobleme zugrunde liegen.

5. Final remark

In gymnasium math teaching (in the federal state Bavaria where I am teaching) Runge-Kutta-methods are hardly used. The reason is easy to find: differential equations do not appear in the curriculum. Things are quite different in physics. The new curriculum for the 10th form provides calculating practical one dimensional initial value problems. One example is the vertical accelerated rocket with constant thrust losing mass linear wrt time. The Euler-method – which in fact is the most elementary Runge-Kutta-method – is used for calculation. Based on this preparation one can imagine that students in a following seminar supported by a ready made “Black Box“ – like RK6 – are able to make more complicated motion problems accessible for themselves.

Teachers can use RK-methods as tools for creating specific simulation software. It occurs often that one cannot find the software which meets one’s own teaching method. Then a CAS together with integrated plotting facilities is in many cases an appropriate tool. With RK6, and especially with RK6A, DERIVE is now also able to simulate processes based on ambitious initial value problems.

6. Anhang

`RK6A(u,y,y0,xend,h0,sel,tol)` hat folgende Parameter: *u*, *y*, *y*0 wie bei `RK`. *xend* ist das Ende der Integration. Integriert wird von `y0↓1` bis *xend* (und ein bisschen darüber hinaus). *h*0 ist die Schrittweite des 1. Schritts. Der Wert ist unkritisch. Die eingebaute Schrittweitensteuerung sorgt für die Anpassung an richtige Werte. *sel* ist ein Vektor, dessen Elemente Indizes auf den Vektor *y* sind. Ausgehend von den indizierten Elementen von *y* wird der lokale Diskretisierungsfehler abgeschätzt und aus diesem die Weite des folgenden RK-Schritts abgeleitet. Ist der Funktionswert *y* eindimensional, setzt man *sel* = [2], bei zweidimensionalem *y* bedeutet *sel* = [2,3], dass die euklidische Norm aus beiden Koordinaten verwendet wird. *tol* ist ein Maß für die Genauigkeit der Berechnung. *tol* ist ungefähr log10 des relativen Fehlers der Fortschreitungsrichtung. Sinnvolle Werte liegen zwischen -6 und -16. Noch kleinere Werte sind nicht sinnvoll, es machen sich dann Rundungsfehler des Algorithmus bemerkbar. Bei größeren Werten kommt der Algorithmus (abhängig vom Anfangswertproblem) ins „Stottern", es müssen viele Einzelschritte zurückgenommen und mit kleinerer Schrittweite wiederholt werden. Empfohlen wird *tol* = -10 für erste Untersuchungen eines Anfangswertproblems. Die Variable *hprot* dient dazu, die Schrittweitensteuerung zu überprüfen. *hprot* ist ein Protokoll der Korrekturfaktoren der Schrittweitensteuerung. Die Werte sollten nahe bei 1 liegen. Werte kleiner als 0,8 bedeuten, dass Schritte verworfen werden mussten. Kommt das häufig vor, sollte *tol* kleiner gewählt werden.

6. Appendix

`RK6A(u,y,y0,xend,h0,sel,tol)` uses the following parameters: *u*, *y*, *y*0 have the same meaning as in `RK`. *xend* is the end of integration. Integration is running from `y0↓1` to *xend* (and a little bit further). *h*0 is the step size of the first step. Its value is not critical. The built in control of the step size takes care for adaptation on appropriate values. *sel* is a vector which elements are indices on vector *y*. Starting with the indicated elements of *y* the local discretisation error is estimated and this is the base for setting the following RK-step. In case of a one dimensional function value set *sel* = [2], in case of a two dimensional *y* then set *sel* = [2,3] which has the meaning that the Euclidean norm of both coordinates is used. *tol* is a measure for the accuracy of the calculation which is approximately log 10 of the relative error in the direction of progress. Values between -6 and -16 make sense. Smaller values don't make sense because of rounding errors of the algorithm. Taking greater values make the algorithm "stuttering" dependent on the given IVP. Many single steps must be taken back and then repeated applying smaller steps. I recommend *tol* = -10 for the first investigations of an IVP. Variable *hprot* serves testing the step size control. It is a report of the correcting factors of the step size control. Its values should lie close to 1. Values less 0.8 say that the steps had to be rejected. If this occurs often then you should choose a smaller value for *tol*.

Comment of the editor

I left Heinrich Ludwig´s contribution in its original German version to the convenience of the many German speaking members of the DUG. For our English speaking friends I translated it into English. So don't blame Heinrich for any mistakes in the English version.

Josef

Runge-Kutta – Unveiled

Josef Böhm, Würmla, Austria

When I read Heinrich Ludwig´s great contribution on the Runge-Kutta-Method I found myself inspired enough to inform (again after many years) about this standard method solving DEs and systems of DEs numerically – and how this method has been treated by DERIVE (and other tools) so far. This is a short summary of the procedure (which can be found in many textbooks):

$$\frac{du_1}{dt} = f_1(t, u_1, u_2, \ldots, u_m)$$
$$\frac{du_2}{dt} = f_2(t, u_1, u_2, \ldots, u_m)$$
$$\vdots$$
$$\frac{du_m}{dt} = f_2(t, u_1, u_2, \ldots, u_m)$$

for $a \le t \le b$ with initial values $u_1(a) = \alpha_1, u_2(a) = \alpha_2, \ldots, u_m(a) = \alpha_m$.

The classic RK-method of order 4 is given by

$$y' = f(t, y),\ \ a \le t \le b,\ y(a) = \alpha$$

$$w_0 = \alpha$$
$$k_1 = h\, f(t_i, w_i)$$
$$k_2 = h\, f\left(t_i + \frac{h}{2}, w_i + \frac{k_1}{2}\right)$$
$$k_3 = h\, f\left(t_i + \frac{h}{2}, w_i + \frac{k_2}{2}\right)$$
$$k_4 = h\ f(t_i + h + w_i + k_3)$$

then $w_{i+1} = w_i + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$

for $i = 0, 1, \ldots, N-1$ with $h = \frac{b-a}{N}$.

Heinrich Ludwig refers in his article to DERIVE´s built in RK-function (contained in the Utility file `ODEApproximation.mth`, which replaced `ODE_APPR.MTH` from earlier DERIVE versions. DERIVE changed and improved from the DOS-versions via Dfw4 and DERIVE 5 to DERIVE 6 offering now its great programming features – but the RK-routine did not change. It still consists of some auxiliary functions which are not easy to follow for students because they call themselves applying the `ITERATES`-command in the last step.

It might be difficult for students and for not experienced DERIVE-users to discover the important steps of the RK-method more or less hidden in the DERIVE code. As you might know from other occasions the `ITERATES`-command often slows down calculation. So I had the idea to use the programming features to rewrite the “classic” RK-Method.

See first the original RK-procedure how it is given in DERIVE´s Utility file:

```
#1:   [PrecisionDigits := 12, NotationDigits := 8]

                                            c1 + (   lim    p) + 2·(c2 + c3)
                                                 v→u_ + c3
#2:   RK_AUX3(p, v, u_, c1, c2, c3) := ──────────────────────────────────
                                                      6

      RK_AUX2(p, v, u_, c1, c2) := RK_AUX3(p, v, u_, c1, c2,      lim      p)
#3:                                                          v→u_ + c2/2

      RK_AUX1(p, v, u_, c1) := RK_AUX2(p, v, u_, c1,      lim      p)
#4:                                                  v→u_ + c1/2

      RK_AUX0(p, v, v0, n) := ITERATES(u_ + RK_AUX1(p, v, u_, lim  p), u_, v0, n)
#5:                                                           v→u_

#6:   RK(r, v, v0, h, n) := RK_AUX0(h·APPEND([1], r), v, v0, n)

#7:   RK([x·(1 - y), 0.3·y·(x - 1)], [t, x, y], [0, 2, 1], 0.1, 10)
```

And here is my program which does without any auxiliary function and without any ITERATES.

```
rk4(r, v, v0, h, n, i, k1, k2, k3, k4, tbl) :=
  Prog
    tbl := [v0]
    i := 1
    Loop
      If i > n
         RETURN tbl
      k1 := VECTOR(SUBST(k, v, v0), k, r)
      k2 := SUBST(r, v, ADJOIN(v0↓1 + h/2, REST(v0) + h/2·k1))
      k3 := SUBST(r, v, ADJOIN(v0↓1 + h/2, REST(v0) + h/2·k2))
      k4 := SUBST(r, v, ADJOIN(v0↓1 + h, REST(v0) + h·k3))
      v0 := ADJOIN(v0↓1 + h, REST(v0) + h/6·[1, 2, 2, 1]·[k1, k2, k3, k4])
      tbl := APPEND(tbl, [v0])
      i :+ 1
```

The parameter list for entering the function remains the same.

I tested my program on several examples. The next one is an also "classic" prey-predator-system (one predator- and two prey populations). This example was provided by Josef Lechner (some years ago and based on a book on Systems Dynamic by Bossel, 1994). I will come back again to this model later.

Compare the calculation times. (I select the last row of the result matrix to demonstrate that both functions are giving the same results.)

```
       (RK([0.1·x - 0.1·x·z, 0.12·y - 0.1·y·z, 0.1·x·z + 0.1·y·z - 0.1·z], [t, x, y,
#19:
         z], [0, 1, 1, 1], 1, 200))
                                   201

#20:   [200, 0.010246976, 0.55946290, 1.9882390]
```

needs 0.61 sec.

```
       (rk4([0.1·x - 0.1·x·z, 0.12·y - 0.1·y·z, 0.1·x·z + 0.1·y·z - 0.1·z], [t, x, y,
#21:
         z], [0, 1, 1, 1], 1, 200))
                                   201

#22:   [200, 0.010246976, 0.55946290, 1.9882390]
```

needs 0.31 sec.

In this case RK6 from Heinrich Ludwig does not give much better results – but calculation time increases up to 2.4 sec : `[200, 0.010246757, 0.55945398, 1.9882254]`.

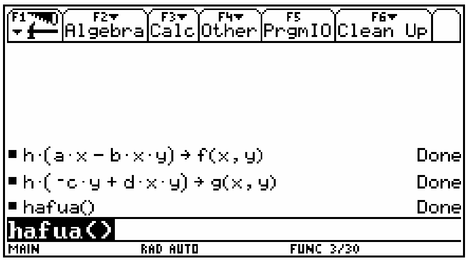
I was satisfied with my program – but not satisfied enough to change the subject. I thought “And what about the TIs?” I browsed in my collection of papers and what did I find? A letter from our DUG-member from the very first days, Richard Schorn. In 1996 he sent a diskette with a TI-92 program for treating the prey-predator-model using the “classic” Runge-Kutta-method. In his letter Richard reminds on his comment in DNL#5 where he gave a numerical DERIVE solution applying the RK-function from the utility file. This is the header of Richard’s letter from July 1996:

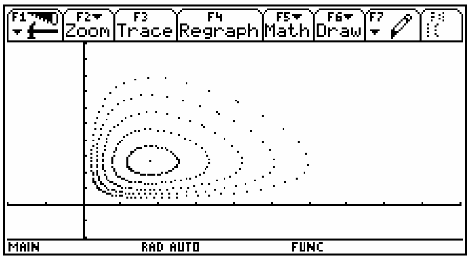
OStD i.R. Richard Schorn
Am Bleichanger 10
D-87600 Kaufbeuren

Kaufbeuren, 13. Juli 1996

As you can see, (almost) nothing gets lost in my stack of papers. Sometimes I forget one or the other of my treasures provided by the DUG-members but then suddenly they appear and ask for being published.

```
hafua()
Prgm
Local  k1,k2,k3,k4,l1,l2,l3,l4
Ø.Ø4→a
Ø.ØØ15→b
Ø.Ø3→c
Ø.ØØ17→d
1→h
ClrGraph:ClrDraw:FnOff
⁻2Ø→xmin:1ØØ→xmax:1Ø→xscl
⁻2Ø→ymin:1ØØ→ymax:1Ø→yscl:2→xres
setGraph("Axes","On")
c/d→xs:a/b→ys
PtOn  xs,ys
For  k,1,5
Ø→t:xs+5→xØ:ys+7→yØ:⁻1→i:xØ→xs:yØ→ys
6→dd
While  dd>2
i+1→i
abs(xs-xØ)+abs(ys-yØ)→dd
If  i<2Ø:6→dd
f(xØ,yØ)→k1:g(xØ,yØ)→l1
f(xØ+k1/2,yØ+l1/2)→k2:g(xØ+k1/2,yØ+l1/2)→l2
f(xØ+k2/2,yØ+l2/2)→k3:g(xØ+k2/2,yØ+l2/2)→l3
f(xØ+k3,yØ+l3)→k4:g(xØ+k3,yØ+l3)→l4
xØ+(k1+2*k2+2*k3+k4)/6→xØ
yØ+(l1+2*l2+2*l3+l4)/6→yØ
t+h→t
If  mod(i,3)=Ø:PtOn  xØ,yØ
EndWhile
EndFor
EndPrgm
```





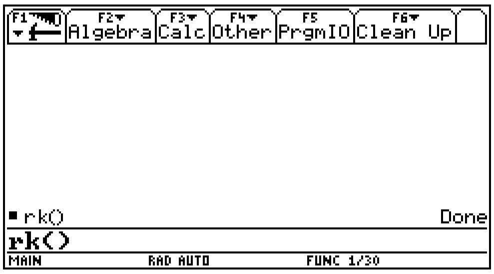
The coefficients and the initial values are given within the program which produces a family of phase diagrams.

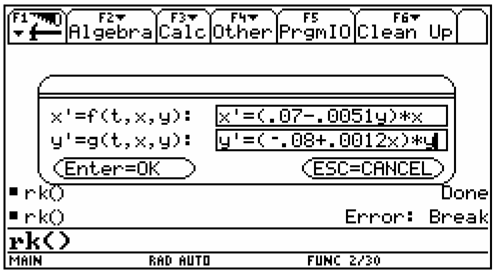
Changing for plotting a-t-*x* or t-*y* diagram needs some change in the program code. The “ordinary” user will in most cases not be able to perform the appropriate changes.
The problem given in DNL#5 was:

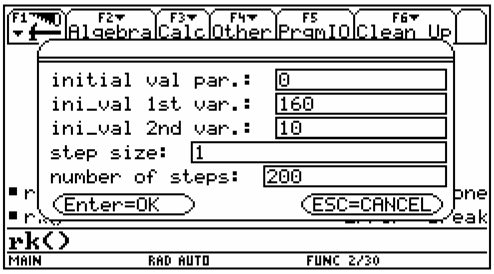
```
RK([(0.07–0.0051y)x,(–0.08+0.0012x)y],[t,x,y],[0,160,10],1,200)
```

If you want to solve it using `hafua()` then you have to edit the program. This might be too complicated for some users. On the other hand, in 1996 the TI-92 was in its childhood and programming on the TI was really a challenge. Thanks Richard for facing this challenge and Hello from Würmla.

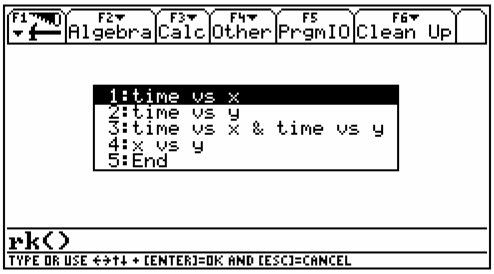
Now I felt another challenge: producing a more user friendly program. Program `rk()` is the result of my efforts:

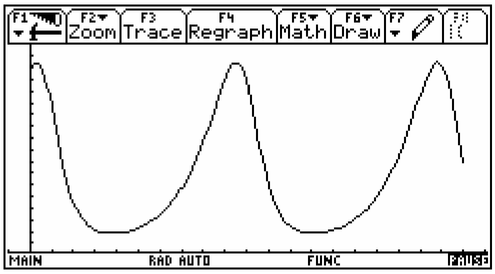




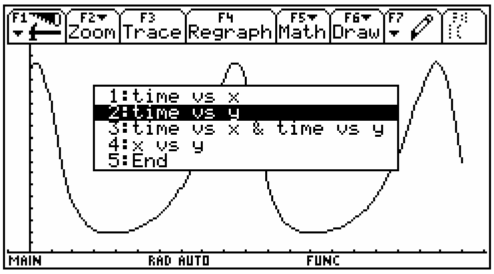


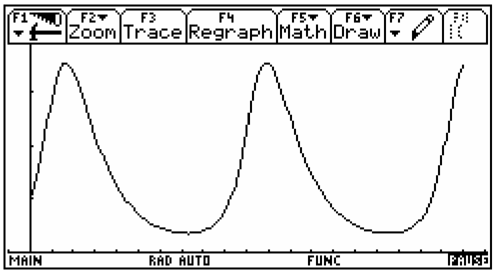
The device is busy some time … and then you are presented a pop up window to make your choice: All settings (WINDOW and PLOTS are set automatically by the program!)



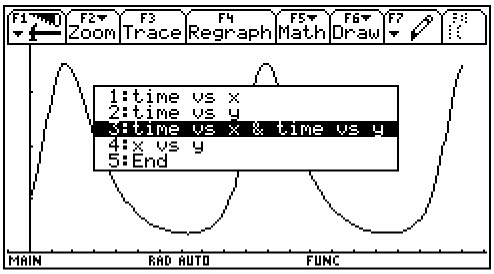


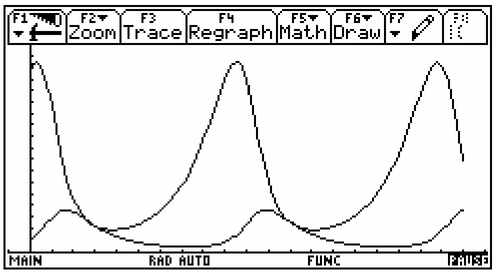
Press ENTER and the Pop Up Window will open again to receive your choice:

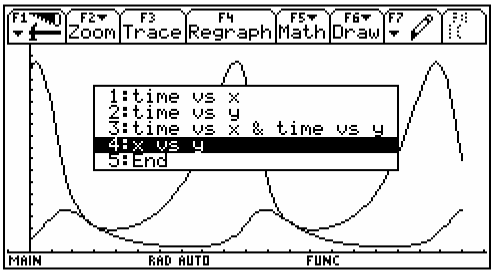


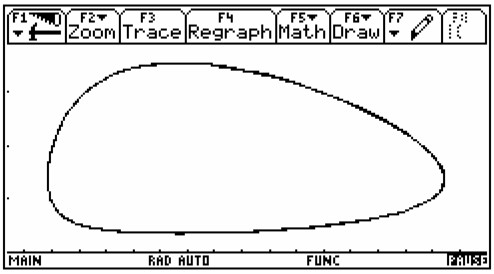


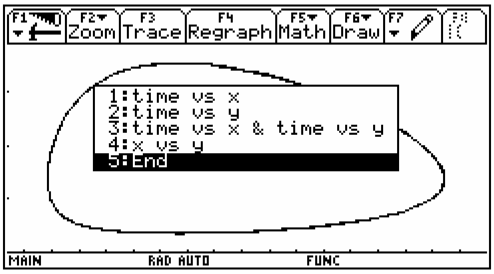
It is quite better to have both time vs – graphs on the same axes because of different scaling. (You can interrupt the program by pressing the ON-key and then trace the plot.) But let's have both populations in the same graph, and press the third option which shows both diagrams on the same axes:

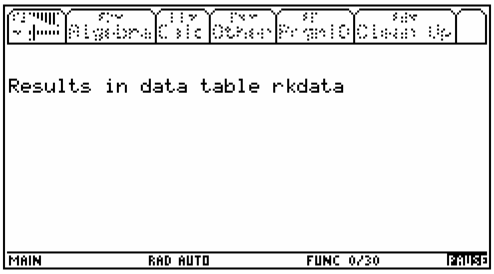


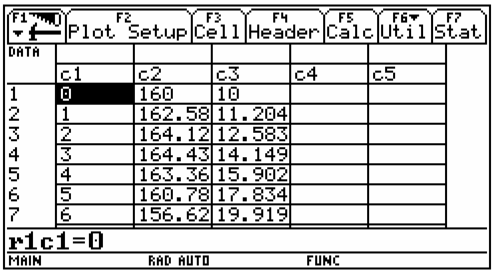












DATA	c1	c2	c3	c4	c5
1	0	160	10		
2	1	162.58	11.204		
3	2	164.12	12.583		
4	3	164.43	14.149		
5	4	163.36	15.902		
6	5	160.78	17.834		
7	6	156.62	19.919		

r1c1=0

The last option gives the phase diagram.

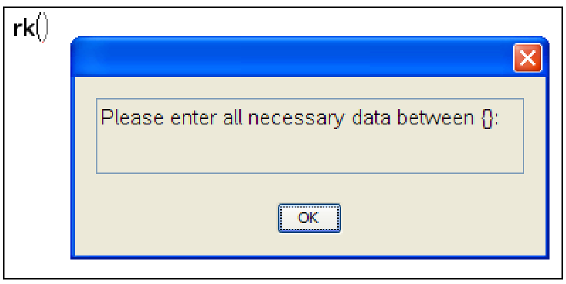
The values for t, $x(t)$ and $y(t)$ can be found in the data sheet `rkdata`.

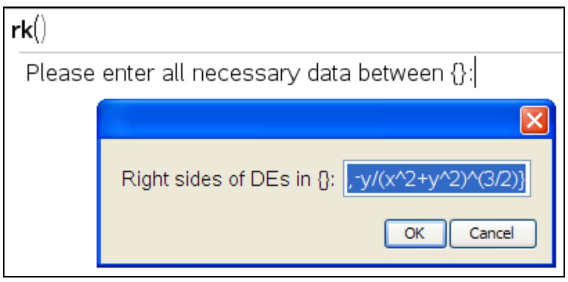
I don't want to bore you by printing the program. You can find it among the files provided for downloading on our website.

This was quite nice but I was not completely satisfied with myself. RK, RK4 and as well are much more flexible because they are not restricted to two equations only. Instead of improving my TI-92 / Voyage 200 program I downloaded the latest version of TI-Nspire (Version 2) and tried to write a program.

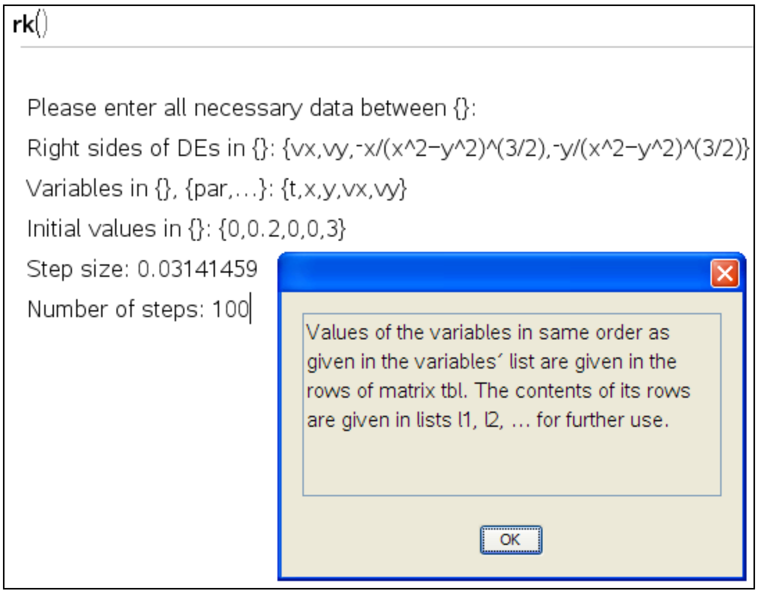
The TINspire Challenge:

I am starting with presenting some screen shots – treating one of Heinrich Ludwig's examples with four equations (see page 11):





What you will notice first is that something like a report containing all the input is presented. I'll show how to do this. (It is a new feature of Nspire version 2).



I enter all data as requested between {}. In this case it would be more comfortable to predefine the equations in the calculator eg eqs:={vx, vy, ..} because here you can use the tablets. In the dialogue boxes you cannot. (This should be improved!)
The final result is a matrix *tbl* with the rows giving the approximated values of the variables appearing in the order of the 2nd input. For plotting the diagrams it is necessary to have lists of values. The program does this job, too. It converts the rows of the matrix to lists.

tbl

0	0.03141459	0.06282918	0.09424377	0.12565836	0.15707295
0.20000000	0.18807955	0.15628744	0.11240027	0.06264693	0.01067245
0	0.09237391	0.17543537	0.24552034	0.30338625	0.35116325
0	-0.73515619	-1.24496662	-1.51593299	-1.63276989	-1.66644438
3	2.82894771	2.44134419	2.02637756	1.66961718	1.38317539

l1

{0,0.03141459,0.06282918,0.09424377,0.12565836,0.15707295,0.18848754,0.21990▸

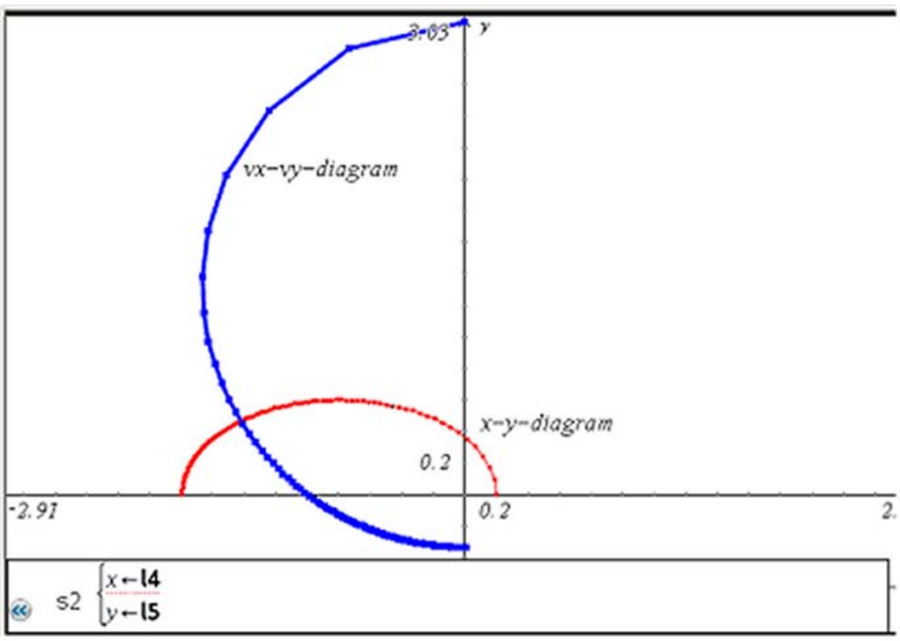
l3

{0,0.09237391,0.17543537,0.24552034,0.30338625,0.35116325,0.39091079,0.42427▸

l5

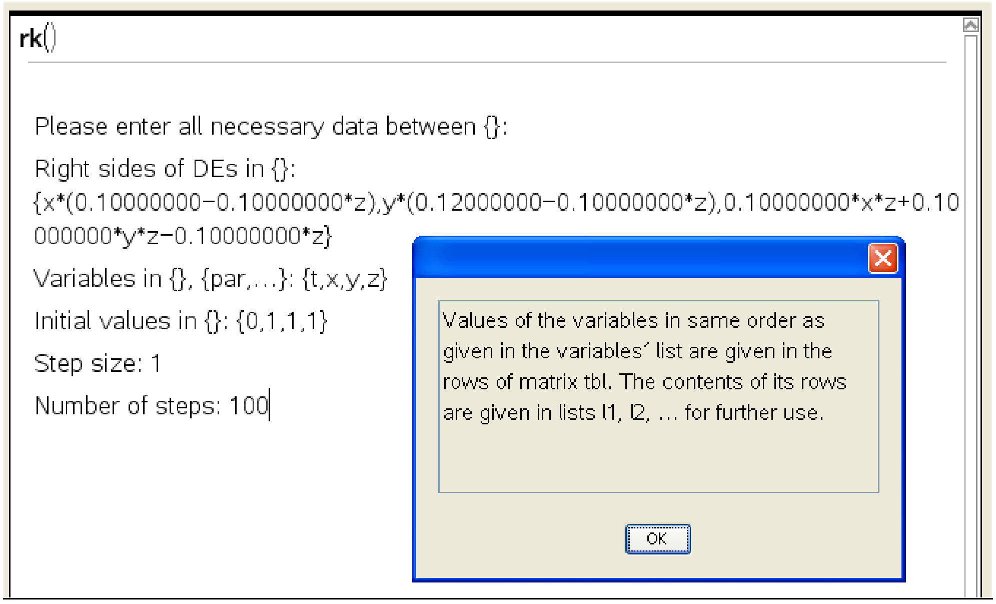
{3,2.82894771,2.44134419,2.02637756,1.66961718,1.38317539,1.15607553,0.97465▸

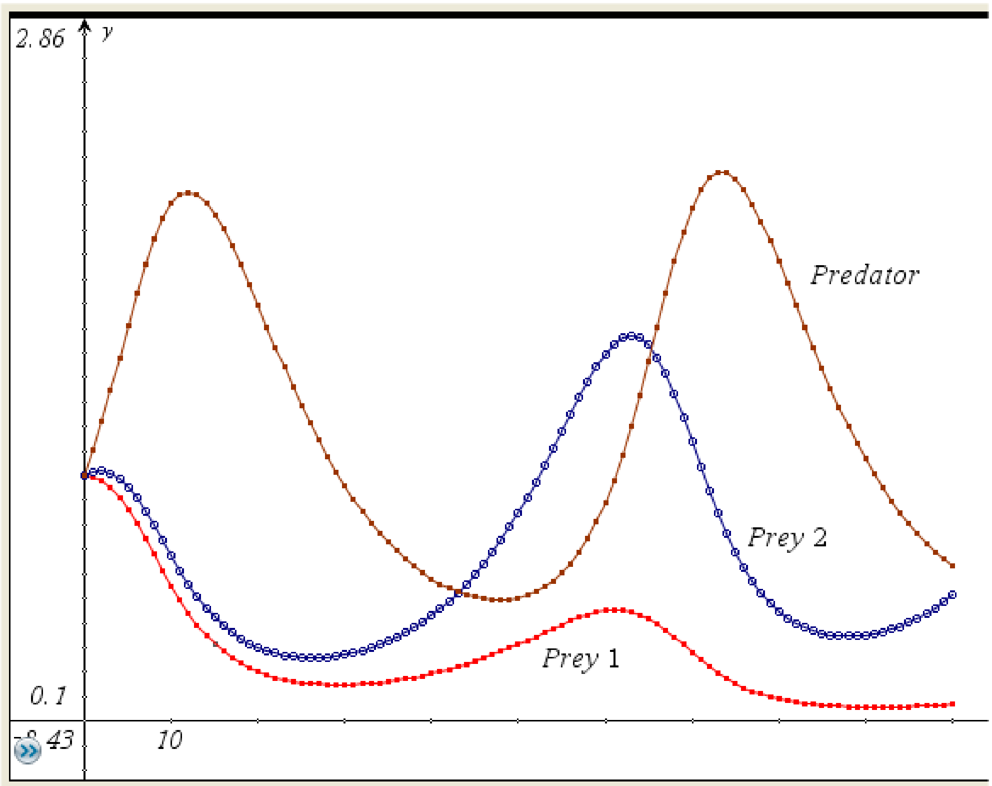
Now you can produce the scatter plot and choose the attributes of your choice.



You may compare this figure with the DERIVE plot from page 12.

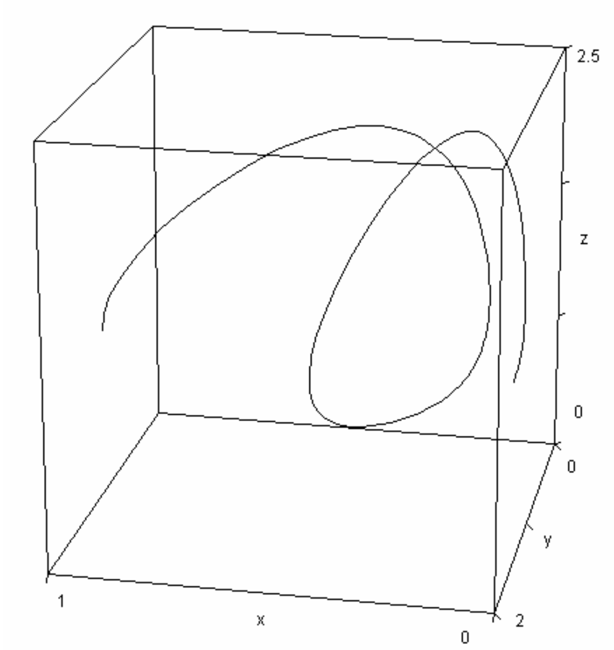
Before showing the program I will present another nice example (provided by Josef Lechner a couple of years ago). Josef was and still is an expert in dynamic systems. The next screen shows the report of my input including text and dialog boxes as well. There are dialog-boxes with a text and a field to enter the requested input. The first line (“Please …”) is a “text box”. Both features are new!! Explanations are given within the program code (page 32). The TI-Nspire program is running now.



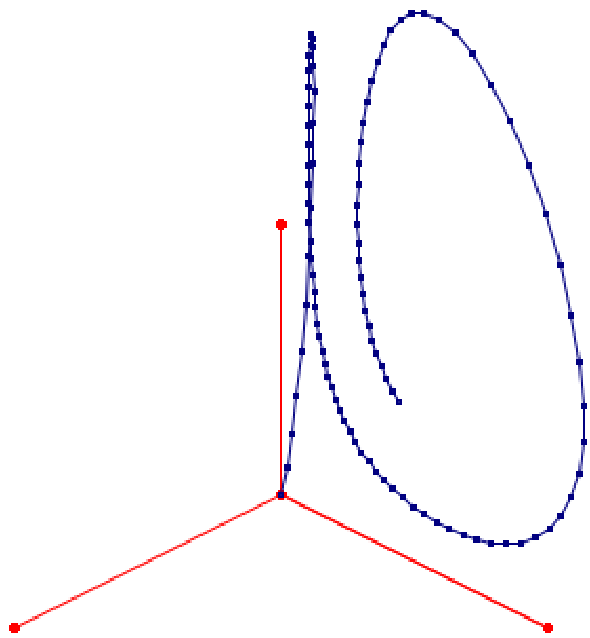


This is a nice coloured plot of the development of the three populations for the first 100 time steps. You can see three scatter plots on the same axes using lists l1 (time), l2 (prey 1), l3 (prey 2) and l4 (predator).

What we cannot do with TI-Nspire but what we can do with DERIVE is producing a 3D-phase diagram of all populations!



Believe it or not, the following is the graph of the *x*-*y*-*z*-phase diagram presented on the Nspire-CAS screen! I produced an isometric projection of the object onto the *x*-*y*-plane multiplying the matrix consisting of the three columns for *x*, *y* and *z* with the respective projection matrix (isometric projection), converted the resulting columns again into lists and plotted them as a scatter plot.



This is the TI-Nspire program. Comments are the lines starting with ©. There was one problem which made the program a lot bulkier than the DERIVE program: I miss my favourite DERIVE command, the powerful VECTOR-command which must be substituted by For−EndFor-loops.

```
Define rk()=
Prgm
:Local k1,k2,k3,k4,I,j,rv0,df,dv
:© The text and dialog boxes:
:Text "Please enter all necessary data between {}:"
:Request "Right sides of Des in {}:",f
:Request "Variables in {}, {par,...}:",v
:Request "Initial values in {}:",v0
:Request "Step size:",h
:Request "Number of steps:",n
:df:=dim(f):dv:=dim(v)
:tbl:=newMat(n+1,dv)
:tbl[1]:=list▸mat(v0)
:i:=1
:© The big loop for the n steps:
:While i≤n
:rv0:=seq(v0[i],I,2,dv)
:k1:=f: k2:=f:k3:=f:k4:=f
:© The 4 loops for calculating the auxiliary values
:© replacing the VECTOR−command in DERIVE
:For j,1,dv
:k1:=lim(k1,v[j],v0[j])
:EndFor
:v2:=augment({v0[1]+((h)/(2))},rv0+((h)/(2))*k1)
:For j,1,dv
:k2:=lim(k2,v[j],v2[j])
:EndFor
:v3:=augment({v0[1]+((h)/(2))},rv0+((h)/(2))*k2)
:For j,1,dv
:k3:=lim(k3,v[j],v3[j])
:EndFor
:v4:=augment({v0[1]+h},rv0+h*k3)
:For j,1,dv
:k4:=lim(k4,v[j],v4[j])
:EndFor
:© Calculating the next values and filling the next row in the matrix
:v0:=augment({v0[1]+h},rv0+((h)/(6))*(k1+2*k2+2*k3+k4))
:tbl[i+1]:=list▸mat(v0)
:i:=i+1
:EndWhile
:tbl:=tblᵀ
:Text "Values of the variables in same order as given in the variables´ list are given in the rows of matrix tbl. The contents of its rows are given in lists l1, l2, ... for further use."
:© Using indirection to dynamically create the lists according to the number of variables
:For I,1,dim(tbl)[1]
:mat▸list(tbl[i])→#("l"&string(i))
:EndFor
:EndPrgm
```

I can imagine that one does not intend to include programming into his mathematics course on one hand but that he wants to "unveil" the Runge-Kutta-Method as an improvement of the Euler Methods on the other hand because he/she is not satisfied with only using the Black Box RK/RK4 or RK6/RK6A or rk(). Spreadsheets or programs with spreadsheet features implemented like Nspire or GeoGebra are excellent tools for systems of two equations. In GeoGebra in its recent version it is not so comfortable because functions of two variables cannot be defined. I'd like to present the TI-Nspire realization combined with a nice application for the sliders to study the influence of the parameters and included with a moveable initial point. So finally we will have a very flexible model.

This is a realization of the **Prey–Predator**–dynamic system.

It is decribed by the system of differential equations:

x' = a x − b x y = f(x,y) … x = number of prey animals
y' = −c y + d x y = g(x,y) … y = number of predators

The numeric solution is calculated using the **Runge–Kutta–procedure** with h = 0.1.

All parameters a, b, c, d of the equations and the initial numbers of prey and predators can be controlled by sliders in the Graphs & Geometry Application.

We show the phase diagram prey vs predators.

The diagrams time vs pray and time vs predator could be added.

In the Lists & Spreadsheet Application one can follow the Runge–Kutta–Approximation.

$f(x,y):=a\cdot x-b\cdot x\cdot y$	*Done*
$g(x,y):=-c\cdot y+d\cdot x\cdot y$	*Done*
$h:=0.1$	0.10000000

First of all we define the two equations in the Calculator. Then we switch to the spreadsheet:

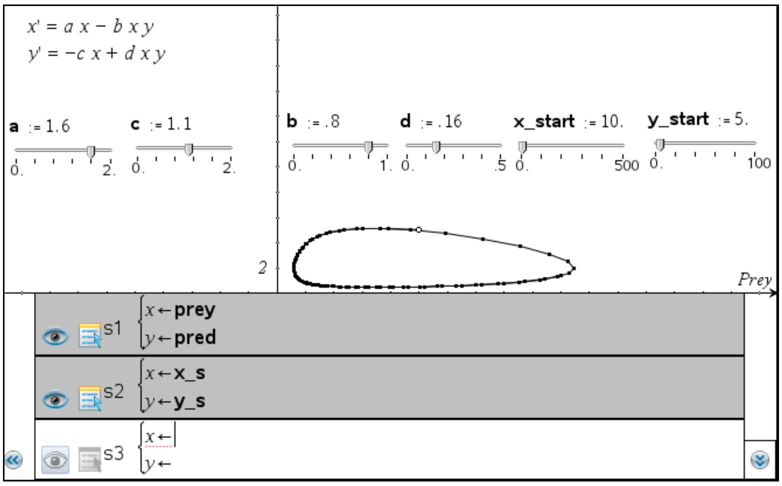
	A time	B	C	D prey	E pred	F f_1	G g_1	H f_2	I g_2
◆									
1	0	x0	**10.0…**	10.0000…	20.00…	0	0	0	0
2	0.10000…	y0	**20.0…**	9.02403…	20.50…	-10.000…	6.00000…	-9.78500…	5.07500…
3	0.20000…			8.10958…	20.83…	-9.4823…	4.15086…	-9.16160…	3.21072…
4	0.30000…			7.27091…	20.97…	-8.7830…	2.31131…	-8.39611…	1.40428…
5	0.40000…			6.51518…	20.94…	-7.9786…	0.56819…	-7.56041…	-0.2688…
6	0.50000…			5.84384…	20.77…	-7.1337…	-1.01566…	-6.71196…	-1.7586…
7	0.60000…			5.25418…	20.47…	-6.2978…	-2.40213…	-5.89206…	-3.0387…
8	0.70000…			4.74086…	20.06…	-5.5044…	-3.57476…	-5.12708…	-4.1021…
9	0.80000…			4.29705…	19.57…	-4.7735…	-4.53386…	-4.43120…	-4.9562…
10	0.90000…			3.91539…	19.01…	-4.1149…	-5.29132…	-3.80966…	-5.6171…
11	1.00000…			3.58858…	18.40…	-3.5304…	-5.86594…	-3.26163…	-6.1059…
12	1.10000…			3.30972…	17.76…	-3.0174…	-6.27985…	-2.78260…	-6.4457…
13	1.20000…			3.07258…	17.10…	-2.5701…	-6.55595…	-2.36609…	-6.6590…

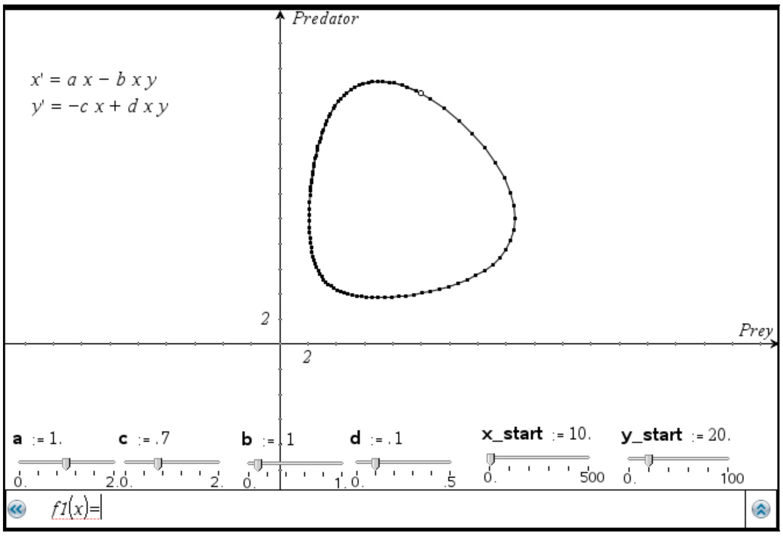
D2 $=d1+\frac{h}{6}\cdot(f2+2\cdot h2+2\cdot j2+l2)$

$F2: f(d1,e1),\ H2: f\left(d1+\frac{h}{2}\cdot f2, e1+\frac{h}{2}\cdot g2\right),\ J2: f\left(d1+\frac{h}{2}\cdot h2, e1+\frac{h}{2}\cdot i2\right),\ L2: f(d1+h\cdot j2, e1+h\cdot k2)$

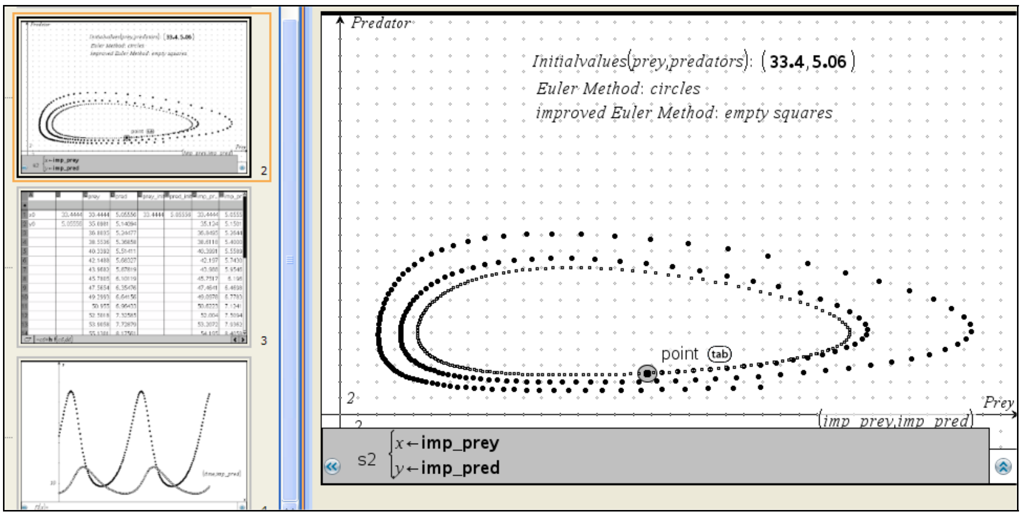
In cells *E2, G2, I2,K2* and *M2* replace *f* by *g*. Then copy down row #2.

Cells *c1* and *c2* are linked with the coordinates of the initial point.





The last realisation shows another form of animation: You can grab and drag the initial point and demonstrate the difference between Euler Method and Improved Euler Method. In a second plot window you can show the time vs population diagrams. Grabbing the point could be implemented in the RK-model, too (instead of the sliders for its coordinates).



References: J.F. Faires, R.L. Burden: *Numerische Methoden*, Spektrum, 1994
Steven Schonefeld, *Numerical Analysis via Derive*, MathWare, 1994
C.H. Edwards, D.E. Penney: *Differential Equations*, Prentice Hall, 1996

The Octahedron of Horror

Wolfgang Alvermann, Emden, Germany
(translated by J. Böhm)

An octahedron is a regular polyhedron. Its surface is formed by eight congruent equilateral triangles. Each octahedron can be inscribed in a cube such that the vertices of the octahedron are lying in the centre points of the faces of the cube.

The octahedron $ABCDS_1S_2$ presented in figure 1 (*Abbildung 1*) is given by its vertices $A(13 \mid -5 \mid 3)$, $B(11 \mid 3 \mid 1)$, $C(5 \mid 3 \mid 7)$ and $S_1(13 \mid 1 \mid 9)$. This octahedron is inscribed a cube with vertices P_1 through P_8 as explained above.

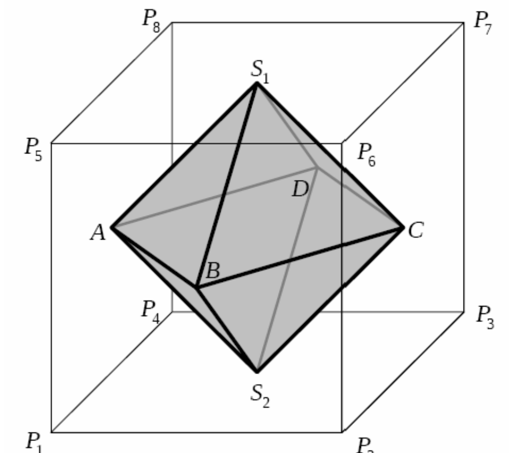


Abbildung 1

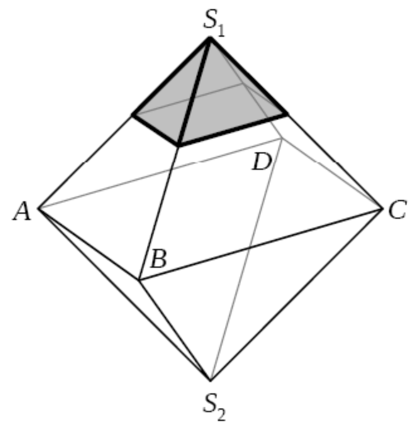


Figure 2

a) The distance between two parallel faces of the octahedron is called the "Thickness of the body". Calculate the thickness of the given octahedron as distance between point C and plane ABS_1.

b) Find the coordinates of the vertices P_6 and P_8 of the cube given in figure 1.

c) The midpoint of segment AB is M_{AB}, midpoint of segment CD is M_{CD}. Let g the straight line connecting the two midpoints. The octahedron is rotated around g as rotation axis in such a way that point $A(13|-5|3)$ is moved to a new position $A'(12+2\sqrt{2} \mid -1+\sqrt{2} \mid 2+2\sqrt{2})$.

   Show that the respective rotation angle is $\alpha = 90°$.
   What are the coordinates of point B' as new position of point B after rotating the cube?

d) A family of planes E_a: $2x_1 + x_2 + 2x_3 + 9\cdot(2a-5) = 0$, $a \in \mathbb{R}$, is given; let h a straight line passing S_1 and $S_2(5 \mid -3 \mid 1)$. Show that each plane E_a of the family is perpendicular wrt to line h. Find the intersection point P_a of plane E_a with line h.

   [Control: $P_a(13 - 4a \mid 1 - 2a \mid 9 - 4a)$]

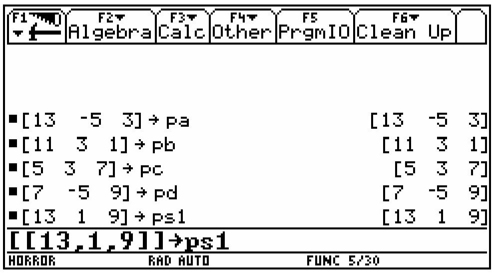
   For $0 < a \leq 1$ the plane E_a cuts off a pyramid with vertex S_1 from the octahedron (see figure 2). Find the volume V_a of this cut off pyramid.

e) Six pyramids with equal volumes V_a are cut off the octahedron in such a way that each vertex of the octahedron is the vertex of a pyramid and the base of each cut off pyramid is parallel to the opposite face of the cube (compare with task d)). A body R_a $\left(0 < a \le \frac{1}{2}\right)$ is remaining.

Explain the properties of this remaining body R_a for $a = \frac{1}{3}$ and $a = \frac{1}{2}$ with respect to the number and properties of its side faces.

Solutions

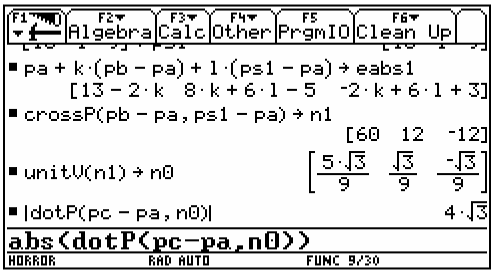
Entering the data



It is recommended entering the given points as variables before starting calculation.
ps2 will follow later.

$$\overrightarrow{pa} + \left(\overrightarrow{pc} - \overrightarrow{pb}\right) = \overrightarrow{pd}$$

Task a)

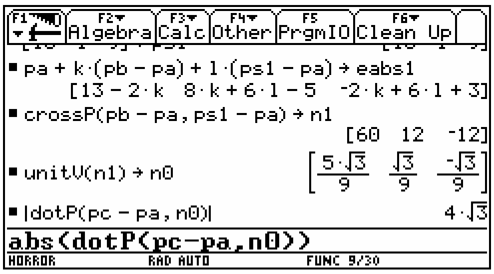


Definition of plane E_{ABS1}; calculation of the normal vector and the normal unit vector.

Formula for the distance from the formulary

$$d = \left|\left(\overrightarrow{x_1} - \overrightarrow{x_0}\right) \cdot \overrightarrow{n_0}\right|$$

Task b)



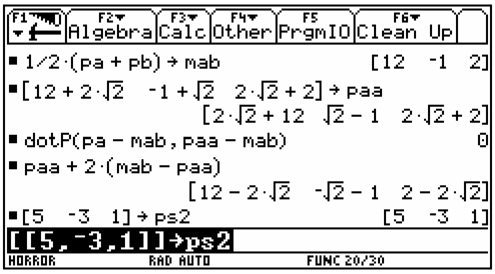
$$\overrightarrow{AB} = \overrightarrow{S_1P_6} = -\overrightarrow{S_1P_8}$$

So the edge length of the cube a = 12 because of $d = a \cdot \sqrt{2}$.

The edge length of the octahedron is $6 \cdot \sqrt{2}$.

Task c)

Point M_{AB} remains fixed during the rotation; it must be shown that the dot product of $\overrightarrow{AM_{AB}} \cdot \overrightarrow{A'M_{AB}} = 0$.



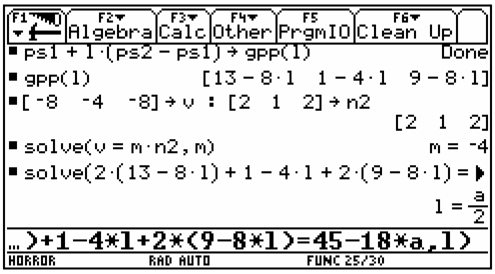
B' is calculated from

$$\overrightarrow{P_{B'}} = \overrightarrow{P_{A'}} + 2 \cdot (\overrightarrow{M_{AB}} - \overrightarrow{P_{A'}})$$

$$B' = \left(12 - 2\sqrt{2} \,|\, 1 - \sqrt{2} \,|\, 2 - 2\sqrt{2}\right)$$

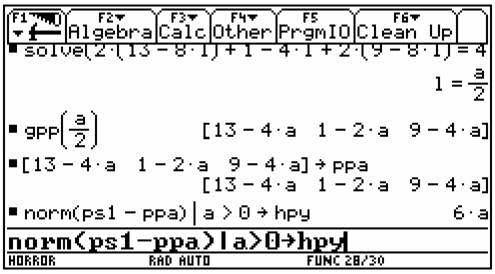
Task d)

To show: The normal vector n_2 of plane E_a and the direction vector v of straight line h are collinear; then follows that $E \perp h$.



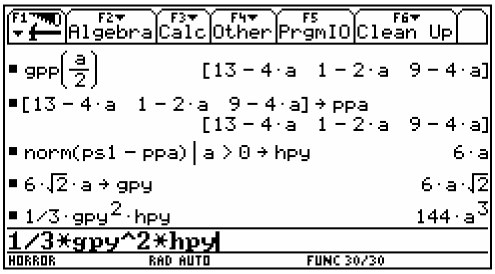
m = -4 confirms collinearity.

Substitution of *x*-, *y*- and *z*-coordinates of the line into the equation of the plane leads to $l = \frac{1}{2}$.



Substitution of $l = \frac{1}{2}$ into the equation of the line confirms the given point.

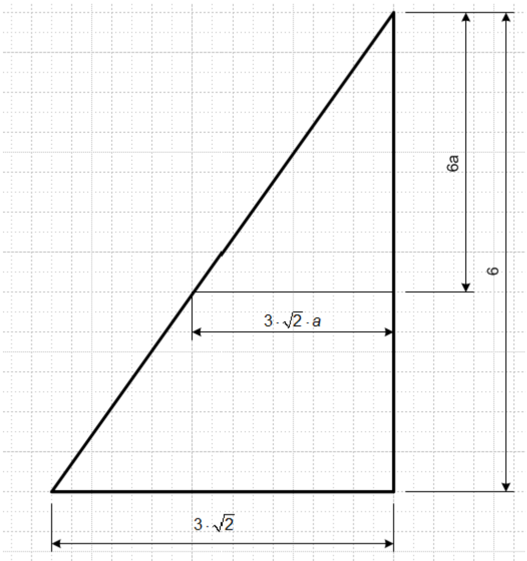
The height of the pyramid is 6*a*; calculation of its base edge is remaining.



See the sketch below:

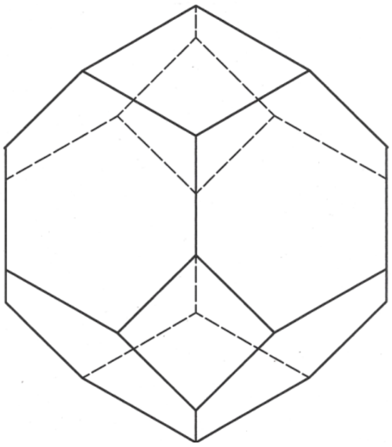
$$\frac{6}{3 \cdot \sqrt{2}} = \frac{6a}{x} \Rightarrow x = 3a \cdot \sqrt{2}$$

x is half base edge of the pyramid

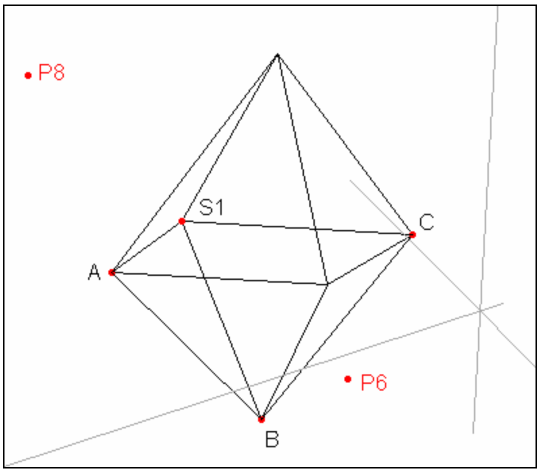


Task e)

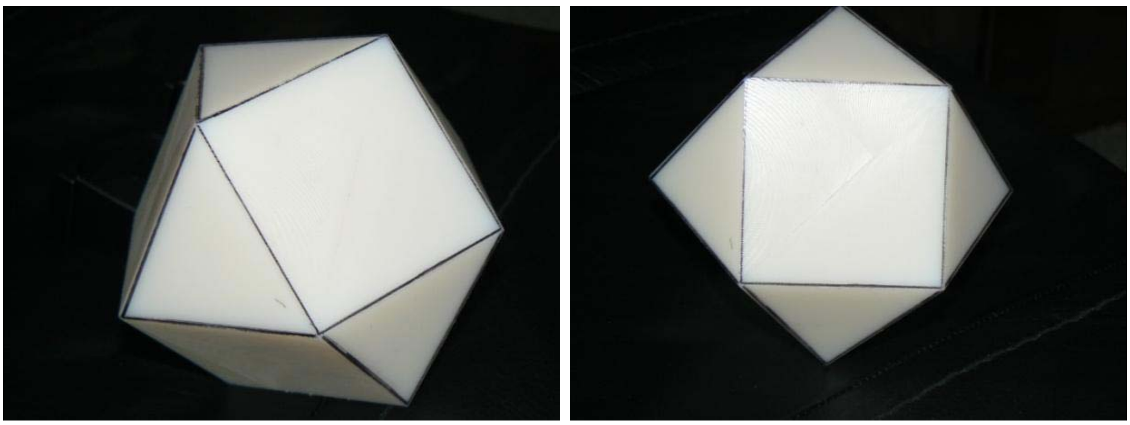
$a = \frac{1}{3} \rightarrow$ 6 squares and 8 regular hexagons



The DERIVE sketch of the given problem



$a = \frac{1}{2} \rightarrow$ 6 squares and 8 equilateral triangles



Pictures of a polyamide - model

Comments:

- Tasks a) and b) are standard problems in analytic geometry applying vector calculation.
- Application of the dot product in task c) is difficult because it demands a high degree of spatial sense which cannot be expected from average students.
- In my opinion task d) is too complex for an end examination problem (see Professor Steinberg´s comment below).
- Recognizing the remaining bodies makes problems not only for students!

Comment of Prof. Steinberg (University Oldenburg):

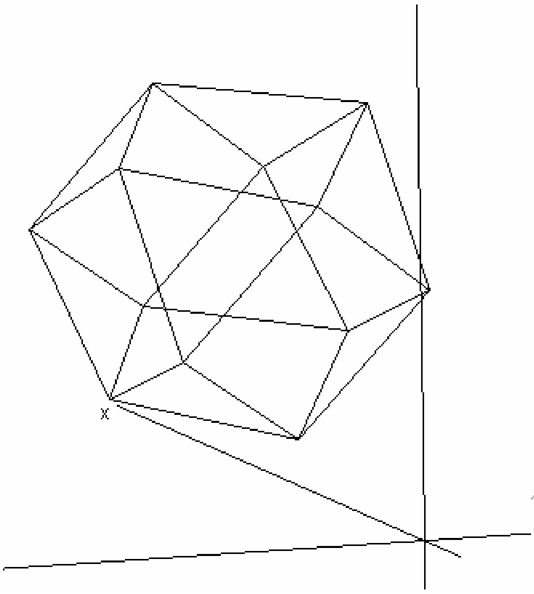
> *I treat the most beautiful and most difficult problems during my teaching, the less difficult I pose in tests and the easiest problems are appropriate for end examinations (Abitur).*

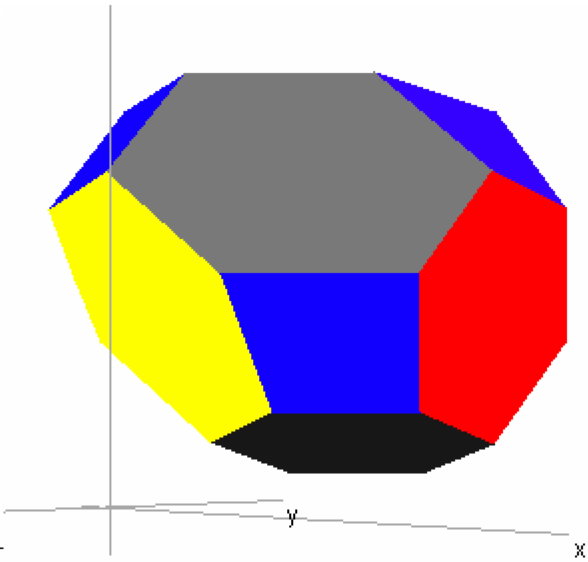
I completely share his opinion, W.A.

Reference:

http://www.spiegel.de/schulspiegel/wissen/0,1518,558559,00.html

I could not resist modelling and plotting the two remaining bodies with DERIVE. The right figure was produced applying the POLYGON_FILL-function which is provided among the utility files. Josef





About the author:

studied Mechanical Engineering and Mathematics at TU Hannover 1970 - 1975,
since 1977 teacher at Vocational Schools II Emden
(Fachgymnasium Technik, Fachoberschule Technik, Fachschule Technik)

Aim: Including real life applications in mathematics teaching

TIME 2010, MALAGA; Spain

The Invited Lectures

Bärbel Barzel, Germany

The role of CAS when learning algebra and developing functional thinking

Eugenio Roanes-Lozano,Spain

Mechanical Theorem Proving in Geometry with DERIVE using Wu's Method

Michel Beaudin, Canada

Using the Real Power of Computer Algebra

Colette Laborde, France

Tangible and dynamic mathematics

Lectures and Workshops (WS) accepted for the DERIVE and TI-Nspire strand so far

C++ as a programming language for CAS

Coding Theory for the Classroom (WS)

20 Years International DERIVE and CAS-TI user Group

Semiotic Acts and Process in the definition of Limit of a function according to Weierstrass

Semiotics Tools and Mathematics Curriculum

From Inductive Reasoning to Proof by Induction with Geometry Expressions (WS)

Mathematical Modelling with Geometry Expressions and TI-Nspire CAS

Teaching Linear Regression in the PC Lab

Solving Second Order ODEs, Two Non-analytical Methods Revisited

Using Rational Arithmetic to Develop a Proof, "*What Josef and Carl Saw*"

Beyond Newton's Law of Cooling - Updated Methods for Estimating Time Since Death

Using DERIVE®'s Graphics Tools for Geographic Profiling of Serial Offenders

Using Mathematical Tools in Forensic Investigations (WS)

Laplace Transforms, ODEs and CAS

Could it be possible to replace DERIVE by MAXIMA?

WIRIS – DERIVE, enabling compatibility

Construction of mathematical knowledge using graphic calculators (TI-84 plus & CAS) in the mathematics classroom

3D Analytic geometry in using the link between CAS and Geometry applications of the TI-NSpire (WS)

WIRIS, a complete CAS with DGS (WS)

Experiments with geometric loci

Mathematical Models in Biology (WS)

The ElGamal Public-Key Cryptosystem – A Full Implementation Using DERIVE

A virtual laboratory for blended-learning: Numerical Methods using WIRIS

Integrating a new DERIVE 6 Video User Guide into Virtual Teaching

A CAS routine for obtaining eigenfunctions for Bryan's effect

Interactive Explorations of Mathematics with TI-Nspire Technology

11 Years of Master Thesis in Engineering using DERIVE in the University of Málaga

Teaching Differential Equations and its Applications Using DERIVE 6 as a PeCas

On some images of a straight line mapped via the harmonic cross-ratio

Teaching Calculus and Numerical Analysis using CAS according to Bologna Process

E-Learning and Joomla

Using Notes with Interactive Math Boxes in TI-Nspire Software Version 2 (WS)

Exploring Rose Curves with the TI-Nspire Calculator (WS)

Lectures and Workshops (WS) accepted for the ACDCA-strand so far:

Teaching and Assessing Polygons Using Technology

Learning Math, Doing Math: Fractals and Dynamic Geometry (WS)

Enhancing Mathematical Problem Solving through the use of Dynamic Software Autograph

Arguing and Reasoning supported by Technology

Base Technologies for Tutoring

The Positive Aspects of Modeling Process in Teaching Mathematics

Sliders – A Dynamic Support for Teaching Mathematics

Physics Through GeoGebra Window

Web based education and assessment in the Bologna process

Problems and Prospects of Remote Teacher Training in Uniform Environment of E-Learning

Mathematics lessons and Classroom examples, inspired by the "Dresden Morning Post"

Software GOLUCA: Knowledge representation in Mental Calculation

WebQuest on Conic Sections as a Learning Tool for Undergraduate Students

An analysis of arguments for and against the CAS

Competence, didactic situations and Virtual Environments for Teaching and Learning

The Intergeo Project (WS)

Analogy and dynamic geometry software together in approaching 3d geometry

Expanding Room for Tacit Knowledge in Mathematics Education

Using Technology to Support Mathematics Teaching

Teaching Math With Advanced Learning Blocks

Flexible Mathematics Content Preparation

Designing Task for CAS Classrooms

Web Based Multimedia Symbolic Package (Linear Motion) Using a Computer Algebra System (CAS)

The use of notebooks in mathematics instruction. What is manageable? What should be avoided? A field report after 10 years of CAS-application

The Role of Dynamic Geometry Software in the Process of Learning: GeoGebra Example about Triangles

Visualization and Interface in Exploration of Functions

The Development of Immunology Educational Package using Computer Algebra System (CAS) Modelling

Categorizing CAS Use within One Reform-Oriented United States Mathematics Textbook

Critical Issues of Technology Use in Undergraduate Mathematics

On Assessment of Teaching A Mathematical Topic Using Neural Networks Models (with a case study)

Utilization of Free Software on Teaching Integral Function in the Schools of Economics

The Impact of Computer Use on the Teaching of Geometry in Grade 8

Folding and unfolding cones and cylinders with Cabri 3D: how to do it and how to use it in the classroom

3D-Dynamical Geometry in Building Construction

Those Magic Moments when you realise you could not do this with chalk! (WS)

Overcoming difficulties in understanding of the nonlinear programming concepts

A Computational Measure of Heterogeneity on Mathematical Skills

Using Maxima in the Mathematics Classroom

Applications of Multimedia Technology to study the ordinal thinking evolution of scholars from 3 to 7 years old

The Past and the Future of Computer Algebra and its Integration into Mathematics Education

Technology and the Yin&Yang of Mathematics Education

An example of learning based on competences: Use of Maxima in Linear Algebra for Engineers

Learning Math in the context of European Space for Higher Education

Lessons learned using Publishers'Web-based software to assess student work

Using intelligent adaptive assessment models for teaching mathematics

Taking advantage of Sherman's march

GeoGebra Workshop for the Initial Teacher Training in Primary Education

Problem solving using Geogebra

Dynamic Applets for Differential Equations and Dynamical Systems

There are some lectures more which are now involved in the reviewing process. You can find all full abstracts together with all Conference details at http://www.time2010.uma.es.